IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received February 18, 2020, accepted February 29, 2020, date of publication March 6, 2020, date of current version March 16, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2978893

A General Approach to Uniformly Handle Different String Metrics Based on Heterogeneous Alphabets

FRANCESCO CAUTERUCCIO[1], ALESSANDRO CUCCHIARELLI[2], CHRISTIAN MORBIDONI[2], GIORGIO TERRACINA[1], AND DOMENICO URSINO[2]

[1]Department of Mathematics and Computer Science, University of Calabria, 87036 Arcavacata di Rende, Italy
[2]Department of Information Engineering, Polytechnic University of Marche, 60121 Ancona, Italy

Corresponding author: Domenico Ursino (d.ursino@univpm.it)

ABSTRACT In the last few years, we have assisted in a great increase of the usage of strings in the most disparate areas. In the meantime, the development of the Internet has brought the necessity of managing strings from very different contexts and possibly using different alphabets. This issue is not addressed by the numerous string comparison metrics previously proposed in the literature. In this paper, we aim at providing a contribution in this context. In fact, first we propose an approach to measure the similarity of strings based on different alphabets. Then we show that our approach can be specifically adapted to several classic string comparison metrics and that each specialization can lead to addressing completely different issues.

INDEX TERMS String metrics, generalized string similarity framework, edit distance, Jaccard distance.

I. INTRODUCTION

Strings, i.e. ordered sequences of symbols, have been widely investigated in computer science. Indeed, their possible applications comprise the most disparate areas. For instance, with suitable semantics, they can be used to support the interaction and the cooperation of several systems. Research on string comparison and match dates back to 1977 [1] and periodically has received renewed interest as application areas evolve. For example, important advancements in the research on string comparison and match were brought about in code clone detection [2] in the 90s, and in bioinformatics [3] ten years later. Furthermore, data provided as streams of strings are increasingly exploited in modern areas, such as sensor networks and IoT contexts [4], and biomedical data analysis [5].

In all these scenarios, the common issue to address consists of determining how far two strings are correlated and/or how (dis)similar they are. As a consequence, in past literature, a large variety of techniques for string comparison have been proposed. In short, these techniques receive two strings s_1 and s_2 and return a value d representing how much s_1 and s_2 are (dis)similar based on some metric.

Actually, several (dis)similarity metrics have been presented in the literature; they greatly depend on the application context they have been thought for [6]. However, this wide set of metrics share a common principle, i.e. that identical symbols among strings represent identical information, whereas different symbols introduce some form of heterogeneity.

While this principle is correct in many contexts, it is going to be questioned in a global scenario, such as the current one, dominated by the Internet. In fact, data sources and systems coming from disparate contexts and locations, and hence characterized by different symbols and alphabets, must interact with each other.

For instance, consider the following scenario. Let $s_1 = \texttt{AABACCDC}$ and let $s_2 = \texttt{11213343}$. Any classical string similarity approach would conclude that s_1 and s_2 are completely different. However, a human would easily conclude that the structure underlying these two strings is identical and that the only difference regards the usage of different symbols.

This example clearly highlights that, in a global scenario, it should be possible to compare strings belonging to different alphabets. Examples of real cases in which this is necessary are numerous and constantly increasing; think, for instance, of two data streams returned by two sensors, the former measuring light in the day and the latter measuring temperature in the day. The values, the scales, and the meaning of these two sensors are totally different but, generally, there is a correlation between the corresponding measures. This correlation

The associate editor coordinating the review of this manuscript and approving it for publication was Noor Zaman.

could not be captured by a data stream management approach based on a classical string comparison method working on the discretized sensors values. As a further case, assume that alphabets of two strings have been deliberately modified to overcome string similarity controls; again, a classical string comparison approach fails in recognizing this fact.

All the previous considerations lead to the conclusion that a metric capable of comparing strings with different alphabets but having a similar structure may be beneficial in several application contexts [7].

Actually, some approaches to compare strings from different alphabets already exist in the literature. However, they address the slightly different problem of parameterized string match, looking for a parameterized match of a (short) pattern in a string. These approaches present some limitations on two orthogonal aspects, namely: *(i)* the function used to match symbols from different alphabets, and *(ii)* the metric used to compare strings. In particular, bijection is the most commonly adopted matching function [7], [8]. However, bijective functions allow only 1-1 matching and, in some cases, this may be not enough. For example, in the study of remote homologous proteins [9] (i.e., proteins having a pairwise sequence identity lower than 25%), it may be necessary to study protein similarities based on different kinds of amino acid property, such as chemical behavior or protein profiles, thus resulting in the necessity to allow many-to-many matches among amino acids. As a further example, bijective matching functions between words may miss common situations while comparing bilingual text corpora, where different words share the same meaning; for instance, English words *Hello* and *Goodbye* can be both interchangeably matched with Italian words *Ciao*, *Salve*, and *Addio*. As it will be clarified in the following, the approach proposed in this paper tries to overcome the limitations of bijective matching functions by allowing many-to-many matches. On the other hand, approaches for parameterized string match are tailored to specific metrics, like the Hamming distance (and its variants), the Longest Common Subsequence, the $\delta\gamma$ distance, or similar ones [7]. However, all of them share one common limitation, in that they are order-preserving comparison metrics, meaning that subsequent pairs of matched symbols must appear in the same relative order in both strings. As a consequence of this fact, they cannot identify similar strings where some portions have been exchanged. In some contexts, like the comparison of bilingual text corpora, grammatical rules may require different orders of corresponding words in a sentence; for instance, texts *"I like this article"* and *"Questo articolo mi piace"* represent the same sentence, even if word order is significantly different in each sentence. The adoption of different string comparison metrics, like the Jaccard distance, might be more appropriate in this case.

Actually, one of the goals of this paper is to go beyond classical parameterized string matching. Indeed, once a general approach (and a corresponding framework) to manage strings with different alphabets by means of many-to-many matching has been defined, it may be interesting to specialize

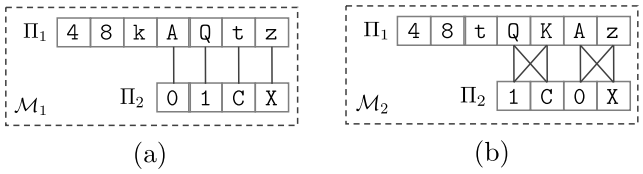

FIGURE 1. An intuitive example of two matching schema: (a) 1:1 matching; (b) n:m matching.

it to one or more string comparison metrics proposed in past literature. In this way, if we specialize it to a certain metric (for instance, edit distance), it can be exploited to address several application contexts, whereas, if we specialize it to another metric (say, Jaccard distance), it becomes well suited to face other application scenarios, and so forth.

In this paper, first, we present an approach to handle strings belonging to different alphabets (thus addressing the first form of generalization described above) supporting many-to-many matching. Then, we show how our approach, used to implement an "alphabet aware" variant of the classical Jaccard distance, can effectively address the problem of sentence alignment in comparable corpora. On the other hand, modifying the classical edit distance along the guidelines underlying our approach and allowing many-to-many matchings, we show the possibility to handle a complex heterogeneous sensor network scenario as well as the possibility to extract and characterize white matter fiber bundles from a brain.

Parameterized string match, largely investigated in the past literature, can be seen as a simplified version of the parameterized string comparison problem addressed in this paper. As a matter of fact, parameterized pattern matching focuses on answering the question: "Does the pattern P have a parameterized match in text T?", whereas our approach aims at answering the more general question: "Given a generic string similarity metric f, how (dis)similar are texts $T1$ and $T2$ using f and a many-to-many parameterized matching schema?".

As we can see below, in our approach a key role is played by the concept of matching schema. In order to illustrate what it is and to give an idea of the behavior of our approach, consider the following example, in which we are interested in the computation of the (dis)similarity between two strings, and we calculate it in two different ways. In fact, we first compute both Hamming and Jaccard distances; then, we carry out the same computations enhanced with the support of the new concept of matching schema. Let $s_1 = \texttt{AQQkk48zzttQ}$ and $s_2 = \texttt{01111CXXX0CC}$ be two strings defined over the alphabets $\Pi_1 = \{\texttt{4}, \texttt{8}, \texttt{A}, \texttt{Q}, \texttt{k}, \texttt{t}, \texttt{z}\}$ and $\Pi_2 = \{\texttt{0}, \texttt{1}, \texttt{C}, \texttt{X}\}$. Let us denote the Hamming and the Jaccard distances between s_1 and s_2 as $h(s_1, s_2)$ and $j(s_1, s_2)$, respectively. In this case, we have that $h(s_1, s_2) = 11$ and $j(s_1, s_2) = 1$; as a consequence, s_1 and s_2 should be considered completely dissimilar. Now, we enhance the computation of these two distances providing a matching schema. Intuitively, this is a mapping between Π_1 and Π_2 that states which symbols of s_1 can be considered matching with symbols of s_2. Figure 1 shows two

examples $\mathcal{M}_1$ and $\mathcal{M}_2$ of matching schemas. Here, a solid line indicates a match between two symbols. For instance, $\mathcal{M}_1$ indicates that the symbols `Q` and `1` match. Note that it is possible to have n-to-m matches; for instance, in $\mathcal{M}_2$, the symbol `Q` matches the symbols `1` and `C`.

We now recompute $h(s_1, s_2)$ and $j(s_1, s_2)$ by taking matching schemas into account. We denote the computation as $h_{\mathcal{M}_i}(s_1, s_2)$ and $j_{\mathcal{M}_i}(s_1, s_2)$, $i = 1, 2$. Using $\mathcal{M}_1$, we obtain $h_{\mathcal{M}_1}(s_1, s_2) = 6$ and $j_{\mathcal{M}_1}(s_1, s_2) = 0.43$. Instead, $\mathcal{M}_2$ gives $h_{\mathcal{M}_2}(s_1, s_2) = 4$ and $j_{\mathcal{M}_2}(s_1, s_2) = 0.27$.[1] Note how both Hamming and Jaccard distances consistently change based on the provided matching schemas. In other words, both distances decrease because of the constraint relaxation caused by the matching schema. In particular, if the first matching schema is valid, the Hamming distance states that s_1 and s_2 differ for six symbols, while the Jaccard distance indicates a medium similarity between these two strings. Instead, if the second matching schema (which is even more "permissive" than the first one) is assumed as valid, then both the Hamming and the Jaccard distances further decrease w.r.t. the first matching schema, and this behavior is right because it reflects the higher "permissivity".

This paper is organized as follows: in Section II, we present related literature. In Section III, we provide a general description of our approach. In Section IV, we propose a specialization of our approach to the Jaccard distance and present one possible application for bilingual sentence alignment in comparable corpora. In Section V, we specialize our approach to the edit distance and present two of the many possible applications, namely anomaly detection in wireless sensor networks and extraction of white matter fiber bundles of brain. Finally, in Section VI, we draw our conclusions and illustrate some possible developments of our approach.

II. RELATED WORK

String comparison and match has a long history in computer science literature [1]–[3]. In fact, there is a large number of applications in which string similarity computation plays a key role. In [10], a pattern recognition system based on sequence alignment to detect malware on a mobile operating system is developed; it focuses on suspicious boot sequences and compares legitimate and malicious system call sequences. Time series analysis often exploits string similarity techniques; for example, in [11], an empirical evaluation of similarity measures for time series classification is presented. This approach leverages measures such as Dynamic Time Warping and Edit Distance for Real sequences.

String similarity techniques and their derivatives have been adopted in different domains. For instance, in pattern recognition, they have been used to perform handwritten character recognition [12], to compute time series similarity [13], to manage graphs [14], [15], and to define kernel functions for pattern classification [16].

[1]Details for these computations are given later in the paper.

Surveys on classical string and text similarity approaches are available (see for instance [17], [18]). However, relevant literature for the present work regards scenarios where "exact match" is not enough for string matching. In literature, exact match has been extended by the notion of parameterized string matching. Here, a parameterized match is detected by consistently renaming texts and patterns with the help of a bijective mapping.

Parameterized string matching received growing interest in the literature; comprehensive surveys covering several solutions to this problem are available in [7], [19]. A seminal work on parameterized match is presented in [2]. Here, the author defines an approach to identify exact and approximate occurrences of a pattern in a text via parameterized strings, or *p-strings*. Symbols in p-strings are considered either constants or parameters. Identity match is replaced by no-cost substitutions of parameters, using *bijective global* transformation functions allowing *exact p-matches* only. The work in [20] is based on the approach proposed in [2]; it introduces an order-preserving match, but it limits the number of mismatches to k.

Some solutions to the parameterized string matching problem based on bijective matchings focus on efficiency. In particular, [8] surveys some approaches that exploit q-grams in order to achieve linear time complexity; this last feature is also obtained with the support of position heaps [21]. More recently, parameterized matching has been studied also in its compressed version [22], [23]. In particular, in [22], the authors aim at finding all of the parameterized matches between a pattern P and a text T using only P and a compressed version T_c of the text. In [23], compressed parameterized matching performs matching on text and patterns, both in compressed form, without decompressing any of them; it exploits Word Based Tagged Code (WBTC) for compression and Wavelet tree for efficient searching. In the present work we are not focused on efficiency, but rather on extending the range of aplications of parameterized string matching.

As for this last aspect, parameterized matching has applications in several areas. It was formerly used in software maintenance for code clone detection [24]–[26]; its extensions can be profitably exploited in image processing and computational biology [27]–[29], in music information retrieval [30], and even for solving the graph isomorphism problem [31]. While [2] introduces *exact* parameterized match, *mismatches* are studied in [32]. Here, the authors analyze the problem of finding all the locations in a string s for which there exists a *global bijection* π that maps a pattern p into the appropriate substring of s minimizing the Hamming distance. [33] extends the parameterized matching in such a way as to find approximate parameterized match under the weighted Hamming distance.

As pointed out in the Introduction, parameterized string matching approaches based on bijective mappings cover several application domains. However, there are cases in which bijection is not enough. As an example, in biology, it may be interesting to match amino acid sequences of proteins based

on their chemical characteristics. In particular, based on the propensity of the side chain of a protein to be in contact with water, amino acids can be classified as hydrophobic (which have a low propensity to be in contact with water), polar and charged (which are energetically favorable to the contact with water). In this context, a bijective matching between different amino acids may miss some relevant chemical correspondences; more general matching schemas may allow the accommodations of hydrophobic/polar/charged matchings among different amino acids. As another example, in the context of heterogeneous sensors data streams, similarity computation using bijective matching introduces several limitations. For instance, battery powered sensors may introduce slow, but constant, variations in sensed data during time in different sensors. In this case, many-to-many matching between sensed data would allow us to recognize important matches that could be missed by bijective matching functions. As a final example, in comparing texts from two different languages, bijective match functions may miss common situations in which different words in the two languages share the same meaning. One of the objectives of the present paper is precisely to overcome the limitations of bijective matching functions.

As a matter of fact, one of the research directions in parameterized string comparison relates to the function adopted to express parameterized match. As an example, *injective* functions, instead of the bijective ones used in [2], are considered in [34], whereas *generalized function matching* applied to the pattern matching problem is introduced in [35]. In [36], a frequency-based approach to a *many-to-many* mapping function for string alignment can be found; it computes alignments between two parameterized strings and gives preferences to alignments based on the co-occurrence frequency. The approach proposed in this paper generalizes all matching functions mentioned above and moves from the parameterized string matching problem to the more general problem of measuring string (dis)similarity with parameterized matching.

On the other hand, in past literature, several metrics to measure the similarity between two parameterized strings have been considered. Among them we cite the Longest Common Subsequence (LCS), the $\delta\gamma$ distance and the Edit/Hamming distance. The parameterized version of LCS has been studied in [37], where a proof of NP-Hardness is provided and an approximate algorithm is proposed. The *longest parameterized common subsequence* has been studied also in [38], where a quite different notion of parameterized string is adopted. Interestingly, LCS allows only insertions and deletions, but no substitutions.

The $\delta\gamma$ distance has been designed to compare equal length integer strings; they are said to $\delta\gamma$-match if their corresponding symbols differ at most by δ, and the sum of such differences is at most γ. This metric has found interesting applications in bioinformatics [27] and music information retrieval [30]. In [39], a parameterized version of the $\delta\gamma$ distance has been presented. In the context of parameterized Edit/Hamming distance, this problem has been stated as the identification of a proper transformation script allowing a parameterized string to be obtained from another. A simplified version of this problem considers parameterized matching under the Hamming distance, also called *parameterized matching with k mismatches* [32], [34].

In [25], the notion of *p-edit* distance is introduced. It focuses on the edit distance; allowed edit operations are *insertions*, *deletions* and *exact p-matches*. Mismatches are not allowed. Furthermore, two substrings that participate in two distinct exact p-matches are independent of each other, so that mappings have local validity over substrings not broken by insertions and deletions. In particular, within each of these substrings, the associated mapping function must be bijective.

The work presented in [37] extends the approach proposed in [25] by requiring the transformation function to have a global validity; however, it still limits the set of allowed edit operations (in particular, substitutions are not possible). Finally, in [36], the full edit distance (allowing insertions, deletions and substitutions) is considered, but parameterized symbol matches are detected by considering frequency-based co-occurrences of symbols in the strings; here NP-Hardness of the problem is proved and an approximate algorithm is proposed.

All the metrics mentioned above share a common limitation in that they require that matching symbols preserve their order of appearance in the corresponding strings. However, there are common and interesting situations in which the order of matching symbols can be different in the corresponding strings; think, for instance, of the comparison of sentences in bilingual text corpora: grammar rules could force matching words to be positioned in different order within the sentences. This is another limitation of past approaches, and our approach aims at proposing a solution also in this setting.

The related literature outlined above, and our considerations about its limitations, point out the wide variety of metrics, mapping functions and applications of interest in the research addressed by this paper. One of the main contributions of our work is a general framework which can be easily specialized to different metrics, covers a wide range of mapping functions and, consequently, can be applied to very different contexts with very limited specialization steps. Furthermore, the proposed approach explores metrics and mapping functions not considered in past literature, so as to further extend the range of possible application scenarios. As a matter of fact, in this paper we show an application of our framework to the alignment of sentences in bilingual comparable corpora, to the analysis of sensor networks and to the extraction of white matter fiber bundles; not all of these contexts have been previously addressed by parameterized match literature.

III. DESCRIPTION OF OUR APPROACH

In this section, we provide a general description of our approach. Its key components are matching schemas and a generalized metric function. Given two input strings s_1 and

s_2, a set $\mathcal{M}$ of possible matching schemas and a generalized metric function $f(\cdot,\cdot)$, our approach aims at computing the minimum value of $f(s_1, s_2)$ which can be obtained by applying the different matching schemas of $\mathcal{M}$.

We start by defining the first core component of our framework, i.e., the matching schema.

Let Π_1 and Π_2 be two (possibly disjoint) *alphabets* of symbols and let s_1 and s_2 be two strings defined over Π_1 and Π_2, respectively. We call *length* of a string s_i ($i \in \{1, 2\}$), denoted by $len(s_i)$, the number of its symbols. Furthermore, for each position $1 \leq j \leq len(s_i)$, we indicate with $s_i[j]$ the j^{th} symbol of s_i.

Given an alphabet Π and an integer π such that $0 < \pi \leq |\Pi|$, we call π*-partition* a partition Φ^π of Π such that $0 < |\phi| \leq \pi$, for each $\phi \in \Phi^\pi$.

Given two alphabets Π_1 and Π_2 and two integers π_1 and π_2, we call $\langle \pi_1, \pi_2 \rangle$***-matching schema*** a function $M_{\langle \pi_1,\pi_2 \rangle} : {\Phi_1}^{\pi_1} \times {\Phi_2}^{\pi_2} \rightarrow \{true, false\}$, where ${\Phi_i}^{\pi_i}$ ($i \in \{1, 2\}$) is a π_i-partition of Π_i and, for each $\phi_v \in {\Phi_1}^{\pi_1}$ (resp., $\phi_w \in {\Phi_2}^{\pi_2}$), there is at most one $\phi_w \in {\Phi_2}^{\pi_2}$ (resp., $\phi_v \in {\Phi_1}^{\pi_1}$) such that $M(\phi_v, \phi_w) = true$. This means that all the symbols in ϕ_v match with all the ones in ϕ_w. $M(\phi_v, \phi_w) = false$ indicates that all the symbols in ϕ_v mismatch with all the ones in ϕ_w. Intuitively, given two strings s_1 and s_2 defined over Π_1 and Π_2, M states which symbols of s_1 can be considered matching with symbols of s_2. Many-to-many matches are expressed with π-partitions, and partitions disallow ambiguous matches.

Example 1: Let $\Pi_1 = \{3, \mathtt{K}, \mathtt{Z}, \mathtt{g}\}$ and $\Pi_2 = \{0, 1, \mathtt{a}, \mathtt{x}\}$. Let $s_1 = \mathtt{KggZ333Z}$ and $s_2 = \mathtt{101xxxaa}$. For $\pi_1 = \pi_2 = 2$, one (of the many) possible matching schema(s) is $\{\{\mathtt{K}, \mathtt{g}\}\text{-}\{0, 1\}, \{3, \mathtt{Z}\}\text{-}\{\mathtt{a}, \mathtt{x}\}\}$. It is worth observing how the matching schema changes depending on the values of π_1 and π_2. For instance, for $\pi_1 = \pi_2 = 1$, a possible matching schema is $\{\{3\}\text{-}\{1\}, \{\mathtt{K}\}\text{-}\{\mathtt{a}\}, \{\mathtt{Z}\}\text{-}\{\mathtt{x}\}, \{\mathtt{g}\}\text{-}\{0\}\}$. □

Observe that, given Π_1, Π_2, π_1 and π_2, many possible matching schemas can be defined. In some contexts, it may be useful to limit *valid* matching schemas through some constraints.

Formally speaking, a *constraint* χ associated with a matching schema $M_{\langle \pi_1,\pi_2 \rangle}$ is a set of unordered pairs of symbols (c_i, c_j), such that $c_i \in \Pi_1$, $c_j \in \Pi_2$ and, for each $(c_i, c_j) \in \chi$, there exists no pair (ϕ_v, ϕ_w), $\phi_v \in {\Phi_1}^{\pi_1}$, $\phi_w \in {\Phi_2}^{\pi_2}$, having $c_i \in \phi_1, c_j \in \phi_2$ and $M(\phi_1, \phi_2) = true$. A $\langle \pi_1, \pi_2, \chi \rangle$-*constrained matching schema* is represented by $M_{\langle \pi_1,\pi_2,\chi \rangle}$ only if $\chi \neq \emptyset$.

Example 2: Continuing the previous example, if we have $\chi = \{(\mathtt{K}, \mathtt{a}), (\mathtt{Z}, 0), (\mathtt{g}, 1)\}$, the matching schema $\{\{\mathtt{K}, \mathtt{g}\}\text{-}\{0, 1\}, \{3, \mathtt{Z}\}\text{-}\{\mathtt{a}, \mathtt{x}\}\}$ is no longer valid, whereas the matching schema $\{\{\mathtt{K}, \mathtt{Z}\}\text{-}\{1, \mathtt{x}\}, \{3, \mathtt{g}\}\text{-}\{0, \mathtt{a}\}\}$ is a valid one. □

In the following, whenever it is clear from the context, for the sake of simplicity, we avoid writing $M_{\langle \pi_1,\pi_2,\chi \rangle}$, and we simply denote it by M.

The generalizability of the proposed framework is based on the fact that it can generalize any string metric based on the assumption of symbol identity. Therefore, in defining it, we are not interested in a particular function specification. To highlight this fact, we formally introduce the concept of generalized metric function. Given a metric function $f(\cdot,\cdot)$, based on symbol identity, and given a valid constrained matching schema M, the ***generalized metric function*** ${f^M}_{\langle \pi_1,\pi_2,\chi \rangle}(\cdot,\cdot)$ (or simply ${f^M}(\cdot,\cdot)$) is obtained from $f(\cdot,\cdot)$ by substituting symbol identity with the symbol matchings defined in M.

The last definition, necessary before formalizing our approach, is the one of generalized distance. Given two strings s_1 and s_2 over Π_1 and Π_2, respectively, and given the set $\mathcal{M}$ of valid constrained matching schemas, the *generalized distance* $F(s_1, s_2)$ between s_1 and s_2 returns the minimum value returned by ${f^M}(s_1, s_2)$ which can be obtained by taking any possible matching schema M of $\mathcal{M}$. Formally:

$$F(s_1, s_2) = \min_{M_{\langle \pi_1,\pi_2,\chi \rangle} \in \mathcal{M}} \{{f^M}(s_1, s_2)\} \tag{1}$$

We are now able to formalize our approach. It can be represented as a tuple $\mathfrak{A}$ consisting of five components:

$$\mathfrak{A} = \langle \Pi_1, \Pi_2, \langle \pi_1, \pi_2, \chi \rangle, \mathcal{M}, {f^M}(\cdot,\cdot) \rangle \tag{2}$$

Here, Π_1 and Π_2 are the alphabets on which the strings under consideration are defined, $\langle \pi_1, \pi_2, \chi \rangle$ are the parameters necessary to define valid matching schemas, $\mathcal{M}$ is the set of all the valid constrained matching schemas over Π_1 and Π_2, and ${f^M}(\cdot,\cdot)$ is the generalized metric function. When applied on two strings s_1 and s_2 over the alphabets Π_1 and Π_2, respectively, $\mathfrak{A}$ returns the value of $F(s_1, s_2)$, where $F(s_1, s_2)$ is the generalized distance of s_1 and s_2 over $f(\cdot,\cdot)$ and $\mathcal{M}$.

IV. SPECIALIZATION OF OUR APPROACH TO THE JACCARD DISTANCE

In this section, we show how $\mathfrak{A}$ can be applied to generalize the Jaccard Distance in such a way that symbol identity is substituted by many-to-many symbol correlations, where identifying the best matching schema is part of the problem. In particular, we next introduce the Multi-Parameterized Jaccard Distance (MPJD).

A. DESCRIPTION OF THE SPECIALIZATION ACTIVITY

In order to define MPJD, we must preliminarily introduce some concepts. The Jaccard Distance is defined over sets of elements and is generally applied to measure (dis)similarities between texts. In our context, tokens in a text play the role of the symbols of our framework. Let t_i be a text, then Π_i consists of the set of tokens of t_i. The classical Jaccard Distance measures the (dis)similarity between two texts t_1 and t_2 by means of the following formula:

$$\mathcal{J}(t_1, t_2) = \frac{|\Pi_1 \cup \Pi_2| - |\Pi_1 \cap \Pi_2|}{|\Pi_1 \cup \Pi_2|} \tag{3}$$

Here, $\cup$ and $\cap$ are the usual union and intersection operators on sets. The discriminating factor to compute them is based on the question: "Is $\Pi_1[i] = \Pi_2[j]$?", where $\Pi[i]$

is the i-th element of Π. In our framework, this question is substituted by the following one: "According to M, do tokens $\Pi_1[i]$ and $\Pi_2[j]$ match?".

After this premise, we can introduce the notion of MPJD between two texts t_1 and t_2.

Specifically, let t_1 and t_2 be two texts and let Π_1 and Π_2 be the corresponding sets of tokens; let π_1 and π_2 be two integers such that $0 < \pi_1 \leq |\Pi_1|$ and $0 < \pi_2 \leq |\Pi_2|$. The ***Multi-Parameterized Jaccard Distance*** between t_1 and t_2 ($\mathcal{J}_{\langle\pi_1,\pi_2,\chi\rangle}(t_1, t_2)$, for short) is the minimum distance that can be obtained with any $\langle\pi_1, \pi_2, \chi\rangle$-constrained matching schema. Formally:

$$\mathfrak{A}_{\mathcal{J}} = \langle \Pi_1, \Pi_2, \langle\pi_1, \pi_2, \chi\rangle, \mathcal{M}, \mathcal{J}^M(\cdot,\cdot)\rangle \tag{4}$$

and

$$F_{\mathcal{J}}(t_1, t_2) = \mathcal{J}_{\langle\pi_1,\pi_2,\chi\rangle}(t_1, t_2) = \min_{M_{\langle\pi_1,\pi_2,\chi\rangle}\in\mathcal{M}}\{\mathcal{J}^M(t_1, t_2)\} \tag{5}$$

where $\mathcal{J}(\cdot,\cdot)$ is the classical Jaccard distance.

Now, while for the case $\pi_1 = \pi_2 = 1$ the semantics of the union and the intersection operators when using our framework is obvious, since each element of the first set matches with at most one element of the second set, some considerations must be drawn when $\pi_i = 2$ or higher. In fact, in this case, one element of the first set (resp., one element of the second set) might match with more elements of the second set (resp., more elements of the first set). In order to better understand this fact, consider the following example. Let $\Pi_1 = \{A, B, C\}$ and let $\Pi_2 = \{1, 2\}$. For $\pi_1 = \pi_2 = 1$, a possible matching schema is `{{A}-{1}, {B}-{2}}` and, consequently, we can use either A or 1 (resp., either B or 2) when expressing their identity in $\Pi_1 \cup \Pi_2$. In particular, $\Pi_1 \cup \Pi_2 = \{\{A|1\}, \{B|2\}, \{C\}\}$, and, thus, $|\Pi_1 \cup \Pi_2| = 3$; similarly, $\Pi_1 \cap \Pi_2 = \{\{A|1\}, \{B|2\}\}$, and $|\Pi_1 \cap \Pi_2| = 2$. Now, if $\pi_1 = \pi_2 = 2$, a possible matching schema is `{{A,B}-{1,2}}`, meaning that we can use both $\{A|1\}$ and $\{A|2\}$ (resp., $\{B|1\}$ and $\{B|2\}$) to represent them in the union set. As a consequence, we can assume that $\Pi_1 \cup \Pi_2 = \{\{A|1\}, \{A|2\}, \{B|1\}, \{B|2\}, \{C\}\}$, and, thus, $|\Pi_1 \cup \Pi_2| = 5$; similarly, $\Pi_1 \cap \Pi_2 = \{\{A|1\}, \{A|2\}, \{B|1\}, \{B|2\}\}$, and $|\Pi_1 \cap \Pi_2| = 4$.

Actually, in real cases, the extension presented above for the Jaccard distance, when applied to a single pair of texts and without constraints on the matching schemas, may have a limited relevance. In fact, since the Jaccard distance is based only on the presence of tokens in the texts, without considering their relative positions or frequency, the obvious matching schema that minimizes the distance is the one that matches as many tokens as possible. Consequently, the MPJD would depend only on the number of tokens in the text. In particular, every pair of texts with the same number of tokens would share the same distance, independently of their contents and structures. A much more interesting scenario arises when we want to compare *several* pairs of texts to determine the best *common* matching schema that minimizes some objective function defined on the overall set of pairs.

This further extension can be simply obtained in our framework by extending $F(\cdot,\cdot)$ in such a way as to handle sets of elements, instead of single strings/texts.

As an example, let T_1 and T_2 be two ordered sets of texts; if we want to obtain the minimum *average* Jaccard distance between the pairs of texts in the same position in T_1 and T_2, using a common matching schema M, the function $F(\cdot,\cdot)$ can be modified as follows:

$$\begin{aligned} &F_{\mathcal{J}}(T_1, T_2) \\ &= \mathcal{J}_{\langle\pi_1,\pi_2,\chi\rangle}(T_1, T_2) \\ &= \min_{M_{\langle\pi_1,\pi_2,\chi\rangle}\in\mathcal{M}}\{avg\{\mathcal{J}^M(t_1, t_2)|\forall i, t_1 = T_1[i], t_2 = T_2[i]\}\} \end{aligned} \tag{6}$$

In this framework, even more complex objective functions can be accommodated. They may find interesting applications in several contexts. In particular, let T_1 and T_2 be two sets of texts; we might be interested in determining the best pairs of matching texts between T_1 and T_2, regardless of their position in T_1 and T_2, based on their dissimilarity. Determining the best pairs of matches can be obtained by running a minimum weighted bipartite matching algorithm (*mwbm*, for short). In more detail, let M be a matching schema; a complete bipartite graph G can be built from T_1 and T_2 by introducing one node for each text in T_1 and T_2. G contains an arc between each $t_i \in T_1$ and $t_j \in T_2$ and the weight of this arc is set to $\mathcal{J}^M(t_i, t_j)$. The application of a function $mwbm_{\mathcal{J}^M}(T_1, T_2)$ that computes the *mwbm* over this graph, would then find the best pairs of matching texts allowing us to obtain, overall, the minimum Jaccard distances between them, given a matching schema M.

In order to determine the best pairs of matching texts over all possible matching schemas, $F(\cdot,\cdot)$ can be extended as follows:

$$\begin{aligned} F_{\mathcal{J}}(T_1, T_2) &= \mathcal{J}_{\langle\pi_1,\pi_2,\chi\rangle}(T_1, T_2) \\ &= \min_{M_{\langle\pi_1,\pi_2,\chi\rangle}\in\mathcal{M}}\{mwbm_{\mathcal{J}^M}(T_1, T_2)\}. \end{aligned} \tag{7}$$

B. APPLICATION TO SENTENCE ALIGNMENT IN BILINGUAL COMPARABLE CORPORA

In this section, we show how the MPJD, discussed in detail in the previous section, can be applied to the problem of aligned sentence identification in comparable corpora, and we present the results of some experiments carried out in order to check the effectiveness of this measure. To the best of our knowledge, this is the first attempt to apply parameterized matching in such a context.

1) BACKGROUND

According to [40], comparable corpora are defined as collections of documents that are comparable in content and form in various degrees and dimensions. This definition also includes parallel multilingual corpora, i.e. sets of pairs of

multilingual documents that are translations of each other (with full sentences alignment). Parallel corpora are valuable resources both in language engineering and in linguistic research areas. In the former case, they are widely used as training and/or test data in statistical machine translation [41] and in cross-lingual retrieval methods [42]; in the latter case, they are the basis of inter-linguistic analysis and language comparison [43].

Many approaches to bilingual sentence alignment in comparable corpora are based on the idea of performing raw sentence translations, using a bilingual lexicon, and then measuring token-based similarity among sentences (see, for instance, [44]). In [45], the authors propose an approach to estimate cross-lingual document similarity and to generate dictionaries with comparable corpora. This approach has two main goals, namely bilingual dictionary generation and cross-lingual similarity estimation. To reach these objectives, it exploits kernel approximation on word embeddings, which are then used to estimate the cross-lingual document similarity. In a recent challenge focusing on the task of extracting parallel sentences from comparable corpora [40], the best approach, described in [46], is based on learning a bilingual dictionary (using a well known tool, i.e., GIZA++) and then using bidirectional Jaccard index to measure similarity among translated sentences. This work extends the STACC approach [44], [47] and outperforms methods based on bilingual neural word embeddings [48] that were considered, in recent literature, the best ones for this task. A similar idea is used in context-based similarity approaches (e.g. [49]) where words are represented by their contexts (co-occurring words) in the source and target languages. A bilingual dictionary is used to project terms from one language to the other, and similarity scores (e.g. cosine and Jaccard distances) are used to find the best alignment. It is worth pointing out that, in general, the best performing approaches are the hybrid ones, combining multiple features, such as sentence length, distance among sentences to be aligned, and so on. String similarity metrics are often used in such methods. For example, in [50], variants of the edit distance are applied to derive bilingual token alignments and Text Entailments (TE) within a monolingual corpus. This allows sentences to be clustered based on their meaning and, in turn, facilitates the sentence alignment task [51].

2) BILINGUAL SENTENCES COMPARISON

The core of our approach is the comparison between two sentences extracted from an English-Italian (en-it) comparable corpus. Let us introduce our method with an example, by considering two scenarios: in the former one, we compare the two sentences by using matches between en-it tokens included only in an input dictionary; in the latter one, we compare the two sentences by using matches included both in the input dictionary and in a matching schema.

Let S_{en} and S_{it} be the two sentences, where S_{en} (resp. S_{it}) is the English (resp. Italian) one, and S_{it} (resp., S_{en}) is the translation of S_{en} (resp., S_{it}). The input dictionary D contains

TABLE 1. Lemmas $\{t_{\text{en}}\}$-$\{t_{\text{it}}\}$ present in D.

Pairs $\{t_{\text{en}}\}$-$\{t_{\text{it}}\}$
{if}-{se}, {the}-{il}
{have}-{avere}, {agree}-{accordo}
{do}-{fare}, {as}-{secondo}

the pairs $\{t_{\text{en}}\}$-$\{t_{\text{it}}\}$ such that t_{en} (resp. t_{it}) is an English (resp. Italian) token, and t_{it} (resp., t_{en}) is the translation of t_{en} (resp., t_{it}). Thus, $\{t_{\text{en}}\}$-$\{t_{\text{it}}\}$ is a matching.

For instance, assume that the two sentences S_{en} and S_{it} are:

S_{en} : `If the House agrees, I shall do as Mr Evans has suggested.`

S_{it} : `Se l'Assemblea è d'accordo seguirò il suggerimento dell'onorevole Evans.`

Now, we compare the two sentences in the first scenario, in which only the input dictionary D is used. Table 1 shows the matches in D appearing in S_{en} and S_{it}. Here, only matches in D concur to the computation of the distance: the identity between symbols is exclusively denoted by pairs present in D.

In the second scenario, suppose we have the following matching schema, inferred by the application of our technique:

$$M_1 = \{\{\texttt{suggested}\}\text{-}\{\texttt{suggerimento}\}, \{\texttt{Mr}\}\text{-}\{\texttt{onorevole}\}, \{\texttt{I}\}\text{-}\{\texttt{seguirò}\}.\{\texttt{House}\}\text{-}\{\texttt{Assemblea}\}\}.$$

By exploiting both the matches in D and the ones in M_1, we observe that the Jaccard distance between S_{en} and S_{it} would decrease because of a higher number of matches between symbols than in the first scenario. It is worth pointing out that M_1 does not carry any semantic information related to contained matches; for instance, the pair {i}-{seguirò} does not represent a valid match from a linguistic point of view, although it concurs as a valid match to the computation of the distance between S_{en} and S_{it}. However, when a large set of sentences in a comparable corpus is analyzed through our method, pairs of tokens frequently occurring together in aligned sentences will be preferred because, overall, they will allow lower Jaccard distances in the minimum weighted bipartite matching.

Summarizing, the application of our approach to bilingual sentences alignment in comparable corpora is carried out as follows. Given T_{en} and T_{it}, the sets of sentences to be aligned and an input dictionary D, the application of $F_{\mathcal{J}}$ on T_{en} and T_{it} with the minimum weighted bipartite matching yields the pairs of sentences that should be considered one the translation of the other. Setting the parameters π_1 and π_2 to 0 implies that only the dictionary D is exploited to compute the Jaccard distance. Instead, setting π_1 and π_2 to values higher than 0 activates a sort of *dictionary enrichment* through our approach, which determines the best matching schemas.

C. EXPERIMENTS

The aim of our experiments is to show how the MPJD can be effectively applied to compare bilingual sentences. We start using an existing bilingual dictionary as the only source of matches between a subset of word lemmas in the source language and the corresponding lemmas in the target language. Then, we show that, by using MPJD to automatically deduce a matching schema (i.e., by adding matches not present in the original dictionary), the performances of the unsupervised sentence alignment increase.

1) TEST CORPORA AND DICTIONARY

In order to test our approach, we created a set of comparable corpora starting from the Europarl parallel corpora.[2] This is a set of parallel documents in the 21 European languages extracted from the proceedings of the European Parliament Sessions [52], which is widely used in the automatic translation research area. For our experiments, we focused on the English-Italian language-pair (*en-it*). To obtain a comparable pair of documents from the parallel corpus, we defined a source document by randomly selecting a set S of sentences from Europarl and we paired it with a target document T containing both sentences being translations of the source and sentences representing noise. To test the effectiveness of the parameterized string comparison in computing the Jaccard similarity in different conditions, we generated 8 different comparable pairs. All the sources have the same value as S, whereas the related targets differ in T (we use 0.6, 1 and 1.4 as values for the T/S ratio), as well as in the ratio between the sentences that are actually translations of sentences in the source, T_s, and noise, T_n (1, 2.5 and 10 for signal-to-noise ratio SNR: T_s/T_n).

In Table 2 we list the characteristics of the 8 generated corpora. For example, in the corpus *C1*, we have a target smaller than the source and a number of aligned sentences equal to the number of noisy ones (600); in *C6*, the target size is equal to the source size; the corpus contains 1819 aligned sentences and 181 noisy ones, Finally, in *C8* the target corpus is greater than the source and contains the translation of all the source sentences. The 8 different corpora structures, obtained by using the selected parameters values, match with a good approximation the ones of the corpora commonly used in real alignment tasks [53]. Note that when T/S is set to 1.4, the maximum possible value of S/N is 2.5.

The English-Italian dictionary used in our experiment is the one available from the Apertium project.[3] As stated in the documentation, this dictionary was extracted from Europarl corpus using the popular GIZA++ tool and contains correspondences between about 22000 Italian and English lemmas.

Before using our test corpora in the experiments, we performed some simple, pretty standard pre-processing. In particular, we removed the empty lines in the original Europarl corpora and we replaced upper-case letters at the beginning of each sentence with the corresponding lower-case ones. Then, we replaced each word with the corresponding lemma. This is a mandatory step in order to correctly match dictionary entries. For this purpose, we used the TreeTagger tool,[4] described in [54] and [55].

The Apertium it-en dictionary provides a partial, but relatively good, coverage of the lemmas used in our test corpora. In fact, on average, it covers about 75% of the lemmas appearing in the English documents and about 66% of the lemmas in the Italian ones.

TABLE 2. Overview of the generated comparable corpora.

Corpus	S	T/S	SNR	T	T_s	T_n
C1	2000	0.6	1	1200	600	600
C2	2000	0.6	2.5	1200	858	342
C3	2000	0.6	10	1200	1091	109
C4	2000	1	1	2000	1000	1000
C5	2000	1	2.5	2000	1429	571
C6	2000	1	10	2000	1819	181
C7	2000	1.4	1	2800	1400	1400
C8	2000	1.4	2.5	2800	2000	800

TABLE 3. Results of the sentences alignment experiments under different conditions.

	Precision			*Recall*			*F1-Score*		
Corpus	*s1*	*s2*	*s3*	*s1*	*s2*	*s3*	*s1*	*s2*	*s3*
C1	0.67	0.70	0.72	0.76	0.77	0.78	0.71	0.73	0.75
C2	0.69	0.74	0.76	0.74	0.75	0.76	0.71	0.74	0.76
C3	0.75	0.78	0.80	0.68	0.71	0.73	0.71	0.74	0.76
C4	0.66	0.70	0.71	0.78	0.79	0.82	0.72	0.74	0.76
C5	0.70	0.74	0.75	0.75	0.77	0.80	0.72	0.75	0.77
C6	0.74	0.77	0.78	0.71	0.76	0.78	0.72	0.76	0.78
C7	0.64	0.68	0.71	0.78	0.83	0.84	0.70	0.75	0.77
C8	0.67	0.73	0.75	0.76	0.80	0.82	0.71	0.76	0.78

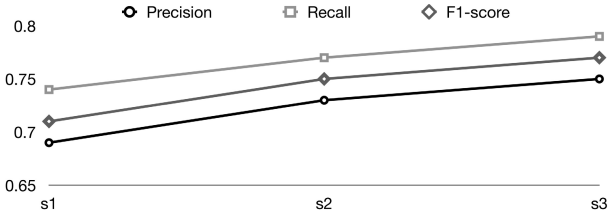

FIGURE 2. Average Precision, Recall and F1-score in the three experimental settings (s1, s2 and s3).

2) RESULTS AND DISCUSSION

In Table 3, we summarize the results of our experiments. For each document pair, we measure the *Precision*, as the fraction of sentence pairs aligned by our method that are really translations of each other, the *Recall*, as the fraction of real translations that are correctly detected by our method, and the *F1-score*, combining Precision and Recall using the harmonic mean.

[2] http://www.statmt.org/europarl/
[3] https://github.com/apertium/apertium-en-it/
[4] http://www.cis.uni-muenchen.de/~schmid/tools/TreeTagger/

For each corpus, we ran experiments using MPJD with three different settings, labeled as *s1*, *s2* and *s3* (see Table 3). In the setting *s1*, we used only matches directly provided by the bilingual dictionary, whereas in *s2* and *s3* we considered additional matches between Italian and English lemmas. More precisely, in *s2* we allowed only 1:1 matches, whereas in *s3* we allowed 2:2 matches as well. With respect to the MPJD definition given in the previous section, *s2* corresponds to the case $\pi_1 = \pi_2 = 1$ and *s3* to the case $\pi_1 = \pi_2 = 2$.

From the analysis of Table 3 we can observe that, as expected, when the percentage of noisy sentences is high (e.g. *C1*, *C4* and *C7*) the alignment task is harder to accomplish; by contrast, when noisy sentences are rare (as in *C3*, *C6* and *C8*), we obtained better performances. As the dictionary provides a relatively good coverage of lemmas, performances with the *s1* settings are already reasonable, showing a F1-score between 0.70 and 0.72. Figure 2 illustrates the trend of the average Precision, Recall and F1-score across the three experimental settings (*s1*, *s2* and *s3*). The results clearly evidence that MPJD increases both Precision and Recall in all considered corpora. Inferred 1:1 matches (*s2*) increase F1-score by 2 to 5 percentage points, and even better results are obtained when inferring 2:2 matches, in which a value of 0.78 is reached. By averaging the performances over all the corpora, we can see that our approach increases F1-score by 6%, which we consider an interesting result.

It is worth pointing out, again, that our approach does not take semantic or linguistic aspects of words into account because its aim is not to augment the original bilingual dictionary. While we cannot expect all inferred matches to be correct translations of lemmas, by looking at the resulting alignments we notice that many of them do in fact make sense, proving that our approach is really able to capture some of the corresponding terms of different languages.

In Table 4, some randomly extracted alignment examples are shown. Within each sentence, the same superscript is associated with English/Italian lemmas that were not paired in the original dictionary and were inferred thanks to our approach. Bold and underlined lemmas represent correct translations, or words having a close meaning in the two languages, while bold ones (without underline) represent matches having no clear meaning. We can see that lemmas such as *understandable* and *comprensibile*, or *confine* and *limitare*, are exact translations. Other terms, like *nationalization* and *rinazionalizzazione*, or *test* and *vaglio*, clearly have a similar meaning.

An example of a 2:2 match is present in the second sentence: *(design, make)* is matched with *(inteso, rendere)*. In this context, two multi-word expressions (*designed to make* and *inteso a rendere*), which have the same meaning, are captured. Other 2:2 matches, where the same English term is matched with different Italian terms in different sentences (an vice versa), appear in the results. Due to space constraints, they are not shown in Table 4.

As a final consideration, we want to highlight how, in absolute terms, the performances of our approach for sentence alignment is barely comparable with the state-of-the-art systems mentioned in the literature (see, for example, [46]). This result was expected, given that we used an approach founded on a single metric, i.e., the Jaccard distance, whereas the most effective systems for sentence alignment are based on multiple combined metrics. However, applying it to the specific domain of aligned corpora construction, the general approach to compute string similarity based on different alphabets has proven to be effective in implementing one of the most widely used measure.

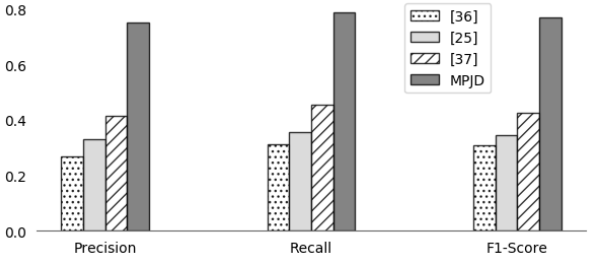

FIGURE 3. Average Precision, Recall and F1-score for each method indicated in Section IV-D.

D. COMPARISON OF MPJD WITH OTHER METRICS

In this section we compare the performance of MPJD with other existing metrics based on parameterized string similarity. In particular, as pointed out in [7] parameterized string matching has been exploited in some approaches to compute the similarity between pairs of parameterized strings. Specifically, extensions of the Hamming distance [32], [34], the Edit distance [25], [36], the $\delta\gamma$ distance [30], and the LCS [37], [38], [56] have been investigated. In our experiments we do not consider the approaches based on the $\delta\gamma$ distance, because they are tailored to numerical sequences, and approaches based on the Hamming distance, because they require that compared strings are of equal lengths. Therefore, we focus on the parameterized edit distance, as computed in [25], which is based on bijective parameterized matches, and its variant proposed in [36], where many-to-many parameterized matches are identified, based on the frequency of co-occurring symbols. Moreover, we consider the Longest Common Parameterized Distance, as computed in [37].

In order to carry out a fair comparison, we adapted the compared methods in order to treat each lemma as a symbol; moreover, we considered matches directly provided by the bilingual dictionary as known matches for all tested methods. We tested MPJD with the setting *s3* introduced above. For each method we computed the average Precision, Recall and F1-Score on the corpora presented in the previous section. Results are shown in Figure 3.

This figure shows that the approach described in [36] is the worst performing one. As a matter of fact, even if it allows many-to-many matches, these are inferred through the frequency of co-occurring symbols; clearly, the same lemma does not appear frequently in a sentence and, consequently, inferred matchings are not meaningful in this application

TABLE 4. Examples of sentences aligned by our method; inferred matching lemmas are denoted by the same superscript.

Language	Lemmatized sentence
en	we shall therefore **confine**[1] ourselves to **highlight|highlighting**[2] a few **issue**[3]
it	ci **limitare**[1] pertanto a **sottolineare**[2] alcun|alcuni **aspetto**[3]
en	all of **these**[1] measure be **design**[2] to **make**[2] it **more**[3] coherent **understandable**[3] and flexible
it	si **trattare**[1] di misura **inteso**[2] a **rendere**[2] il nostro **legislazione**[3] piú coerente piú **comprensibile**[3] e piú flessibile
en	the new model now be **test|tested**[1] by the commission ought not to **lead**[2] to a process of **nationalisation**[3] pure and simple which would **undermine**[4] the established|establish **competition**[5] **policy**[6]
it	il nuovo modello ora al **vaglio**[1] del commissione non dovere tuttavia **condurre**[2] a un mero processo di **rinazionalizzazione**[3] che minare al base il **risultato**[4] sin qui **conseguire**[5] dal **politica**[6] del concorrenza
en	if you **feel**[1] that these **session**[2] after a commission statement be important i would suggest that we require more time than the **half-hour**[3] that be allocate|allocated to they
it	se **ritenere**[1] che il spazio **dedicare**[2] al domanda dopo una dichiarazione del commissione sia importante suggerire di chiedere più tempo rispetto al **mezzora**[3] che essere esser|stare prevedere

context. Actually, the approach of [36] was conceived for the analysis of long sequences of data. The approach of [25] performs slightly better than the one of [36], but the performances are still quite low. In this case, the main motivation relies on the edit distance, which penalizes distant matching symbols, because deletions have a cost in terms of computed distance.

The Longest Common Parameterized Distance [37] resulted to be the best performing one among the competitors of MPJD; this can be motivated by the fact that deletions are not penalized. However, since LCS requires that the relative order of matching symbols is preserved, it misses all those cases in which word order in the two languages is different due to grammar rules.

Finally, it is worth observing also that, while the approaches of [25], [36], [37] identify symbol matches *locally* for each pair of sentences, MPJD finds a matching schema that maximizes matches among lemmas in the entire corpus, possibly allowing many-to-many matches. Furthermore, MPJD does not consider the relative order of lemmas in the sentences and, consequently, it is able to handle word order swaps caused by the different grammar rules of the two languages. These are the main motivations underlying the significantly higher performance of MPJD with respect to the other tested approaches. The results obtained substantiate the usefulness of a framework in which many-to-many matches are allowed and almost any comparison metric can be easily accommodated in order to fit the application context at hand.

V. SPECIALIZATION OF OUR APPROACH TO THE EDIT DISTANCE

In this section, we show how $\mathfrak{A}$ can also be applied to generalize the edit distance. Again, also in this task, symbol identity is substituted by many-to-many symbol correlations, where identifying the best matching schema is part of the problem. As a result of this task, we introduce the Multi-Parameterized Edit Distance (MPED).

A. DESCRIPTION OF THE SPECIALIZATION ACTIVITY

1) DEFINITION OF MPED

In order to define MPED, we must preliminarily introduce some concepts. First of all let us briefly recall the definition of Edit Distance.

In its simplest form, the edit distance between two strings s_1 and s_2 is the minimum number of edit operations (insertions, deletions or substitutions) of single characters needed to transform s_1 into s_2, where each operation has a cost equal to 1 [57]. This version of the edit distance is also referred to as Levenshtein distance. A basic algorithm for edit distance computation exploits a dynamic programming approach; in it, the choice of the edit operation is carried out by a recurrence formula, where the discriminating factor is based on the question "Is $s_1[i] = s_2[j]$?". Intuitively, in our framework, this question is substituted by "According to M, do symbols $s_1[i]$ and $s_2[j]$ match?".

In more detail, let s_1 and s_2 be two strings defined over the alphabets Π_1 and Π_2. Let $-$ be a symbol not included in $\Pi_1 \cup \Pi_2$. Then, a string $\bar{s}_i$ over $\Pi_i \cup \{-\}$, $i \in 1, 2$, is a *Indel-Edit* of s_i if s_i can be obtained from $\bar{s}_i$ by deleting all the occurrences of $-$. The set of all the possible Indel-Edits of s_i is denoted by $\mathcal{IE}(s_i)$.

An *alignment* of the strings s_1 and s_2 is a pair $\langle \bar{s}_1, \bar{s}_2 \rangle$, where $\bar{s}_1 \in \mathcal{IE}(s_1)$, $\bar{s}_2 \in \mathcal{IE}(s_2)$ and $len(\bar{s}_1) = len(\bar{s}_2)$. Here, $-$ is meant to denote an insertion/deletion operation performed on s_1 or s_2.

Let $\langle \bar{s}_1, \bar{s}_2 \rangle$ be an alignment for s_1 and s_2, let $M_{\langle \pi_1, \pi_2, \chi \rangle}$ be a $\langle \pi_1, \pi_2, \chi \rangle$-constrained matching schema over π-partitions $\Phi_1^{\pi_1}$ and $\Phi_2^{\pi_2}$ and the set of constraints χ, and let j be a position with $1 \leq j \leq len(\bar{s}_1) = len(\bar{s}_2)$. We say that $\langle \bar{s}_1, \bar{s}_2 \rangle$ has a match at j if:

- $s_1[j] \in \phi_v$, $s_2[j] \in \phi_w$, $\phi_v \in \Phi_1^{\pi_1}$, $\phi_w \in \Phi_2^{\pi_2}$, and
- $M_{\langle \pi_1, \pi_2, \chi \rangle}(\phi_v, \phi_w) = true$.

The *distance* between $\bar{s}_1$ and $\bar{s}_2$ under $M_{\langle \pi_1, \pi_2, \chi \rangle}$ denotes the number of positions in which the pair $\langle \bar{s}_1, \bar{s}_2 \rangle$ does not have a match.

We are now able to introduce the notion of Multi-Parameterized Edit Distance (hereafter, MPED) between two strings s_1 and s_2.

Specifically, let π_1 and π_2 be two integers such that $0 < \pi_1 \leq |\Pi_1|$ and $0 < \pi_2 \leq |\Pi_2|$; the ***Multi-Parameterized Edit Distance*** between s_1 and s_2 ($\mathcal{L}_{\langle \pi_1, \pi_2, \chi \rangle}(s_1, s_2)$, for short) is the minimum distance that can be obtained with any $\langle \pi_1, \pi_2, \chi \rangle$-constrained matching schema and any alignment

$\langle \bar{s}_1, \bar{s}_2 \rangle$. Formally:

$$\mathfrak{A}_{\mathcal{L}} = \langle \Pi_1, \Pi_2, \langle \pi_1, \pi_2, \chi \rangle, \mathcal{M}, \mathcal{L}^M(\cdot,\cdot) \rangle \tag{8}$$

and

$$\begin{aligned} F_{\mathcal{L}}(s_1, s_2) &= \mathcal{L}_{\langle \pi_1, \pi_2, \chi \rangle}(s_1, s_2) \\ &= \min_{M_{\langle \pi_1, \pi_2, \chi \rangle} \in \mathcal{M}} \{\mathcal{L}^M(s_1, s_2)\} \end{aligned} \tag{9}$$

where $\mathcal{L}(\cdot,\cdot)$ is the classical edit distance.

Analogously to what we have seen with MPJD, also MPED has several applications. Among them we cite the anomaly detection in heterogeneous Wireless Sensor Networks (i.e., networks where the involved sensors might be heterogeneous and correlation between mate sensors might be unexpected) and the extraction of White Matter fiber-bundles from the high number of streamlines generated by a tractography algorithm. The interested reader is referred to [4] for the former application and to [5] for the latter one. In the former application, MPED allowed us to deal with heterogeneous sensor networks, where classical anomaly detection approaches were not perfectly suited, due to the heterogeneity of the data and the kind of anomalies addressed. In particular, the usage of MPED made possible to identify long-term anomalies, difficult to be addressed by classical machine learning tools. Furthermore, the possibility to use MPED to compare heterogeneous data streams allowed us to reduce the computational overhead. In the latter one, MPED allowed us to extract and characterize White Matter fiber-bundles. In particular, this metric, coupled with a novel string-based representation of White Matter fibers, effectively reduced the complexity of the problem at hand. Indeed, thanks to it, the extraction and characterization of White Matter fiber-bundles reduces to a string extraction and comparison problem.

2) EXAMPLE

Let $s_1 = \texttt{QQQx88yy8QQ}$ and $s_2 = \texttt{KKHbEbbHEE}$ be two strings, from which we draw the alphabets $\Pi_1 = \{\texttt{8}, \texttt{Q}, \texttt{x}, \texttt{y}\}$ and $\Pi_2 = \{\texttt{E}, \texttt{H}, \texttt{K}, \texttt{b}\}$. For $\pi_1 = \pi_2 = 1$, the best alignment $\langle \bar{s}_1, \bar{s}_2 \rangle$ that can be computed is obtained by the matching schema {{8}-{E}, {Q}-{K}, {x}-{b}, {y}-{H}}. We visualize the alignment by drawing s_1 and s_2 one below the other, and we add a star ($\star$) in a third row for each position in which they do not have a match:

```
s1 : QQQx88yy8QQ → QQQx88yy8QQ
s2 : KKHbEbbHEE  → KKHbEbbHE-E
                   *   **   **
```

By counting the star symbols, we obtain $\mathcal{L}_{\langle 1,1 \rangle}(s_1, s_2) = 5$. It is worth observing how this approach works properly even in the cases $\Pi_1 \cap \Pi_2 = \emptyset$ and $len(s_1) \neq len(s_2)$.

Furthermore, if we set $\pi_1 = \pi_2 = 2$, the matching schema {{8, y}-{H, b}, {Q, x}-{E, K}} allows us to obtain another alignment, that is

```
s1 : QQQx88yy8QQ → QQQx88yy8QQ
s2 : KKHbEbbHEE  → -KKHbEbbHEE
                   *  * *
```

which gives $\mathcal{L}_{\langle 2,2 \rangle}(s_1, s_2) = 3$, that corresponds to a lower distance.

As previously pointed out, sometimes there may be some constraints over symbols, specifying that some matches are not possible. In this example, suppose we have the constraint $\chi = \{\langle \texttt{Q}, \texttt{K} \rangle\}$. For $\pi_1 = \pi_2 = 1$, the best alignment $\langle \bar{s}_1, \bar{s}_2 \rangle$ is:

```
s1 : QQQx88yy8QQ → QQQx8-8yy8QQ
s2 : KKHbEbbHEE  → --KKHbEbbHEE
                   ***   **
```

that results in $\mathcal{L}_{\langle 1,1,\chi \rangle}(s_1, s_2) = 5$ and sets Q and K to not match. Indeed, the optimal matching schema is now given by {{8}-{H}, {Q}-{E}, {x}-{K}, {y}-{b}}.

VI. CONCLUSION

In this paper, we have proposed an approach to measure the (dis)similarity degree of strings belonging to different alphabets. In order to show that our approach is general and can be adapted to different metrics, we have illustrated its specialization to two metrics, namely the Jaccard and the Edit distances. We have shown how, thanks to the generalized Jaccard distance, it is possible to address the problem of transforming comparable corpora to parallel ones through an alignment process. Then, to illustrate how it can lead to address completely different issues, we have shown how, thanks to the generalized Edit distance, it is possible to handle a complex heterogeneous sensor network scenario and, also, to extract and characterize white matter fiber bundles of brain.

As far as future work is concerned, the generalization of other metrics can pave the way to a wide variety of new application scenarios, such as air flow or weather perturbation clustering, discovery of frequent or specific patterns on multi-dimensional data, and so on. Other interesting research lines regard the multiple alignment problem, extended to parameterized strings, or the comparison of multivariate time series, where several heterogeneous variables are monitored simultaneously.

REFERENCES

[1] D. E. Knuth, J. H. Morris, Jr., and V. R. Pratt, "Fast pattern matching in strings," *SIAM J. Comput.*, vol. 6, no. 2, pp. 323–350, Jul. 1977.

[2] B. Baker, "A theory of parameterized pattern matching: Algorithms and applications (extended abstract)," in *Proc. Annu. ACM Symp. Theory Comput.*, 1993, pp. 71–80.

[3] G. Sandve and F. Drabløs, "A survey of motif discovery methods in an integrated framework," *Biol. Direct*, vol. 1, no. 11, pp. 1–16, Apr. 2006.

[4] F. Cauteruccio, G. Fortino, A. Guerrieri, A. Liotta, D. C. Mocanu, C. Perra, G. Terracina, and M. T. Vega, "Short-long term anomaly detection in wireless sensor networks based on machine learning and multi-parameterized edit distance," *Inf. Fusion*, vol. 52, pp. 13–30, Dec. 2019.

[5] F. Cauteruccio, C. Stamile, G. Terracina, D. Ursino, and D. Sappey-Marinier, "An automated string-based approach to extracting and characterizing white matter fiber-bundles," *Comput. Biol. Med.*, vol. 77, pp. 64–75, Oct. 2016.

[6] A. K. Elmagarmid, P. G. Ipeirotis, and V. S. Verykios, "Duplicate record detection: A survey," *IEEE Trans. Knowl. Data Eng.*, vol. 19, no. 1, pp. 1–16, Jan. 2007.
[7] J. Mendivelso, S. V. Thankachan, and Y. Pinzón, "A brief history of parameterized matching problems," *Discrete Appl. Math.*, vol. 274, pp. 103–115, Mar. 2020.
[8] R. Singh, D. Rai, and R. Prasad, "A review on parameterized string matching algorithms," *J. Inf. Optim. Sci.*, vol. 39, no. 1, pp. 275–283, Nov. 2017.
[9] H. Rangwala, "A survey of remote homology detection and fold recognition methods," in *Introduction to Protein Structure Prediction: Methods and Algorithms*, 2010, pp. 165–194.
[10] J. M. Vidal, M. A. S. Monge, and L. J. G. Villalba, "A novel pattern recognition system for detecting Android malware by analyzing suspicious boot sequences," *Knowl.-Based Syst.*, vol. 150, pp. 198–217, Jun. 2018.
[11] J. Serrà and J. L. Arcos, "An empirical evaluation of similarity measures for time series classification," *Knowl.-Based Syst.*, vol. 67, pp. 305–314, Sep. 2014.
[12] J. Oncina and M. Sebban, "Learning stochastic edit distance: Application in handwritten character recognition," *Pattern Recognit.*, vol. 39, no. 9, pp. 1575–1587, Sep. 2006.
[13] D. Folgado, M. Barandas, R. Matias, R. Martins, M. Carvalho, and H. Gamboa, "Time alignment measurement for time series," *Pattern Recognit.*, vol. 81, pp. 268–279, Sep. 2018.
[14] A. Fischer, C. Y. Suen, V. Frinken, K. Riesen, and H. Bunke, "Approximation of graph edit distance based on Hausdorff matching," *Pattern Recognit.*, vol. 48, no. 2, pp. 331–343, Feb. 2015.
[15] J. Lerouge, Z. Abu-Aisheh, R. Raveaux, P. Héroux, and S. Adam, "New binary linear programming formulation to compute the graph edit distance," *Pattern Recognit.*, vol. 72, pp. 254–265, Dec. 2017.
[16] M. Neuhaus and H. Bunke, "Edit distance-based kernel functions for structural pattern classification," *Pattern Recognit.*, vol. 39, no. 10, pp. 1852–1863, Oct. 2006.
[17] W. H. Gomaa and A. A. Fahmy, "A survey of text similarity approaches," *Int. J. Comput. Appl.*, vol. 68, no. 13, pp. 13–18, Apr. 2013.
[18] M. Yu, G. Li, D. Deng, and J. Feng, "String similarity search and join: A survey," *Frontiers Comput. Sci.*, vol. 10, no. 3, pp. 399–417, Jun. 2016.
[19] J. Mendivelso and Y. Pinzón, "Parameterized matching: Solutions and extensions," in *Proc. Prague Stringology Conf.*, 2015, pp. 118–131.
[20] P. Gawrychowski and P. Uznanski, "Order-preserving pattern matching with k mismatches," in *Proc. 25th Annu. Symp. Combinat. Pattern Matching (CPM)*, Moscow, Russia, vol. 8486. Berlin, Germany: Springer, 2014, pp. 130–139.
[21] D. Diptarama, T. Katsura, Y. Otomo, K. Narisawa, and A. Shinohara, *Position Heaps for Parameterized Strings*, Wadern, Germany: Dagstuhl, 2017, pp. 8:1–8:13.
[22] R. Beal and D. Adjeroh, "Compressed parameterized pattern matching," *Theor. Comput. Sci.*, vol. 609, pp. 129–142, Jan. 2016.
[23] R. Khetan, S. Agarwal, and R. Prasad, "An efficient approach towards compressed parameterized word matching using wavelet tree," *J. Inf. Optim. Sci.*, vol. 37, no. 2, pp. 285–301, Apr. 2016.
[24] T. Kamiya, S. Kusumoto, and K. Inoue, "CCFinder: A multilinguistic token-based code clone detection system for large scale source code," *IEEE Trans. Softw. Eng.*, vol. 28, no. 7, pp. 654–670, Jul. 2002.
[25] B. S. Baker, "Parameterized diff," in *Proc. Annu. ACM-SIAM Symp. Discrete Algorithms*, Baltimore, MD, USA, 1999, pp. 854–855.
[26] R. Prasad and S. Agarwal, "Parameterized string matching: An application to software maintenance," *SIGSOFT Softw. Eng. Notes*, vol. 35, no. 3, pp. 1–5, 2010.
[27] J. Mendivelso, C. Pino, L. Ni no, and Y. Pinzón, "Approximate abelian periods to find motifs in biological sequences," in *Proc.11th Int. Meeting Comput. Intell. Methods Bioinf. Biostatistics (CIBB)*, vol. 8623, 2014, pp. 121–130.
[28] R. Beal and D. Adjeroh, "Efficient pattern matching for RNA secondary structures," *Theor. Comput. Sci.*, vol. 592, pp. 59–71, Aug. 2015.
[29] R. Singh, D. Rai, R. Prasad, and R. Singh, "Similarity detection in biological sequences using parameterized matching and Q-gram," in *Proc. Recent Adv. Eng., Technol. Comput. Sci. (RAETCS)*, Allahabad, India, Feb. 2018, pp. 1–6.
[30] E. Cambouropoulos, M. Crochemore, C. Iliopoulos, L. Mouchard, and Y. Pinzon, "Algorithms for computing approximate repetitions in musical sequences," *Int. J. Comput. Math.*, vol. 79, no. 11, pp. 1135–1148, 2002.
[31] J. Mendivelso, S. Kim, S. Elnikety, Y. He, S. Hwang, and Y. Pinzón, "Solving graph isomorphism using parameterized matching," in *Proc. 20th Int. Symp. String Process. Inf. Retr. (SPIRE)*, Jerusalem, Israel, vol. 8214. Berlin, Germany: Springer, 2013, pp. 230–242.
[32] C. Hazay, M. Lewenstein, and D. Sokol, "Approximate parameterized matching," *ACM Trans. Algorithms*, vol. 3, no. 3, pp. 29–es, Aug. 2007.
[33] S. Das and K. Kapoor, "Weighted approximate parameterized string matching," *AKCE Int. J. Graphs Combinatorics*, vol. 14, no. 1, pp. 1–12, Apr. 2017.
[34] A. Apostolico, P. L. Erdös, and M. Lewenstein, "Parameterized matching with mismatches," *J. Discrete Algorithms*, vol. 5, no. 1, pp. 135–140, Mar. 2007.
[35] A. Amir and I. Nor, "Generalized function matching," *J. Discrete Algorithms*, vol. 5, no. 3, pp. 514–523, Sep. 2007.
[36] G. Greco and G. Terracina, "Frequency-based similarity for parameterized sequences: Formal framework, algorithms, and applications," *Inf. Sci.*, vol. 237, pp. 176–195, Jul. 2013.
[37] O. Keller, T. Kopelowitz, and M. Lewenstein, "On the longest common parameterized subsequence," *Theor. Comput. Sci.*, vol. 410, no. 51, pp. 5347–5353, Nov. 2009.
[38] I. Costas, M. Kubica, M. Rahman, and T. Waleń, "Algorithms for computing the longest parameterized common subsequence," in *Proc. 18th Annu. Symp. Combinat. Pattern Matching (CPM)*, London, ON, Canada, vol. 4580. Berlin, Germany: Springer, 2007, pp. 265–273.
[39] I. Lee, J. Mendivelso, and Y. Pinzón, "Delta-gamma-parameterized matching," in *Proc. 15th Int. Symp. String Process. Inf. Retr. (SPIRE)*, Melbourne, VIC, Australia, vol. 5280. Berlin, Germany: Springer, 2008, pp. 236–248.
[40] R. Rapp, P. Zweigenbaum, and S. Sharoff, "Overview of the third bucc shared task: Spotting parallel sentences in comparable corpora," in *Proc. 11th Workshop Building Using Comparable Corpora*, Paris, France, May 2018, pp. 39–42.
[41] O. Bojar *et al.*, "Findings of the 2016 conference on machine translation," in *Proc. 1st Conf. Mach. Transl.*, vol. 2. Berlin, Germany: Shared Task Papers, 2016, pp. 131–198. [Online]. Available: https://www.aclweb.org/anthology/W16-2301
[42] I. Vulić and M. Moens, "Monolingual and cross-lingual information retrieval models based on (bilingual) word embeddings," in *Proc. 38th Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, New York, NY, USA, 2015, p. 363–372.
[43] V. Z. Agić and N. Schluter, "Baselines and test data for cross-lingual inference," in *Proc. 11th Int. Conf. Lang. Resour. Eval. (LREC)*, Miyazaki, Japan, May 2018, pp. 1–5. [Online]. Available: https://www.aclweb.org/anthology/L18-1614
[44] T. Etchegoyhen and A. Azpeitia, "Set-theoretic alignment for comparable corpora," in *Proc. 54th Annu. Meeting Assoc. Comput. Linguistics (Long Papers)*, Berlin, Germany, vol. 1, Aug. 2016, pp. 2009–2018. [Online]. Available: https://www.aclweb.org/anthology/P16-1189
[45] T. Štajner and D. Mladenić, "Cross-lingual document similarity estimation and dictionary generation with comparable corpora," *Knowl. Inf. Syst.*, vol. 58, no. 3, pp. 729–743, Mar. 2018.
[46] T. E. Andoni Azpeitia and E. M. Garcia, "Extracting parallel sentences from comparable corpora with stacc variants," in *Proc. 11th Int. Conf. Lang. Res. Eval. (LREC)*, Paris, France, May 2018, pp. 1–5.
[47] A. Azpeitia, T. Etchegoyhen, and E. Martínez Garcia, "Weighted set-theoretic alignment of comparable sentences," in *Proc. 10th Workshop Building Using Comparable Corpora*, Vancouver, BC, Canada, Aug. 2017, pp. 41–45. [Online]. Available: https://www.aclweb.org/anthology/W17-2508
[48] H. Bouamor and H. Sajjad, "H2@BUCC18: Parallel sentence extraction from comparable corpora using multilingual sentence embeddings," in *Proc. 11th Int. Conf. Lang. Res. Eval. (LREC)*, 2018, pp. 43–46.
[49] G. Kontonatsios, I. Korkontzelos, J. Tsujii, and S. Ananiadou, "Combining string and context similarity for bilingual term alignment from comparable corpora," in *Proc. Conf. Empirical Methods Natural Lang. Process. (EMNLP)*, 2014, pp. 1701–1712. [Online]. Available: https://www.aclweb.org/anthology/D14-1177
[50] J. Smith, C. Quirk, and K. Toutanova, "Extracting parallel sentences from comparable corpora using document level alignment," in *Proc. Hum. Lang. Technol., Annu. Conf. North Amer. Chapter Assoc. Comput. Linguistics (NAACL HLT)*, 2010, pp. 403–411.
[51] S. Pal, P. Pakray, and S. K. Naskar, "Automatic building and using parallel resources for SMT from comparable corpora," in *Proc. 3rd Workshop Hybrid Approaches Mach. Transl. (HyTra)*, 2014, pp. 48–57. [Online]. Available: https://www.aclweb.org/anthology/W14-1009

[52] P. Koehn, "Europarl: A parallel corpus for statistical machine translation," in *Proc. 10th Mach. Transl. Summit*, Phuket, Thailand, 2005, pp. 79–86. [Online]. Available: http://mt-archive.info/MTS-2005-Koehn.pdf
[53] P. Zweigenbaum, S. Sharoff, and R. Rapp, "Overview of the second BUCC shared task: Spotting parallel sentences in comparable corpora," in *Proc. 10th Workshop Building Using Comparable Corpora*, Vancouver, BC, Canada, Aug. 2017, pp. 60–67. [Online]. Available: https://www.aclweb.org/anthology/W17-2512
[54] H. Schmid, "Probabilistic part-of-speech tagging using decision trees," in *Proc. Int. Conf. New Methods Lang. Process.*, Manchester, U.K., 1994, p. 154.
[55] H. Schmid, "Improvements in part-of-speech tagging with an application to German," in *Proc. ACL SIGDAT-Workshop*, Dublin, Ireland, 1995, pp. 13–25.
[56] A. Gorbenko and V. Popov, "The longest common parameterized subsequence problem," *Appl. Math. Sci.*, vol. 6, no. 58, pp. 2851–2855, 2012.
[57] V. Levenshtein, "Binary codes capable of correcting deletions, insertions and reversals," *Sov. Phys. Doklady*, vol. 10, p. 707, Feb. 1966.

FRANCESCO CAUTERUCCIO received the M.Sc. degree in computer science and the Ph.D. degree in mathematics and computer science from the University of Calabria, in 2014 and January 2018, respectively. He is actively working in research projects, such as the Smarter Solutions in the Big Data World, funded within the call HORIZON2020 PON I&C 2014-2020. He is currently a member of the Advanced Database Research Laboratory, hosted by the University of Calabria. He is also the author of 11 articles. His research interests include machine learning, time series analysis, and biomedical applications.

ALESSANDRO CUCCHIARELLI received the M.Sc. degree in electronic engineering from the University of Ancona (now Polytechnic University of Marche), in 1985. In 1991, he joined the Computer Science Institute, Polytechnic University of Marche, where he has been an Associate Professor with the Department of Information Engineering (DII), since 2005. His research interests are focused on the application of natural language processing techniques to automatic extraction of information from the Web, domain ontologies definition, and definition of tools and metrics for social network analysis and recommendation systems. He is the author or a coauthor of about 90 articles on refereed journals, conference proceedings, and book chapters.

CHRISTIAN MORBIDONI received the M.Sc. degree in electronic engineering and the Ph.D. degree in computer science from the Polytechnic University of Marche, Ancona, Italy, in 2003 and 2006, respectively. He currently carries out his research activity with the Polytechnic University of Marche. His research interests include semantic Web, knowledge representation, information extraction, recommendation systems, machine learning, and deep learning. He leaded research and development work packages are several EU and national funded projects in the area of cultural heritage and digital humanities. He is also the author of around 50 research articles.

GIORGIO TERRACINA received the M.Sc. degree in computer engineering from the University of Calabria, in April 1999, and the Ph.D. degree in electronic engineering from the University of Reggio Calabria, in 2002. From December 2002 to October 2010, he was a tenured Assistant Professor with the University of Calabria, where he is currently an Associate Professor, since November 2010. He has been responsible of several research units in national and international research projects and held several management positions, such as a member of the Board of Directors with the University of Calabria and IDUM, a spin-off from the University of Calabria. His research interests include source and data integration, social network analysis, knowledge representation and reasoning, and bioinformatics. In these research areas, he has published more than 140 articles.

DOMENICO URSINO received the M.Sc. degree in computer engineering and the Ph.D. degree in system engineering and computer science from the University of Calabria, in July 1995 and January 2000, respectively. From January 2005 to December 2017, he was an Associate Professor with the University Mediterranea of Reggio Calabria. Since January 2018, he has been a Full Professor with the Polytechnic University of Marche. His research interests include source and data integration, data lakes, social network analysis, social internetworking, ecosystems consisting of the Internet of Things, innovation management, knowledge extraction and representation, biomedical applications, and recommender systems. In these research fields, he has published more than 180 articles.