SEDiL: Software for Edit Distance Learning

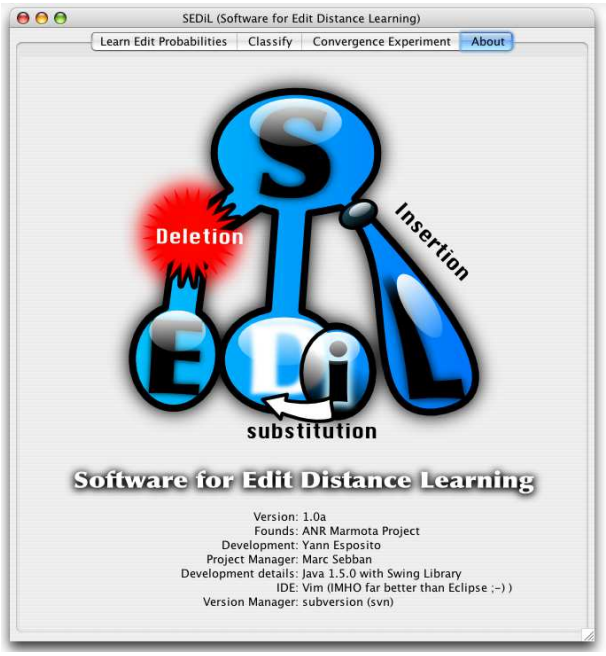
Laurent Boyer, Yann Esposito, Amaury Habrard, Jose Oncina, Marc Sebban

SEDiL: Software for Edit Distance Learning*

Laurent Boyer[1], Yann Esposito[1], Amaury Habrard[2],
Jose Oncina[3], and Marc Sebban[1]

[1] Laboratoire Hubert Curien, Université de Saint-Etienne, France
{laurent.boyer,yann.esposito,marc.sebban}@univ-st-etienne.fr
[2] Laboratoire d'Informatique Fondamentale, CNRS, Aix Marseille Université, France
amaury.habrard@lif.univ-mrs.fr
[3] Dep. de Languajes y Sistemas Informatico, Universidad de Alicante, Spain
oncina@dlsi.ua.es

Abstract. In this paper, we present SEDiL, a **S***oftware for* **E***dit* **Di***stance* **L***earning*. SEDiL is an innovative prototype implementation grouping together most of the state of the art methods [1–4] that aim to automatically learn the parameters of string and tree edit distances.



* This work was funded by the French *ANR Marmota project*, the *Pascal Network of Excellence* and the Spanish research programme *Consolider Ingenio-2010 (CSD2007-00018)*. This publication only reflects the authors' views.

1 Introduction

In many machine learning tasks, there is an increasing need for defining similarity measures between objects. In the case of structured data, such as strings or trees, one natural candidate is the well-known edit distance (ED) [5, 6]. It is based on edit operations (*insertion, deletion* and *substitution*) required for changing an input data (*e.g.* a misspelled word) into an output one (*e.g.* its correction). In general, an *a priori* fixed cost is associated to each edit operation. Without any specific knowledge, these costs are usually set to 1. However, in many real world applications, such a strategy clearly appears insufficient. To overcome this drawback and to capture background knowledge, supervised learning has been used during the last few years for learning the parameters of edit distances [1–4, 7–9], often by maximizing the likelihood of a learning set. The learned models usually take the form of state machines such as stochastic transducers or probabilistic automata.

Since 2004, three laboratories (the LaHC from France, the DLSI from Spain, both members of the RedEx PASCAL, and the LIF from France) have joined their efforts to propose new stochastic models of string and tree EDs in order to outperform not only the standard ED algorithms [5, 6] but also the first generative learning methods proposed in [1, 2]. This joint work has lead to publications in the previous conferences ECML'06 [7] and ECML'07 [4], and in Pattern Recognition [3, 8]. This research has also received funding from the RedEx PASCAL in the form of a pump-priming project in 2007. Since no software platform was available, a part of this financial help has been used to implement these new innovative prototypes in the platform SEDiL [10].

The rest of this paper is organized as follows: in Section 2, we present a brief survey of ED learning methods. Then, Section 3 is devoted to the presentation of our platform SEDiL and its innovative aspects.

2 Related Work

2.1 Standard ED

The standard ED between two strings (or trees) is the minimal cost to transform by edit operations an input data into an output one. To each operation (*insertion, deletion* and *substitution*) is assigned a so-called *edit cost*. Let us suppose that strings or trees are defined over a finite alphabet Σ and that the empty symbol is denoted by $\lambda \notin \Sigma$.

Definition 1. *An edit script $e = e_1 \cdots e_n$ is a sequence of edit operations $e_i = (a_i, b_i) \in (\Sigma \cup \{\lambda\}) \times (\Sigma \cup \{\lambda\})$ allowing the transformation of an input data X into an output one Y. The cost of an edit script $\pi(e)$ is the sum of the costs of the edit operations involved in the script:*

$$\pi(e) = \sum_{i=1}^{n} c(e_i).$$

Let $S(X,Y)$ be the set of all the scripts that enable the emission of Y given X, the edit distance between X and Y is defined by:

$$d(X,Y) = min_{e \in S(X,Y)} \pi(e).$$

In order to improve the efficiency of the ED to solve machine learning tasks, learning algorithms have been developed to capture background knowledge from examples in order to learn relevant edit costs.

2.2 ED Learning by Memoryless State Machines

When there is no reason that the cost of a given edit operation changes according to the context where the operation occurs, *memoryless* models (see [1,3] in the case of strings and [4] for trees) are very efficient from an accuracy and algorithmic point of view. In practice, many applications satisfy this hypothesis. For instance, the correction of typing errors made with a computer keyboard, the recognition of handwritten digits represented with Freeman codes [3] are some examples of problems that can be solved by these *memoryless* state machines. In such a context, these models learn - by likelihood maximization - one matrix of edit costs not only useful for computing edit similarities between strings or trees, but also to model knowledge of the domain. Actually, by mining the learned matrix it enables us to answer questions such that "*What are the letters of a keyboard at the origin of the main typing errors?*" or "*What are the possible graphical distortions of a given handwritten character?*". To give an illustration of the output of these models, let us suppose that strings are built from an alphabet $\Sigma = \{a, b\}$. Memoryless models return a probability δ for each possible edit operation in the form of a 3×3 matrix as shown in Fig.1.

δ	λ	a	b
λ	–	0.3	0.1
a	0.05	0.0	0.2
b	0.05	0.1	0.2

Fig. 1. Matrix δ of edit probabilities.

On this toy example, $\delta(a,b) = 0.2$ means that the letter a has a probability of 0.2 to be changed into a b. From these edit probabilities, it is then possible, using dynamic programming, to compute in a quadratic complexity the probability $p(X,Y)$ to change an input X into an output Y. Note here that, all the possible ways for transforming X into Y are taken into account, in opposition to the standard ED which considers, in general, only the minimal cost. Ristad and Yianilos [1] showed that one can obtain a so-called *stochastic ED* d_s by computing

$$d_s(X,Y) = -\log p(X,Y).$$

2.3 ED Learning by Non-Memoryless State Machines

In specific applications, the edit operation has an impact more or less important in the data transformation according to its location. For instance, in molecular biology, it is common knowledge that the probability to change a symbol into another one depends on its membership of a transcription factor binding site. To deal with such situations, *non-memoryless* approaches have been proposed [2, 9] in the form of probabilistic state machines that are able to take into account the string context[4]. So, the output of the model is not a 2-dimensional matrix anymore, but rather a 3-dimensional matrix where the third dimension represents the state where the operation occurs.

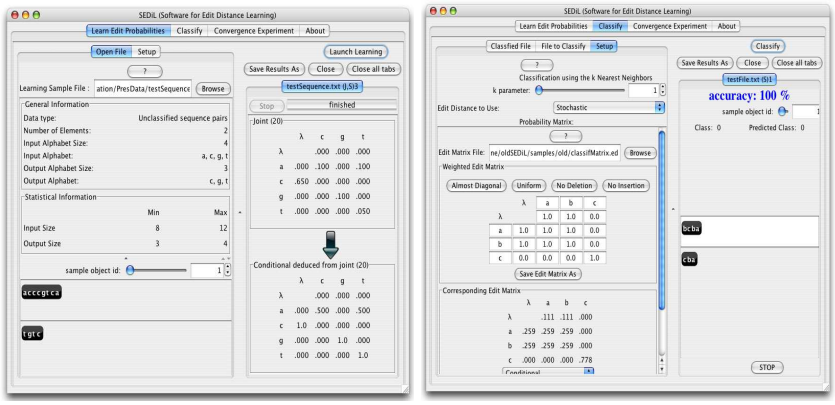


Fig. 2. Screen-shots of SEDiL. On the left, an example of learning from strings; On the right, an illustration of the classification step.

3 Presentation of the Platform SEDiL

The platform SEDiL groups together the state of the art ED learning algorithms. It is an open source software written in Java and can be used on a local machine using Java Web Start technology or directly from our web with an applet [10]. Some screen-shots of the application are presented on Figures 2 and 3. SEDiL enables us to perform two kinds of tasks: *learning* and *classification*.

3.1 Edit Parameters Learning and Classification

All the algorithms we implemented are based on an adaptation of the Expectation-Maximization algorithm to the context of EDs. From a learning sample of labeled pairs of strings or trees, the system automatically generates learning pairs either

[4] Note that so far, no approach has been proposed for trees.

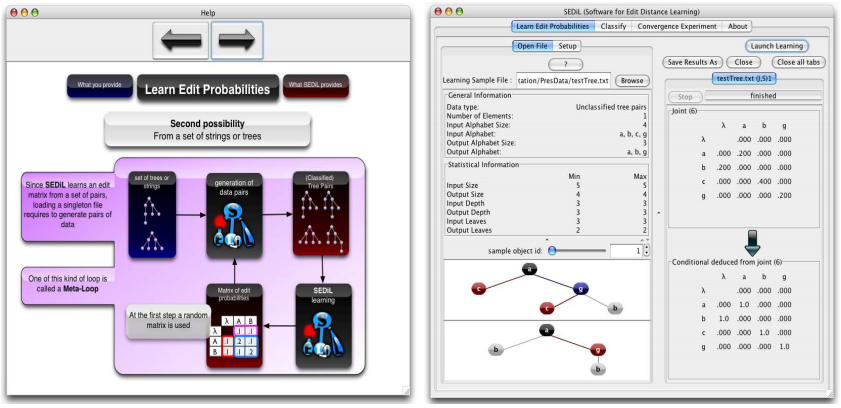


Fig. 3. Screen-shots of SEDiL. On the left, an help panel of SEDiL; On the right, an example of learning from tree-structured data.

randomly (by choosing an output data in the same class) or by associating to each example its nearest neighbor from instances of the same class. Note that the user can also directly provide labeled training pairs (*e.g.* misspelled/corrected words). The result of the learning step takes the form of a matrix of edit probabilities (as shown in Fig.1) that represents either a generative (joint distribution) or a discriminative model (conditional).

Since an ED is useful for computing neighborhoods, we implemented a classification interface allowing the use of a k-nearest neighbor algorithm with the learned edit similarities. Therefore, all the methods we have implemented can be compared (included the standard ED) in a classification task.

3.2 Contribution of SEDiL for the ML and DM communities

What makes our piece of software unique and special?

As far as we know, there does not exist any available software platform to automatically learn the parameters of similarity measures based on the notion of ED. By implementing SEDiL, we filled this gap.

What are the innovative aspects or in what way/area does it represent the state of the art?

The two main historical references in ED learning are [1, 2]. We implemented not only this state of the art but also all the recent new approaches we published during the past 4 years. In its current version, SEDiL provides five learning methods which are based on the following scientific papers:

- The learning of an edit pair-Hidden Markov Model (Durbin et al. 1998 [2]).
- The learning of a joint stochastic string ED (Ristad & Yianilos [1]).
- The learning of a conditional stochastic string ED (Oncina & Sebban [3]).

- The learning of a stochastic tree ED with deletion and insertion operations on subtrees (Bernard et al. 2006 [7, 8]).
- The learning of a stochastic tree ED with deletion and insertion operations on nodes (Boyer et al. 2007 [4]).

For whom is it most interesting/useful?

SEDiL has already been used in various situations not only by academic but also by industrial structures. In the context of the ANR BINGO Project (`http://users.info.unicaen.fr/~bruno/bingo`), SEDiL has been used to search regularities in sequences of promoters in molecular biology. It is used at the moment in music recognition by the laboratory DLSI of the University of Alicante. Moreover, it has been recently exploited in a handwritten recognition task by the company A2IA (`http://www.a2ia.com/`). More generally, SEDiL is devoted to researchers who aim to compute distances between structured data.

4 Conclusion

SEDiL is the first platform providing several ED learning algorithms. It is devoted to improvements. We plan to implement a new edit model in the form of constrained state machines able to take into account background knowledge and a conditional random field approach recently presented in [9].

References

1. Ristad, S., Yianilos, P.: Learning string-edit distance. IEEE Transactions on Pattern Analysis and Machine Intelligence **20**(5) (1998) 522–532
2. Durbin, R., Eddy, S., krogh, A., Mitchison, G.: Biological sequence analysis. Cambridge University Press (1998) Probalistic models of protein and nucleic acids.
3. Oncina, J., Sebban, M.: Learning stochastic edit distance: application in handwritten character recognition. Journal of Pattern Recognition **39**(9) (2006) 1575–1587
4. Boyer, L., Habrard, A., Sebban, M.: Learning metrics between tree structured data: Application to image recognition. In: Proceedings of the 18th European Conference on Machine Learning (ECML'07). Volume 4701 of LNCS., Springer (2007) 54–66
5. Wagner, R., Fischer, M.: The string-to-string correction problem. Journal of the ACM (JACM) **21** (1974) 168–173
6. Bille, P.: A survey on tree edit distance and related problem. Theoretical Computer Science **337**(1-3) (2005) 217–239
7. Bernard, M., Habrard, A., Sebban, M.: Learning stochastic tree edit distance. In: Proceedings of the 17th European Conference on Machine Learning (ECML'06). Volume 4212 of LNCS., Springer (2006) 42–52
8. Bernard, M., Boyer, L., Habrard, A., Sebban, M.: Learning probabilistic models of tree edit distance. Pattern Recognition **41**(8) (2008) 2611–2629
9. McCallum, A., Bellare, K., Pereira, F.: A conditional random field for discriminatively-trained finite-state string edit distance. In: Proceedings of the 21st Conference on Uncertainty in AI. (2005) 388–395
10. `http://labh-curien.univ-st-etienne.fr/informatique/SEDiL`