Automatic Cost Estimation for Tree Edit Distance Using Particle Swarm Optimization

Yashar Mehdad
University of Trento and FBK - Irst
Trento, Italy
mehdad@fbk.eu

Abstract

Recently, there is a growing interest in working with tree-structured data in different applications and domains such as computational biology and natural language processing. Moreover, many applications in computational linguistics require the computation of similarities over pair of syntactic or semantic trees. In this context, Tree Edit Distance (TED) has been widely used for many years. However, one of the main constraints of this method is to tune the cost of edit operations, which makes it difficult or sometimes very challenging in dealing with complex problems. In this paper, we propose an original method to estimate and optimize the operation costs in TED, applying the Particle Swarm Optimization algorithm. Our experiments on Recognizing Textual Entailment show the success of this method in automatic estimation, rather than manual assignment of edit costs.

1 Introduction

Among many tree-based algorithms, Tree Edit Distance (TED) has offered many solutions for various NLP applications such as information retrieval, information extraction, similarity estimation and textual entailment. Tree edit distance is defined as the minimum costly set of basic operations transforming one tree to another. In common, TED approaches use an initial fixed cost for each operation.

Generally, the initial assigned cost to each edit operation depends on the nature of nodes, applications and dataset. For example the probability of deleting a function word from a string is not the same as deleting a symbol in RNA structure. According to this fact, tree comparison may be affected by application and dataset. A solution to this problem is assigning the cost to each edit operation empirically or based on the expert knowledge and recommendation. These methods emerge a critical problem when the domain, field or application is new and the level of expertise and empirical knowledge is very limited.

Other approaches towards this problem tried to learn a generative or discriminative probabilistic model (Bernard et al., 2008) from the data. One of the drawbacks of those approaches is that the cost values of edit operations are hidden behind the probabilistic model. Additionally, the cost can not be weighted or varied according to the tree context and node location.

In order to overcome these drawbacks, we are proposing a stochastic method based on Particle Swarm Optimization (PSO) to estimate the cost of each edit operation based on the user defined application and dataset. A further advantage of the method, besides automatic learning of the operation costs, is to investigate the cost values in order to better understand how TED approaches the application and data in different domains.

As for the experiments, we learn a model for recognizing textual entailment, based on TED, where the input is a pair of strings represented as syntactic dependency trees. Our results illustrate that optimizing the cost of each operation can dramatically affect the accuracy and achieve a better model for recognizing textual entailment.

2 Tree Edit Distance

Tree edit distance measure is a similarity metric for rooted ordered trees. Assuming that we have two rooted and ordered trees, it means that one node in each tree is assigned as a root and the children of each node are ordered. The edit operations on the nodes a and b between trees are defined as: *Insertion* ($\lambda \rightarrow a$), *Deletion* ($a \rightarrow \lambda$) and *Substitution* ($a \rightarrow b$). Each edit operation has an associated cost (denoted as $\gamma(a \rightarrow b)$). An edit script on two trees is a sequence of edit operations changing a tree to another. Consequently, the cost of an edit script is the sum of the costs of its edit operations. Based on the main definition of this approach, TED is the cost of minimum cost edit script between two trees (Zhang and Shasha, 1989).

In the classic TED, a cost value is assigned to each operation initially, and the distance is computed based on the initial cost values. Considering that the distance can vary in different domains and datasets, converging to an optimal set of values for operations is almost empirically impossible. In the following sections, we propose a method for estimating the optimum set of values for operation costs in TED algorithm. Our method is built on adapting the PSO optimization approach as a search process to automate the procedure of cost estimation.

3 Particle Swarm Optimization

PSO is a stochastic optimization technique which was introduced recently based on the social behaviour of bird flocking and fish schooling (Eberhart et al., 2001). PSO is one of the population-based search methods which takes advantage of the concept of social sharing of information. In this algorithm each *particle* can learn from the experience of other particles in the same population (called *swarm*). In other words, each particle in the iterative search process would adjust its flying velocity as well as position not only based on its own acquaintance but also other particles' flying experience in the swarm. This algorithm has found efficient in solving a number of engineering problems. PSO is mainly built on the following equations.

$$X_i = X_i + V_i \tag{1}$$

$$\begin{aligned} V_i = \omega V_i &+ c_1 r_1 (X_{bi} - X_i) \\ &+ c_2 r_2 (X_{gi} - X_i) \end{aligned} \tag{2}$$

To be concise, for each particle at each iteration, the position X_i (Equation 1) and velocity V_i (Equation 2) is updated. X_{bi} is the best position of the particle during its past routes and X_{gi} is the best global position over all routes travelled by the particles of the swarm. r_1 and r_2 are random variables drawn from a uniform distribution in the range [0,1], while c_1 and c_2 are two acceleration constants regulating the relative velocities with respect to the best local and global positions. The weight ω is used as a tradeoff between the global and local best positions. It is usually selected slightly less than 1 for better global exploration (Melgani and Bazi, 2008). Position optimally is computed based on the fitness function defined in association with the related problem. Both position and velocity are updated during the iterations until convergence is reached or iterations attain the maximum number defined by the user.

4 Automatic Cost Optimization for TED

In this section we proposed a system for estimating and optimizing the cost of each edit operation for TED. As mentioned earlier, the aim of this system is to find the optimal set of operation costs to: 1) improve the performance of TED in different applications, and 2) provide some information on how different operations in TED approach an application or dataset. In order to obtain this, the system is developed using an optimization framework based on PSO.

4.1 PSO Setup

One of the most important steps in applying PSO is to define a fitness function, which could lead the swarm to the optimized particles based on the application and data. The choice of this function is very crucial since, based on this, PSO evaluates the quality of each candidate particle for driving the solution space to optimization. Moreover, this function should be, possibly, application and data independent, as well as flexible enough to be adapted to the TED based problems. With the intention of accomplishing these goals, we define two main fitness functions as follows:

1) Bhattacharyya Distance: This statistical measure determines the similarity of two discrete probability distributions (Bhattacharyya, 1943). In classification, this method is used to measure the distance between two different classes. Put it differently, maximizing the Bhattacharyya distance would increase the separability of two classes.

2) Accuracy: By maximizing the accuracy obtained from 10 fold cross-validation on the development set, as the fitness function, we estimate the optimized cost of the edit operations.

4.2 Integrating TED with PSO

The procedure to estimate and optimize the cost of edit operations in TED applying the PSO algorithm, is as follows.

a) *Initialization*

1) Generate a random swarm of size n (cost of edit operations).
2) For each position of the particle from the swarm, obtain the fitness function value.
3) Set the best position of each particle with its initial position (X_{bi}).

b) *Search*

4) Detect the best global position (X_{gi}) in the swarm based on maximum value of the fitness function over all explored routes.
5) Update the velocity of each particle (V_i).
6) Update the position of each particle (X_i).
7) For each candidate particle calculate the fitness function.
8) Update the best position of each particle if the current position has a larger value.

c) *Convergence*

9) Run till the maximum number of iteration (in our case set to 10) is reached or start the search process.

5 Experimental Design

Our experiments were conducted on the basis of Recognizing Textual Entailment (RTE) datasets[1]. Textual Entailment can be explained as an association between a coherent text(T) and a language expression, called hypothesis(H). The entailment function for the pair T-H returns the true value when the meaning of H can be inferred from the meaning of T and false otherwise. In another word, Textual Entailment can be defined as human reading comprehension task. One of the approaches to textual entailment problem is based on the distance between T and H.

In this approach, the entailment score for a pair is calculated on the minimal set of edit operations that transform T into H. An entailment relation is assigned to a T-H pair in the case that overall cost of the transformations is below a certain threshold. The threshold, which corresponds to tree edit distace, is empirically estimated over the dataset. This method was implemented by (Kouylekov and Magnini, 2005), based on TED algorithm (Zhang and Shasha, 1989). Each RTE dataset includes its own development and test set, however, RTE-4 was released only as a test set and the data from RTE-1 to RTE-3 were exploited as development set for evaluating RTE-4 data.

In order to deal with TED approach to textual entailment, we used EDITS[2] package (Edit Distance Textual Entailment Suite) (Magnini et al., 2009). In addition, We partially exploit JSwarm-PSO[3] package with some adaptations as an implementation of PSO algorithm. Each pair in the datasets converted to two syntactic dependency trees using Stanford statistical parser[4], developed in the Stanford university NLP group by (Klein and Manning, 2003).

We conducted six different experiments in two sets on each RTE dataset. The costs were estimated on the training set, then we evaluate the estimated costs on the test set. In the first set of experiments, we set a simple cost scheme based on three operations. Implementing this cost scheme, we expect to optimize the cost of each edit operation without considering that the operation costs may vary based on different characteristics of a node, such as size, location or content. The results were obtained using: 1) The random cost assignment, 2) Assigning the cost based on the expertise knowledge and intuition (So called Intuitive), and 3) Automatic estimated and optimized cost for each operation. In the second case, we applied the same cost values which was used in EDITS by its developers (Magnini et al., 2009).

In the second set of experiments, we tried to take advantage of an advanced cost scheme with more fine-grained operations to assign a weight to the edit operations based on the characteristics of the nodes (Magnini et al., 2009). For example if a node is in the list of stop-words, the deletion cost should be different from the cost of deleting a content word. By this intuition, we tried to optimize 9 specialized costs for edit operations (A swarm of size 9). At each experiment, both fitness functions were applied and the best results were chosen for presentation.

[1]http://www.pascal-network.org/Challenges/RTE1-4

[2]http://edits.fbk.eu/

[3]http://jswarm-pso.sourceforge.net/

[4]http://nlp.stanford.edu/software/lex-parser.shtml

Model		Data set RTE4	RTE3	RTE2	RTE1
Simple	Random	49.6	53.62	50.37	50.5
	Intuitive	51.3	59.6	56.5	49.8
	Optimized	56.5	61.62	58	58.12
Adv.	Random	53.60	52.0	54.62	53.5
	Intuitive	57.6	59.37	57.75	55.5
	Optimized	59.5	62.4	59.87	58.62
Baseline		57.19			
RTE-4 Challenge		57.0			

Table 1: Comparison of accuracy on all RTE datasets based on optimized and unoptimized cost schemes.

6 Results

Our results are summarized in Table 1. We show the accuracy gained by a distance-based baseline for textual entailment (Mehdad and Magnini, 2009) in compare with the results achieved by the random, intuitive and optimized cost schemes using EDITS system. For the better comparison, we also present the results of the EDITS system (Cabrio et al., 2008) in RTE-4 challenge using combination of different distances as features for classification (Cabrio et al., 2008).

Table 1 shows that, in all datasets, accuracy improved up to 9% by optimizing the cost of each edit operation. Results prove that, the optimized cost scheme enhances the quality of the system performance even more than the cost scheme used by the experts (Intuitive cost scheme). Furthermore, using the fine-grained and weighted cost scheme for edit operations we could achieve the highest results in accuracy. Moreover, by exploring the estimated optimal cost of each operation, we could find even some linguistics phenomena which exists in the dataset. For instance, in most of the cases, the cost of deletion was estimated zero, which shows that deleting the words from the text does not effect the distance in the entailment pairs. In addition, the optimized model can reflect more consistency and stability (from 58 to 62 in accuracy) than other models, while in unoptimized models the result varies more, on different datasets (from 50 in RTE-1 to 59 in RTE-3).

7 Conclusion

In this paper, we proposed a novel approach for estimating the cost of edit operations in TED. This model has the advantage of being efficient and more transparent than probabilistic approaches as well as having less complexity. The easy implementation of this approach, besides its flexibility, makes it suitable to be applied in real world applications. The experimental results on textual entailment, as one of the challenging problems in NLP, confirm our claim.

Acknowledgments

Besides my special thanks to F. Melgani, B. Magnini and M. Kouylekov for their academic and technical support, I acknowledge the reviewers for their comments. The EDITS system has been supported by the EU-funded project QALL-ME (FP6 IST-033860).

References

M. Bernard, L. Boyer, A. Habrard, and M. Sebban. 2008. Learning probabilistic models of tree edit distance. *Pattern Recogn.*, 41(8):2611–2629.

A. Bhattacharyya. 1943. On a measure of divergence between two statistical populations defined by probability distributions. *Bull. Calcutta Math. Soc.*, 35:99109.

E. Cabrio, M. Kouylekovand, and B. Magnini. 2008. Combining specialized entailment engines for rte-4. In *Proceedings of TAC08, 4th PASCAL Challenges Workshop on Recognising Textual Entailment*.

R. C. Eberhart, Y. Shi, and J. Kennedy. 2001. *Swarm Intelligence*. The Morgan Kaufmann Series in Artificial Intelligence.

D. Klein and C. D. Manning. 2003. Fast exact inference with a factored model for natural language parsing. In *Advances in Neural Information Processing Systems 15*, Cambridge, MA. MIT Press.

M. Kouylekov and B. Magnini. 2005. Recognizing textual entailment with tree edit distance algorithms. In *PASCAL Challenges on RTE*, pages 17–20.

B. Magnini, M. Kouylekov, and E. Cabrio. 2009. Edits - Edit Distance Textual Entailment Suite User Manual. Available at *http://edits.fbk.eu/*.

Y. Mehdad and B. Magnini. 2009. A word overlap baseline for the recognizing textual entailment task. Available at *http://edits.fbk.eu/*.

F. Melgani and Y. Bazi. 2008. Classification of electrocardiogram signals with support vector machines and particle swarm optimization. *IEEE Transactions on Information Technology in Biomedicine*, 12(5):667–677.

K. Zhang and D. Shasha. 1989. Simple fast algorithms for the editing distance between trees and related problems. *SIAM J. Comput.*, 18(6):1245–1262.