# EDITS 3.0 at RTE-7

**Milen Kouylekov**
Celi SRL
kouylekov@celi.it

**Alessio Bosca**
Celi SRL
bosca@celi.it

**Luca Dini**
Celi SRL
dini@celi.it

## Abstract

This paper overviews CELI's[1] participation in the *Main, Novelty* and *KBP* task organized within the RTE-7 Evaluation Campaign. Our submissions have been produced running the EDITS (Edit Distance Textual Entailment Suite) open source RTE package, which allows to experiment with different combinations of algorithms, entailment rules, and optimization strategies. The evaluation on test data confirmed their effectiveness, with good results in all of the tasks. Our best run in the Main task achieved a Micro-Averaged F-measure of 44.10% (with the best and the median system respectively achieving 48.0% and 41.90%); our best run in the Novelty task achieved a Highest Primary F-measure of 87.21% (with the best and the median system respectively achieving 90.95% and 86.26%).

## 1 Introduction

Our participation in RTE-7 is based on the EDITS system[2] (Kouylekov and Negri, 2010). We have developed a new version (3.0) of the system used in the last edition of the challenge (Kouylekov et al., 2010b). It includes various modifications that improve the overall functionality, usability and performance of the system. Our modifications were motivated by an analysis of the problems that arised during user interaction with the system over the was 6 RTE challenges. Our overall goal was to create a system that can be easily adapted both as a sparring partner as demonstrated in (Kouylekov et al., 2011) as well as a useful library that can be integrated inside different applications. To achieve this goal we have created a fully automatic training procedure based on genetic algorithm described in this paper and evaluated in (Kouylekov et al., 2011).

To facilitate the usage of EDITS for system participating in RTE-7 the 3.0 version of system includes a script that converts the competition format into EDITS input.

We will describe the new version of the system in Section 2. In section 3 and 4 we will present its performance on the Main task, Novelty and KBP tasks.

## 2 EDITS 3.0

EDITS (Kouylekov and Negri, 2010) is an open source package for recognizing textual entailment, which offers a modular, flexible, and adaptable working environment to experiment with the RTE task over different datasets. The package allows to: *i)* create an entailment engine by defining its basic components (i.e. algorithms, cost schemes, rules, and optimizers); *ii)* train such entailment engine over an annotated RTE corpus to learn a model; and *iii)* use the entailment engine and the model to assign an entailment judgment and a confidence score to each pair of an un-annotated test corpus.

A key feature of EDITS is represented by its high configurability, allowed by the availability of different algorithms, the possibility to integrate different sets of lexical entailment/contradiction rules, and the variety of parameters for performance optimization (Mehdad, 2009).

Although configurability is *per se* an important aspect (especially for an open-source and general

[1] http://www.celi.it

[2] http://edits.sf.net

purpose system), there is another side of the coin. In principle, in order to select the most promising configuration over a given development set, one should exhaustively run a huge number of training/evaluation routines. Such number corresponds to the total number of configurations allowed by the system, which result from the possible combinations of parameter settings. When dealing with enlarging dataset sizes, and the tight time constraints usually posed by the evaluation campaigns, this problem becomes particularly challenging, as developers are hardly able to run exhaustive training/evaluation routines. As recently shown by the EDITS developers team, such situation results in running a limited number of experiments with the most "reasonable" configurations, which consequently might not lead to the optimal solution (Kouylekov et al., 2010b).

The need of a mechanism to automatically obtain the most promising solution on one side, and the constraints posed by the evaluation campaigns on the other side, arise the necessity to optimize this procedure. Along this direction, the objective is good a trade-off between exhaustive experimentation with all possible configurations (unfeasible), and educated guessing (unreliable). The remainder of this section tackles this issue introducing an optimization strategy based on genetic algorithms, and describing its adaptation to extend EDITS with the new functionality.

### 2.1 Genetic algorithm

Genetic algorithms (GA) are well suited to efficiently deal with large search spaces, and have been recently applied with success to a variety of optimization problems and specific NLP tasks (Figueroa and Neumann, 2008; Otto and Riff, 2004; Aycinena et al., 2003). GA are a direct stochastic method for global search and optimization, which mimics natural evolution. To this aim, they work with a *population of individuals*, representing possible solutions to the given task. Traditionally, solutions are represented in binary as strings of *0*s and *1*s, but other encodings (*e.g.* sequences of real values) are possible. The evolution usually starts from a population of randomly generated individuals, and at each generation selects the best-suited individuals based on a *fitness function* (which measures the optimality of the solution obtained by the individual). Such selection is then followed by *modifications* of the selected individuals obtained by recombining (crossover) and performing random changes (mutation) to form a new population, which will be used in the next iteration. Finally, the algorithm is terminated when the maximum number of generations, or a satisfactory fitness level has been reached for the population.

### 2.2 EDITS-GA

Our extension to the EDITS package, EDITS-GA, consists in an iterative process that starts with an initial population of randomly generated configurations. After a training phase with the generated configurations, the process is evaluated by means of the fitness function, which is manually defined by the user[3]. This measure is used by the genetic algorithm to iteratively build new populations of configurations, which are trained and evaluated. This process can be seen as the combination of: *i)* a micro training/evaluation routine for each generated configuration of the entailment engine; and *ii)* a macro evolutionary cycle, as illustrated in Figure 1. The fitness function is an important factor for the evaluation and the evolution of the generated configurations, as it drives the evolutionary process by determining the best-suited individuals used to generate new populations. The procedure to estimate and optimize the best configuration applying the GA, can be summarized as follows.

**(1) Initialization**: generate a random initial population (*i.e.* a set of configurations).

**(2) Selection**:

**2a.** The fitness function (*e.g.* accuracy, or F-measure) is evaluated for each individual in the population.

**2b.** The individuals are selected according to their fitness function value.

**(3) Reproduction**: generate a new population of configurations from the selected one, through genetic operators (cross-over and mutation).

**(4) Iteration**: repeat the *Selection* and *Reproduction* until *Termination*.

**(5) Termination**: end if the maximum number of iterations has been reached, or the population has converged towards a particular solution.

[3] For instance, working on the RTE Challenge "Main" task data, the fitness function would be the *accuracy* for RTE1 to RTE5, and the *F-measure* for RTE6.

It's worth to mention that, due to the nature of GAs, the iterative evolutionary process does not explore the entire search space, and is not guaranteed to converge to the best individual solution.

In order to extend EDITS with genetic algorithms, we used a GA implementation available in the JGAP tool[4]. In our settings, each individual contains a sequence of boolean parameters corresponding to the activation/de-activation of the system's basic components (algorithms, cost schemes, rules, and optimizers). The configurations corresponding to such individuals constitute the populations iteratively evaluated by EDITS-GA on a given dataset.

## 3 Main Task & Novelty

Given a corpus *C*, a hypothesis *H*, and a set of "candidate" entailing sentences for that *H* retrieved from *C* by the Lucene search engine, the RTE-7 main task consists in identifying all the sentences that entail *H* among the candidate sentences.

### 3.1 Training the system

As a first step in the training stage we created a set of entailment pairs of the type $TCand_x$-*H* for each hypothesis *H* and for each candidate sentence for that *H*. Then we initialized EDITS-GA with the following features of the system produced as boolean configuration alternatives:

1. Match two words if the two forms are equal (string value)
2. Match two words if the two lemmas are equal (string value)
3. Use Levenshtein Distance to match strings
4. Ignore Case when matching strings
5. Use IDF for Word weight. The Idf of words were calculated on the English version of Wikipedia - a free on-line encyclopedia.[5].
6. Use Stopwords - the words that were found as stopwords in a predefined list of 100 common English stopwords have weight equal to 0.
7. Use word size for weight - to the default word weight is add the number of characters of the word form.
8. Use T/H size for weight -to the default word weight of a word in T is add the number of words in H and vice versa.
9. The cost of deletion is equal to zero (used only for the token edit distance algorithm)
10. The Probability of Entailment Rules is always equal to 1.0
11. Use entailment rules extraxted from Lin Similarity - similarity rules database (Lin, 1998)
12. Use entailment extracted from Wordnet - an electronic lexical database (Fellbaum, 2004)

| Alternative | Main | Novelty | Default |
|---|---|---|---|
| Form | NO | NO | YES |
| Lemma | YES | YES | YES |
| Distance | NO | NO | NO |
| Ignore Case | NO | YES | NO |
| IDF | NO | NO | NO |
| Stopwords | YES | NO | NO |
| Word Size | NO | NO | NO |
| T/H Size | NO | NO | NO |
| Deletion=0 | NO | NO | NO |
| Probability=1 | NO | NO | NO |
| Lin | NO | NO | NO |
| Wordnet | NO | NO | NO |

Table 1: Optimal System Configuration Main & Novelty

The entailment rules were extracted following the approach described in (Kouylekov et al., 2010a).

The optimal configuration found by the EDITS GA is using the Word Overlap algorithm and the values for each alternative is presented in Table 1.

### 3.2 Testing the system

For the main and the novelty tasks we have submitted only one run for each. This run was produced with the optimal configuration found by EDTIS-GA. We have submitted for ablation the output produced with the default configuration system. The results obtained are presented in Table 2.

[4] http://jgap.sourceforge.net/

[5] http://www.wikipedia.org

| | Main | Main Ablation (Default) | Novelty |
|---|---|---|---|
| F1 | 44.10 | 39.38 | 87.21 |

Table 2: System Results Main & Novelty

## 4 KBP Task

The KBP task presented a significant challenge for EDITS. The system had to face significant difficulties in processing large number of pairs. Given a document *D*, and a set *S* of hypotheses *S*={$H_1$,...,$H_n$}, the KBP Validation Pilot task consists in determining if *D* entails *S*.

The task is situated in the Knowledge Base Population scenario, and aims at validating the output of the systems participating in the RTEKBP Slot Filling task by using Textual Entailment techniques. In this framework, *S* is a set of roughly synonymous sentences representing different linguistic realizations of a relation between a target entity, and a possible value (a.k.a. "slot-filler") of one of its attributes (a.k.a. "slots"). The assumption is that an extracted slot filler is correct if and only if the supporting document entails an hypothesis created on the basis of the slot filler.

### 4.1 Training the system

As in the Main task, KBP Validation training data were devided in two portions (TRAIN and DEV) in order to perform reliable routines of training and evaluating the learned models over unseen data. The documents of the dataset were then sentence-splitted, and used as the new training and test sets for the following steps of our experiments. A large entailment corpus of *T*-*H* pairs is created, where each pre-processed sentence in the document is paired with the corresponding *H*s. We have created a filter, used for all the submitted runs, which operates at the level of **documents** automatically discarding as possible entailing candidates all the sentences in a document that do not contain at least one word from the entity (*e.g.* at least "*Chris*" or "*Simcox*" for <*entity*>*Chris Simcox*<*entity*>), and one word from the value (*e.g.* at least "*Tucson*" or "*Ariz.*" for <*value*>*Tucson, Ariz.*<*value*>).

We have run the EDITS-GA to find the optimal configuration for the token edit distance, cosine similarity and word overlap algorithms on the training set. The optimal configurations discovered by the EDITS-GA algorithm are presented in Table 3.

| Word Overlap | | Cosine | Token Edit Distance |
|---|---|---|---|
| Form | YES | YES | NO |
| Lemma | YES | YES | YES |
| Distance | YES | YES | NO |
| Ignore Case | NO | NO | NO |
| IDF | YES | YES | NO |
| Stopwords | YES | YES | YES |
| Word Size | NO | NO | YES |
| T/H Size | NO | NO | NO |
| Deletion=0 | NO | NO | NO |
| Probability=1 | NO | NO | NO |
| Lin | NO | NO | NO |
| Wordnet | NO | NO | NO |

Table 3: Optimal System Configuration KBP

| 11 | cosine | overlap | token edit |
|---|---|---|---|
| F-Measure | 13.71 | 14.09 | 15.39 |

Table 4: System Results KBP

### 4.2 Testing the system

**Run 1.** For the first run the learned model was obtained by using the EDITS-ga produced configuration for the cosine similarity algorithm.

**Run 1.** For the first run the learned model was obtained by using the EDITS-ga produced configuration for the word overlap algorithm.

**Run 3.** For the third run the learned model was obtained by using the EDITS-ga produced configuration for the token edit distance algorithm.

The results obtained are show in Table 4

## 5 Discussion

We participated in the RTE-7 Main, Novelty and KBP Validation tasks with the latest release of EDITS (Edit Distance Textual Entailment Suite) an open source RTE package originally developed by FBK-irst and extended to its current (3.0) version by CELI.

The results obtained demonstrate that the EDITS-GA is capable of selecting a configuration that significantly improves the baseline system performance from 39 to 44 on the Main task. Our best run in the

Main task achieved a Micro-Averaged F-measure of 44.10% (with the best and the median system respectively achieving 48.0% and 41.90%); our best run in the Novelty task achieved a highest primary F-measure of 87.21% (with the best and the median system respectively achieving 90.95% and 86.26%). The system performance is above the median and is slightly behind from the the top system.

The results proofs once more that the Word Overlap is a difficult baseline to beat. Introducing resources of entailment rules does not improve the system performance and were consequently discarded by the EDITS-GA algorithm.

It is worth noticing that in the KBP task the token edit distance algorithm performed better than word overlap. In future system development we will explore a combination of EDITS with paraphrasing systems like (Quirk et al., 2094) in order to build a better entailment recognition system.

## Acknowledgments

This work has been partially supported by the EC-funded project Galateas (CIP-ICT PSP-2009-3-250430).

## References

Margaret Aycinena, Mykel J. Kochenderfer, and David Carl Mulford. 2003. An Evolutionary Approach to Natural Language Grammar Induction. *Stanford CS 224N Natural Language Processing*.

Christiane Fellbaum 2004. WordNet: An Electronic Lexical Database (Language, Speech, and Communication. *The MIT Press*

Alejandro G. Figueroa and Günter Neumann. 2008. Genetic Algorithms for Data-driven Web Question Answering. *Evolutionary Computation 16(1) (2008) pp. 89-125*.

Milen Kouylekov, Yashar Mehdad and Matteo Negri 2010. Mining Wikipedia for Large-scale Repositories of Context-Sensitive Entailment Rules. *Seventh international conference on Language Resources and Evaluation (LREC 2010)*.

Milen Kouylekov, Yashar Mehdad, Matteo Negri, and Elena Cabrio. 2010. FBK Participation in RTE6: Main and KBP Validation Task. *Proceedings of the Sixth Recognizing Textual Entailment Challenge*.

Milen Kouylekov and Matteo Negri. 2010. An Open-source Package for Recognizing Textual Entailment. *Proceedings of ACL 2010 Demo session*.

Milen Kouylekov, Yashar Mehdad and Matteo Negri 2011. Is it Worth Submitting this Run? Assess your RTE System with a Good Sparring Partner. *Proceedings of the TextInfer 2011 Workshop on Textual Entailment*.

Dekang Lin 1998. Automatic Retrieval and Clustering of Similar Words. *In Proceedings of COLING-ACL (1998)*

Yashar Mehdad 2009. *Automatic Cost Estimation for Tree Edit Distance Using Particle Swarm Optimization.* Proceedings of ACL-IJCNLP 2009.

Eridan Otto and María Cristina Riff 2004. Towards an efficient evolutionary decoding algorithm for statistical machine translation. *LNAI, 2972:438447.*.

Chris Quirk, Chris Brockett, and William B. Dolan 2004. Monolingual Machine Translation for Paraphrase Generation *Proceedings of ACL 2004*.