# Learning Textual Entailment on a Distance Feature Space

Maria Teresa Pazienza[1], Marco Pennacchiotti[1], and Fabio Massimo Zanzotto[2]

[1] University of Roma Tor Vergata, Via del Politecnico 1, Roma, Italy,
{pazienza, pennacchiotti}@info.uniroma2.it,
[2] DISCo, University of Milano Bicocca, Via B. Arcimboldi 8, Milano, Italy,
zanzotto@disco.unimib.it

**Abstract.** Textual Entailment recognition is a very difficult task as it is one of the fundamental problems in any semantic theory of natural language. As in many other NLP tasks, Machine Learning may offer important tools to better understand the problem. In this paper, we will investigate the usefulness of Machine Learning algorithms to address an apparently simple and well defined classification problem: the recognition of Textual Entailment. Due to its specificity, we propose an original feature space, the *distance feature space*, where we model the distance between the elements of the candidate entailment pairs. The method has been tested on the data of the Recognizing Textual Entailment (RTE) Challenge.

## 1 Introduction

The task of recognizing if a textual expression, the text $T$, entails another expression, the hypothesis $H$, is a very difficult challenge. Indeed, as described in [1], Textual Entailment (TE) recognition (as referred in [2]) can be seen as the basic ability that any semantic theory for natural languages must have. Only those semantic theories able to detect textual entailment can be considered correct. Techniques for detecting textual entailment can be then seen as a first step in the building process of a semantic theory for natural language. We are thus facing a very complex problem.

Textual Entailment recognition can be tackled using two main strategies: *shallow and robust techniques* mostly based on shallow and probabilistic textual analysis, or more complex *syntactic-based models* involving a deeper syntactic analysis and the use of ad-hoc rules.

*Shallow and robust models* are based on the assumption of independence among words. They can be used to tackle Textual Entailment recognition, as it has already been done in several difficult open domain NLP tasks (e.g. Text Categorization and Information Retrieval). Even if these simple and effective models are still far from being perfect, they usually outperform more sophisticated techniques based on deep syntactic and semantic properties. For instance, in Text Categorization bag-of-word models seem to be more successful than any

other approach based on more complex "linguistic" features [3], even if in specific domain (e.g. the medical domain) performances are not so high [4].

As in Text Categorization, shallow models seem to be the most promising also for Textual Entailment recognition. Indeed, some top-performing approaches to TE strongly rely on the assumption that every word is independent from the other, as in the probabilistic textual entailment model presented in [5] and in the lexical model presented in [6]. The first approach [5] has been shown to have one of the highest performances in the 2005 Pascal Challenge on *Recognising Textual Entailment* (RTE) [7]. It assumes that the entailment $T \rightarrow H$ holds if the posterior probability $P(H|T)$ of $H$ being true given $T$ is higher than the prior probability $P(H)$. The actual probabilistic textual entailment indicator $P(H|T)$ is then evaluated at word level $P(w_H|w_T)$ and recombined using the word independence assumption. $P(w_H|w_T)$ is estimated over a large textual collection (i.e. the web) assuming that the joint event $(w_H, w_T)$ is verified when the two words, $w_H$ and $w_T$ are found in the same document, while $P(H)$ is set to a fix value for all pairs. The second approach [6] uses a very similar model (even if not probabilistic) where the distance between the two words, $w_H$ and $w_T$, is estimated over a semantic hierarchy (i.e. WordNet, [8]) with different semantic similarity measures (e.g. [9, 10]).

These approaches, even if top-performing, are still far from offering a satisfactory solution to the Textual Entailment task. In fact, on the RTE dataset [7] they reached an accuracy of less than 60%. The manual study in [11] suggests that there is room for improvement if more complex text representations are used than those based on independence assumption. According to this investigation, 49% cases of the RTE dataset can be correctly predicted using syntactic clues that consider the dependence between words in the sentence. Thus, the use of a *perfect* parser and syntactic constraint checking techniques would correctly predict (100% accuracy) all the 49% pairs identified in the study. Using a random guesser (50% accuracy) on the remaining 51% pairs, the overall accuracy would increase to 74%. Models based on structural syntactic information ("syntax-based models") should then guarantee a high margin of improvement.

*Syntax-based models*, generally rely on the notion of distance or similarity between the syntactic representations of the two text fragments T and H, as it usually happens for some lexical model. Moreover, sometimes these models include also semantic information (e.g. WordNet [8]). The approaches presented in [12] and in [13] define a distance between dependency syntactic graphs: these approaches, while being pretty similar, use different syntactic representations and parsers. Moreover, in [14] a variant of the edit distance, the *tree edit distance*, is applied to estimate differences between structures: this distance is defined on the dependency trees produced by the syntactic parser described in [15].

Syntax-based models are still far from the performance manually estimated in [11] (the "perfect" syntactic parser is still not available). Moreover, even using a "perfect" parser, it may happen that rules for detecting entailment, written in these syntactic models, are inadequate as not totally correct or not covering all entailment cases.

A solution to the inadequacy of manually built rules adopted in several NLP applications (e.g. Information Extraction [16]) is the application of supervised or semi-supervised machine learning algorithms able to extract weighted rules using set of training examples. Syntax-based models, that use Machine Learning to derive rules, appear promising, as they have the potential to improve and outperform shallow approaches in Textual Entailment recognition.

Following this approach, in this paper we investigate the possibility of using machine learning models to tackle the problem of Textual Entailment recognition. The aim is to understand whether or not machine learning represents a practical and promising way to improve the performance of a generic recognition model. It is evident that the application of classical machine learning algorithms to the Textual Entailment task is not straightforward, mainly because of the small number of training examples. As we believe that this number can never be large enough to learn significant regularities in a simple direct feature space (see Sec. 2), we propose an alternative feature space based on the distance between the elements of textual entailment pairs. We will discuss this distance feature space in Sec. 4. Finally, in Sec. 5, we will evaluate our model using SVM-light [17] on the RTE data [7].

## 2 Classifiers, Machine Learning, and Textual Entailment

Machine Learning algorithms learn how to classify instances into categories. Instances are seen in a feature space and the role of the algorithm is to learn regularities that help in the classification. The learnt final function $T$ maps elements in the feature space $F_1 \times \ldots \times F_n$ to one element in the final set of categories $C$, that is:

$$T : F_1 \times \ldots \times F_n \rightarrow C$$

where $F_1$, ..., $F_n$ are the features for observing instances.

One of the main issues in applying a learner to a particular classification task is to identify the most suitable feature space. However, as we will see later in this section, this is not the only problem when trying to exploit classifiers for the Textual Entailment task. For example, it is not clear which is the most suitable set of categories $C$ and, as a consequence, which equivalences should be learnt by the algorithms. In this section we analyse these problems, in order to explain the rationale behind our *distance feature space*.

At a first glance, the Textual Entailment task seems to be a clearly defined classification problem. Instances are represented as triples $(T, H, value)$ where $T$ is the text, $H$ is the hypothesis, and where $value$ is $true$ if $T \rightarrow H$ holds and $false$ otherwise. For example:

(1) **T**: *The carmine cat devoured the mouse in the garden*
**H**: *The cat killed the mouse*
**value**: *true*

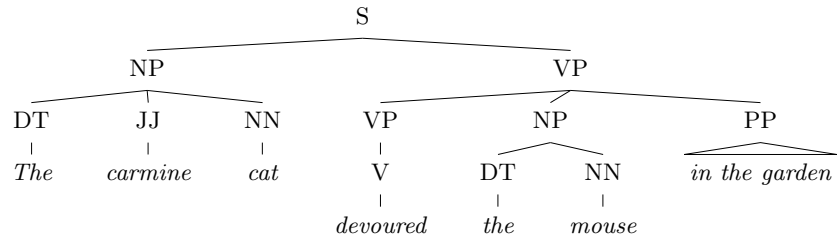
This definition can lead to very simple feature spaces based on the bag-of-word abstraction. A good example is given by a bipartite bag-of-word feature space

$FS_T \times FS_H$, where $FS_T$ is related to the text $T$ while $FS_H$ is related to the hypothesis $H$. $FS_T$ and $FS_H$ are two complete bag-of-word feature spaces, where dimensions are words and values are 1 or 0 (i.e., a specific word is present or not in the text fragment $T$ or $H$). Finally, the classification is on the $\{true, false\}$ categories, that is, the pair is a positive or a negative instance of the entailment relation.

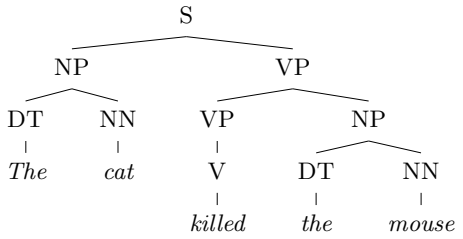
This *bipartite feature space* $FS_T \times FS_H$ is, in principle, inadequate for a small number of training examples and for the classification problem as it has been defined. Indeed, in this space, a learner is unlikely to infer useful properties. For instance (relating to previous example), the learner can infer that, if there is *cat* and *mouse* both in $FS_T$ and in $FS_H$, $T$ and $H$ can be positive instances of an entailment pair. This information is quite sparse and the solution cannot be obtained from adding the syntactic interpretation or some lexical semantic abstraction in the feature space.

In particular, tree kernels [18, 19] can be used to integrate syntactic information in the feature space; unfortunately, this is not a feasible solution, because of the sparseness of the space. For example, the possible syntactic interpretations of the sentences in the example (1) are:

(2) **T**:

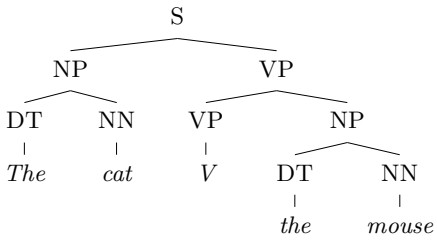



**H**:

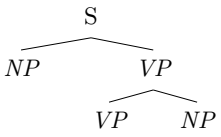



The regularities that can be learnt from this pair by a tree kernel are either too specific, e.g. both $T$ and $H$ should have the subtree (3), or too general, e.g. the subtree (4).

(3)



(4)




When working with a small number of examples, the equivalence between the elements in the two classes (entailed and not entailed pairs), may only be found at the level of general syntactic properties as the one in (4). This latter property may be read as: *both $T$ and $H$ should have a sentence that has a main verb and two direct arguments.* As such, this property is not a very significant clue to detect whether or not $T$ entails $H$, as it is too general. In the same way, too specific properties are not useful, as they lack generalization power and can thus be applied only to a small set of cases. For example the property in (3) states: *both $T$ and $H$ should have a sentence that has a main verb and two direct arguments, whose values must be "the cat" and "the mouse".*

In order to extract more useful properties for the TE classification task (neither too generic nor too specific) , the number of training examples should be bigger and bigger, and the classification problem should be differently formulated. In fact, the equivalence classes to be discovered should be more complex than the simple *true* and *false* entailment prediction. They should be centred on the hypothesis $H$: that is, in principle, each $H$ should correspond to an equivalence class. By consequence, the whole process of verifying if entailment holds for a (T;H) pair is divided in two steps. Firstly, the most suitable equivalence class is chosen for a fragment $T$, using ML classification. Then, entailment is said to hold if the predicted class corresponds to the H of the pair (T;H).

The step of categorising in equivalence classes representing possible $H$ is a neat shift with respect to the $true/false$ classification approach. As such, the classification subtask needs different types of features. Indeed, the feature set can not model the properties of the fragments $T$ and $H$ independently as for the bipartite feature space: in fact, the aim here is to find the equivalence class $H$ that is most similar to a given instance $T$. A sort of distance should then be evaluated between the instance $T$ (to be classified), and the elements of each possible class $H$. The feature space can then be the classical one as the text $T$ and the hypothesis $H$ should be modelled as different points.

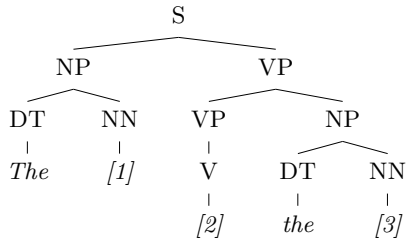
Even if it seems an infeasible solution, due to the infinite number of possible $H$, this technique is adopted in learning a large number of equivalence classes often called *inference rules* [20]. We will use the *distributional hypothesis* [21] to cluster instances around an equivalence class that is the *inference rule* [20, 22]. Unluckily, these methods can not solve the problem in our case, as they need a large set of examples, that can be met only in very large corpora.

As a consequence, it emerges that the direct representation of $H$ and $T$ in a simple or a bipartite feature space is not an interesting solution. What is relevant should then be found elsewhere.

Let us consider again example (1). The interesting property of $H$ and $T$ is that they share two words, *cat* and *mouse*, not the specific words shared.

The same consideration applies to the syntactic feature space (2). The relevant property is that $H$ and $T$ share the structure:

(5)




and that in position *[1]* and *[3]* they have the same words. The relevant property is then that $H$ and $T$ share a similar structure and that they share similar values in strategic positions of the structure itself. The actual values of *[1]* and *[3]* are not important, while it is important that they have similar meaning. A sort of syntactic-semantic "distance" between $T$ and $H$ could be then a more suitable approach to really grasp the notion of TE, as it is done in [12–14].

We propose to study the learnability on a feature space representing the distance between the two elements in the candidate entailment pair, instead of on the previously described bipartite feature space . This should make the entailment problem learnable with a small number of training examples as the problem is not studied on the actual values of the examples but on their distance. In the following section we describe both the formalisms and techniques to model the notion of distance between $T$ and $H$, while in Sec. 4 we will introduce our *distance feature space*.

## 3 Modelling Textual Entailment as Syntactic Graph Similarity

Here we introduce the model to investigate the similarity between syntactic graphs originally used to directly detect entailment. Furthermore, the model can also be used to build the distance feature space we are interested in, transforming textual entailment into a learnable classification problem. The distance feature space will be described in Sec. 4.

As textual entailment is mainly concerned with syntactic aspects (as outlined in [2] and observed in [11]), a model for its recognition should basically rely on lexical and syntactic techniques, enriched only with shallow semantic analysis.

We model the entailment pair $T - H$ as two *syntactic graphs* augmented with lexical and shallow semantic information. Each graph *node* represents a phrase of the sentence, together with its syntactic, morphological and semantic information. Each graph *edge* expresses a syntactic relation among phrases. We can thus consider the recognition process as the comparison between two graphs: the more similar the graphs are, the higher is the probability of entailment. A *similarity measure* among graphs can be meant as a measure of entailment, and *graph matching theory* can be used as a tool to verify if entailment relation holds.

In the following section we introduce the basic notions on graph theory needed to describe how we implemented the graph similarity measure. Then, in Sec.3.2 we introduce the Extended Dependency Graph (XDG), the syntactic graph formalism we rely on. Peculiar properties of textual entailment that require specific adaptations of graph matching theory are then outlined in Sec.3.3. Finally, the model for textual entailment recognition is described in Sec.3.4, together with the strategy we adopted to deal with syntactic transformations (Sec.3.5).

### 3.1 Graph Matching Theory

Graph matching theory aims to evaluate the similarity between two graphs. The power of graph matching theory resides in the generality of graphs, as they can be used to represent roughly any kind of objects. Graph matching is then used in many different settings, as vision applications (such as video indexing [23] and 3D object recognition [24]), case-based reasoning [25] and planning [26]. Graph nodes usually represent object parts, while edges represent relations among parts. Matching algorithms recognize how similar two objects are, looking at the structural similarity of their graph representation. It is thus possible, for instance, to turn a recognition problem of an unknown visual object, into a graph matching task over a given repository of known instances.

In the rest of this section we firstly outline basic definitions of graph matching theory, as presented in [27], and then briefly discuss how graph similarity is evaluated in practice.

**Definition 1.** *A* graph *is defined as 4-tuple $G = (N, E, \mu, \nu)$, where $N$ is the finite set of labelled nodes, $E$ the finite set of labelled edges connecting the nodes in $N$, $\mu : N \rightarrow L_N$ the function that assigns labels to nodes, and $\nu : E \rightarrow L_E$ the function that assigns labels to edges.*

**Definition 2.** *A* graph isomorphism *is a bijective function $f : N \rightarrow N'$, from a graph $G = (N, E, \mu, \nu)$ to a graph $G' = (N', E', \mu', \nu')$, such that:*

- *$\mu(n) = \mu'(f(n))$ for all $n \in N$*
- *for any edge $e \in E$ connecting two nodes $n_1$, $n_2$, it exists an edge $e'$ connecting $f(n_1), f(n_2)$, and vice versa.*

**Definition 3.** *A* subgraph isomorphism *is an injective function $f : N \rightarrow N'$, from $G = (N, E, \mu, \nu)$ to $G' = (N', E', \mu', \nu')$, if it exists a subgraph $S \subseteq G'$ such that $f$ is a graph isomorphism from $G$ to $S$.*

**Definition 4.** *A graph $G$ is called* common subgraph *between two graphs $G_1$ and $G_2$ if it exist a subgraph isomorphism from $G_1$ to $G$ and from $G_2$ to $G$.*

**Definition 5.** *The common subgraph $G$ of $G_1$ and $G_2$ with the highest number of nodes is called the* maximal common subgraph *($mcs(G_1, G_2)$).*

The concept of *maximal common subgraph* (*mcs*) is often central in the definition of a *similarity measure*. In fact, in real applications, errors and distortions in the input graphs are frequent. Consequently, as perfect matching between two objects becomes often impossible, graph matching algorithms must be error tolerant, returning as result a degree of similarity between graphs, rather than a deterministic matching value. As an example, in [28] a similarity measure between two graphs $G_1$ and $G_2$ is proposed as a *distance* between the number of nodes of the *mcs* and the number of nodes of the biggest graph:

$$d(G_1, G_2) = 1 - \frac{|mcs(G_1,G_2)|}{max(|G_1|,|G_2|)}$$

### 3.2 The XDG formalism

In the context of textual entailment, graph matching theory can be applied to two graphs representing the syntactic structure of $T$ and $H$, together with relevant lexical information. As useful syntactic representation we decided to use the extended dependency graph (XDG) [29]. An XDG is a dependency graph whose nodes $C$ are *constituents* and whose edges $D$ are the *grammatical relations* among the constituents, i.e. $\mathcal{XDG} = (C, D)$.

The XDG formalism has two interesting properties: it hides unnecessary ambiguity in possibly underspecified constituents and it may represent alternative interpretations in a single graph.

Constituents (i.e. $c \in C$) are classical syntactic trees with explicit *syntactic heads*, $h(c)$, and *potential semantic governors*, $gov(c)$. Constituents can be represented as feature structures, having as relevant features:

- the *head* and the *gov*, having as domain $\mathcal{C}$ (the set of trees and subtrees derived from $C$), and representing respectively *syntactic heads* and *potential semantic governors*;
- the *type* representing the syntactic label of the constituent and having as domain $\Lambda$.

Moreover, a constituent can be either *complex* or *simple*. A *complex constituent* is a tree containing other constituents as sons (which are expressed by the feature *subConstituents*). A *simple constituent* represents a leaf node, i.e., a token span in the input sentence, that carries information about lexical items through the following features:

- $surface$, representing the actual form found in the token span,
- $lemma$, taking values in the lexicon $\mathcal{L}$ and representing the canonical form of the target surface,
- *morphology*, representing the morphological features of the inflected form.

On the other hand, dependencies in $(h, m, T) \in D$ represent typed (where $T$ is the type) and ambiguous relations among a constituent, the *head* $h$, and one of its $modifiers$ $m$. The ambiguity is represented using $plausibility$, a real value ranging between 0 and 1, where 1 stands for unambiguous. Then, $D$ is defined as

a subset of $C \times C \times \Gamma \times (0, 1]$, where the sets represent respectively the domains of the features *head*, $modifier$, *type*, and $plausibility$.

The syntactic analysis of entailment couples has been carried out by Chaos [29], a robust modular parser based on the XDG formalism.

### 3.3 Adapting XDG and Graph Matching to Textual Entailment

In Sec.3.1 and Sec.3.2 a general methodology (graph matching) and a formalism (XDG) for recognizing textual entailment have been proposed. Entailment recognition can thus be described as a matching process between two XDG graphs representing the hypothesis and the text, where nodes are the set of constituents $C$ and edges are the set of dependencies $D$. In order to obtain a model of the process, it is necessary to verify if either the chosen formalism and methodology satisfy all the characteristics needed for entailment recognition or they need some adaptation.

Concerning the XDG formalism, it is necessary to define the specific kind of information that a graph for entailment recognition must hold. Then, it must be verified if XDG graphs are able to capture all this information. In order to be detected, entailment requires both syntactic and shallow lexical-semantic information:

- *syntactic information*: in general, graphs that have similar syntactic and lexical structure are likely to express the same fact. Moreover, syntactic addition to the $T$ graph with respect to $H$ can reveal a strict entailment relation, as capturing *syntactic subsumption entailment*. Finally, syntactic variations such as nominalization and active/passive transformations must be treated as invariant operations on graphs, since the meaning of the fact expressed is preserved. A graph is thus required to represent both syntactic dependencies and syntactic variations.
- *shallow lexical-semantic information*: syntactic similarity can be supported by lexical-semantic information needed to grasp *semantic subsumption entailment*, such as verb and noun generalization, antinomy and synonymy. Moreover, *direct implication* requires the recognition of verb entailments. As all these information must be modelled in the graph similarity measure, graphs must express all the morphological and lexical properties needed to carry out semantic operations, such as word stems and derivational properties.

The XDG formalism captures all needed information, as syntactic dependencies are explicitly represented, and lexical information about nodes is carefully treated.

Regarding the graph matching methodology, it seems suitable for the textual entailment task. In fact, a classical graph matching problem and textual entailment reveal some similarities:

- there are complex objects to be matched, composed by independent parts (represented as graph nodes) and relations among them (represented as edges).

– in order to tackle errors and distortion, it is better to adopt a similarity measure able to express the degree of similarity between two objects (e.g. using *mcs*), rather than a deterministic value.
– meta-operation can be performed over collection of objects, such as *graph clustering* and *graph querying*. For example, textual entailment clustering of $H$ and $T$ sentences can be viewed as a sort of semantic generalization of surface forms into semantic relation ([30]).

However, some peculiar properties of textual entailment require major adaptations of the standard graph matching methodology:

– *Node complexity*. In the graph theory, nodes are matched by simply looking at their *label level*. In textual entailment node similarity can not be reduced to a surface analysis, as both morphological and semantic variations must be taken into account (for example *ate*, *has eaten* and *devour* should have some degree of similarity). Moreover, textual entailment nodes are not atomic, since they represent complex constituents that can be further divided in sub-constituents for deeper lexical-semantic analysis. For these two reasons, matching between two nodes is a complex process, that can not produce a simple true value (as it usually happens at the label level). It is necessary to evaluate a graded level of linguistically motivated *node semantic similarity* $sm(c_h, c_t)$.
– *Edge complexity*. Edges are complex structures too: the matching over them must consider also the type of dependency they express. A graded *syntactic similarity* $ss(c_h, c_t)$ has then to be defined to capture this aspects.
– *Transformation invariance*. Textual entailment must account for graph invariant transformations: specific type of syntactic phenomena (nominalization, active/passive transformation, etc.) should be properly treated. Two graphs representing syntactic variations of the same fact, while structurally dissimilar, should be considered as equivalent.
– *Asymmetry*. Textual entailment, unlike the classical graph problems, is not symmetric, since it represents a direct relation of subsumption from $T$ to $H$. By consequence, the *graph isomorphism* definition must be further refined in a more specific notion of *XDG subsumption isomorphism*.

Considering these remarks, definition in Sec. 3.1 have been extended as follows.

**Definition 6.** *An* XDG subsumption isomorphism *is an oriented relation from a text* $\mathcal{XDG}_{\mathcal{T}} = (C_T, D_T)$ *to an hypothesis* $\mathcal{XDG}_{\mathcal{H}} = (C_H, D_H)$ *(*$\mathcal{XDG}_H \preceq \mathcal{XDG}_T$*), expressed by two bijective functions:*

– $f_C : C_T \rightarrow C_H$
– $f_D : D_T \rightarrow D_H$

*where* $f_C$ *and* $f_D$ *describe the oriented relation of subsumption between constituents (nodes) and dependencies (edges) of* $H$ *and* $T$*.*

$f_C$ and $f_D$ play the role of function $f$ in the definition of *graph isomorphism* in Sec. 3.1. Unluckily, due to the *node and edge complexity* factors, a definition of $f_C$ and $f_D$ can not be easily stated as for $f$. Sec. 3.4 will thus give an extensive description on how these two functions are modelled.

**Definition 7.** *A* subgraph subsumption isomorphism *between* $\mathcal{XDG}_H$ *and* $\mathcal{XDG}_T$*, written as* $\mathcal{XDG}_H \sqsubseteq \mathcal{XDG}_T$*, holds if it exists* $\mathcal{XDG}'_T \subseteq \mathcal{XDG}_T$ *so that* $\mathcal{XDG}_H \preceq \mathcal{XDG}'_T$.

As in the graph matching theory, an *mcs* must be defined in order to cope with distortions and errors in the input graphs. Specifically, in entailment recognition, errors are mainly introduced by syntactic parser erroneous interpretations, both at the morphological and syntactic level.

**Definition 8.** *The* maximal common subsumer subgraph *(*mcss*) between* $\mathcal{XDG}_H$ *and* $\mathcal{XDG}_T$ *is the graph with the highest number of nodes, among all the subgraph of* $\mathcal{XDG}_H$ *which are in isomorphic subgraph subsumption with* $\mathcal{XDG}_T$.

### 3.4 Graph Syntactic Similarity Measure for Textual Entailment

The similarity measure $\mathcal{E}(\mathcal{XDG}_T, \mathcal{XDG}_H)$, used to estimate the degree of confidence with which $\mathcal{XDG}_H$ and $\mathcal{XDG}_T$ are in entailment relation, must be modelled on the subsumption between nodes and edges in $T$ and $H$, grasping the notion of *mcss*. Four main steps are required:

1. *Model the bijective function* $f_C : C'_T \rightarrow C'_H$, that maps constituents in $C'_H \subseteq C_H$ to subsuming constituents in $C'_T \subseteq C_T$. A semantic similarity $sm(c_h, c_t)$ must be associated to each mapping. For example in the pair $H$:[the cat eats the mouse], $T$:[the cat devours the mouse], *eats* could be mapped in *devours*.
2. *Model the bijective function* $f_D : D'_T \rightarrow D'_H$, that maps dependencies in $D'_H \subseteq D_H$ to dependencies in $D'_T \subseteq D_T$. A syntactic similarity $ss(c_h, c_t)$ is then derived to better capture the implications of such mappings.
3. *Find the mcss*, that is, the common subgraph identified by $f_C$ and $f_D$. The *mcss* must be associate to an overall similarity, deriving from the *sm* and *ss* of its nodes and edges.
4. *Model* $\mathcal{E}(XDG_T, XDG_H)$ using *mcss* and the two input graphs $\mathcal{XDG}_H$ and $\mathcal{XDG}_T$. Textual entailment between a pair $T - H$ will be thus predicted verifying $\mathcal{E}(XDG_T, XDG_H)$ against a manually tuned threshold.

In the following paragraphs the different steps are described in detail. For an extensive example refer to Figure 1.

**Node Subsumption** The node subsumption function $f_C$ must identify constituents in $C_H$ that can be mapped to constituents $C_T$. We will define the function with a set $A$ containing the *anchors* (the correspondences between the constituents of $C_H$ and $C_T$). The set $A$ will thus represent the nodes of the *mcss*.

In $A$, each constituent $c_h \in C_H$ is associated, where possible, to its most similar constituent $c_t \in C_T$ (that is, the $c_t$ that most likely subsumes $c_h$). The definition follows:

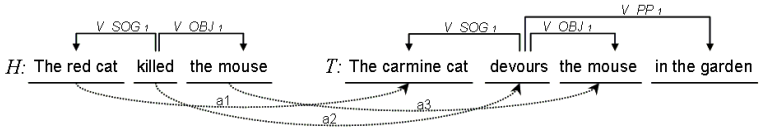



**Fig. 1.** A complete example of entailment pair, represented in the XDG formalism. Solid lines indicate grammatical relations $D$ (with *type* and *plausibility)*; dotted lines indicate anchors $a_i$ between $H$ and $T$ constituents.

**Definition 9.** *Given the* anchors $a = (c_h, c_t)$ *as linking structures, connecting constituents* $c_h \in C_H$ *to constituents* $c_t \in C_T$ *and a function of* semantic similarity $sm(c_h, c_t) \in (0, 1]$ *expressing how much similar* $c_h$ *and* $c_t$ *are looking at their lexical and semantic properties, the set of anchors A is:*

$$A = \{(c_h, c_t) | c_h \in C_H, c_t \in C_T, sm(c_h, c_t) = \max_{c \in C_T} sm(c_h, c) \neq 0\}$$

If a subsuming $c_t$ can not be found for a $c_h$ (i.e. $\max_{c \in C_T} sm(c_h, c) = 0$), then $c_h$ has no anchors. For example in the entailment pair of Fig. 1, $f_C$ produces the mapping pairs *[The red cat - The carmine cat], [killed - devours], [the mouse - the mouse].*

The semantic similarity $sm(c_h, c_t)$ is derived on the basis of the syntactic type of $c_h$, that is, if it is a noun-prepositional phrase $sm(c_h, c_t) = sm_{np}(c_h, c_t)$ or a verb phrase $sm(c_h, c_t) = sm_{vp}(c_h, c_t)$. If $c_h$ is a noun-prepositional phrase, similarity $sm_{np}(c_h, c_t)$ is evaluated as:

$$sm_{np}(c_h, c_t) = \alpha * s(gov(c_h), gov(c_t)) + (1 - \alpha) * \frac{\sum_{sh \in S(c_h)} \max_{s_t \in S(c_t)} s(s_h, s_t)}{|S(c_h)|}$$

where $gov(c)$ is the governor of the constituent $c$, $S(c_h)$ and $S(c_t)$ are the set simple constituents excluding the governors respectively of $c_h$ and $c_t$, and $\alpha \in [0, 1]$ is an empirically evaluated parameter used to weigh the importance of the governor. Meanwhile, $s(s_h, s_t) \in [0, 1]$ expresses the similarity among two simple constituents: it is maximal if they have same surface or stem (e.g. *cat* and *cats*), otherwise a semantic similarity weight $\beta \in (0, 1)$ is assigned looking at possible WordNet relations (synonymy, entailment and generalization). For example, setting $\beta = 0.8$ for synonymy and $alpha = 0.5$, the constituents *ch:* [the red cat] and *ct:* [the carmine cat], have $sm = 0.95$, as the governors ($gov_{ch} = cat$ and $gov_{ct} = cat$) and the first simple constituents ($sh_1 = the$ and $st_1 = the$) have $s = 1$, and the second simple constituents are synonyms ($sh_2 = red$ and $st_1 = carmine$).

If $c_h$ is a verb phrase, different *levels* of similarity are taken into consideration, according to the semantic value of its modal. For example *must go-could go* should get a lower similarity than *must go-should go.* A verb phrase is thus composed by its governor *gov* and its modal constituents *mod.* The overall similarity is thus:

$$sm_{vp}(c_h, c_t) = \gamma * s(gov(c_h), gov(c_t)) + (1 - \gamma) * d(mod(c_h), mod(c_t))$$

where $d(mod(c_h), mod(c_t)) \in [0, 1]$ is empirically derived as the semantic distance between two modals (e.g., *must* is nearer to *should* than to *could*) (classified as generic auxiliaries, auxiliaries of possibility and auxiliaries of obligation).

**Edge Subsumption** Once $f_C$ is defined, the existence of the bijective fuction $f_D$ can be easily verified by construction. The edge subsumption function $f_D$ maps $(c_h, c'_h, T_h) \in D_H$ to $f_D(c_h, c'_h, T_h) = (c_t, c'_t, T_t) \in D_T$ if $T_h = T_t$ and $(c_h, c_t), (c'_h, c'_t) \in A$. The set of mapped $D_H$ will thus represent the edges linking the nodes of the *mcss*.

The definition of $f_D$ allows to investigate the external *syntactic similarity* $ss(c_h, c_t)$ of a certain anchor $(c_h, c_t) \in A$. This should capture the similarity of the relations established by elements in the anchor. Our syntactic similarity $ss(c_h, c_t)$ depends on the semantic similarity of the constituents connected with the same dependency to $c_h$ and $c_t$ in their respective $\mathcal{XDG}$s, that is, the set $A(c_h, c_t)$ defined as:

$$A(c_h, c_t) = \{(c'_h, c'_t) \in A | f_D(c_h, c'_h, T) = (c_t, c'_t, T)\}$$

For example in Fig. 1, $A(killed, devours)$= {([*the_red_cat*], [*the_carmine_cat*]), ([*the_mouse*], [*the_mouse*])}. The syntactic similarity $ss(c_h, c_t)$ is then defined as:

$$ss(c_h, c_t) = \frac{\sum_{(c'_h, c'_t) \in A(c_h, c_t)} sm(c'_h, c'_t)}{|D_H(c_h)|}$$

where $D_H(c_h)$ are the dependencies in $D_H$ originating in $c_h$.

**Similarity measure** Once nodes and edges of the *mcss* have been identified through $f_C$ and $f_D$, an overall similarity $S(mcss)$ is evaluated for *mcss*. $S(mcss)$ must express how much similar the two subgraphs $\mathcal{XDG}'_T$ and $\mathcal{XDG}'_H$ in isomorphic subsumption are, both from a syntactic and a semantic point of view.

For each pair $(c_h, c_t) \in A$ a global similarity $S$ is thus derived as:

$$S(c_h, c_t) = \delta * sm(c_h, c_t) + (1 - \delta) * ss(c_h, c_t)$$

where $\delta$ is a manually tuned parameter. The similarity measure $\mathcal{E}(\mathcal{XDG}_T, \mathcal{XDG}_H)$ can be evaluated in analogy to the measure described in Sec. 3.1. In this specific case, numerator and denominator will not be expressed as the number of nodes, but as probabilities, since, as stated before, textual entailment must account for node and edges complexity. The numerator will thus be the overall *mcss* similarity . The denominator will express the best case, in which *mcss* corresponds to $\mathcal{XDG}_{\mathcal{H}}$, and all nodes and edges match with probability 1 to elements of a hypothetic $T$.

$$\mathcal{E}(\mathcal{XDG}_T, \mathcal{XDG}_H) = \frac{S(mcss)}{|C_H|} = \frac{\sum_{(c_h, c_t) \in A} S(c_h, c_t)}{|C_H|}$$

### 3.5 Graph Invariant Transformations

Entailment pairs are often expressed through syntactic variations. For example in the pair:

$H$:[The cat killed the mouse], $T$:[The killing of the mouse by the cat]

entailment is syntactically express by a nominalization. We had thus to model some of the basic and most important variation phenomena in our system, in order to cope with pairs with different syntactic structures used and expressing the same fact. That is, a graph matching must be guaranteed for $H$ and $T$ when they have two different syntactic graphs which are one the syntactic variation of the other. Before the graph matching procedure can be activated, a set of graph transformation rules have been applied to $\mathcal{XDG}_H$ and $\mathcal{XDG}_T$, in order to normalize form sentences that have a syntactic variation. For example in the abovementioned example, the text is brought back to the normal form $T$:[the cat killed the mouse]. We modelled the following type of invariant transformation:

- *nominalization in* $T$. Different cases such as $T$:[The killing of the mouse by the cat] and $T$:[The cat is the killer of the mouse] are treated. Only nominalization of $T$ is taken into consideration, as usually in entailment relations nominalization happens only in $T$;
- *passivization in* $H$ *or* $T$. Passive sentences are brought to active forms, e.g. $H$:[the cat eats the mouse], $T$:[the mouse is eaten by the cat];
- *negation in H or T.* If one sentence is the negative form of the other, the two sentences are recognized to be not in entailment (*negative subsumption*). Negation can be expressed in different ways. For example $H$:[the cat eats the mouse], could have two simple negative counterparts T:[the cat doesn't eat the mouse] or T:[the cat does not eat the mouse].

Due to the RTE deadlines, at the time of the experiments presented in this paper only the above mentioned syntactic normalization were integrated in the system. In order to improve performance and coverage of the entailment phenomenon, much more normalization rules should be taken into account. Argument movements, verb subcategorization frames and other syntactic inversion could in fact play a crucial role in the recognition process. In this line, two major issues should be addressed.

On the one hand, it could be useful to study extensively the relevance and the impact of syntactic normalizations on the Textual Entailment task, in order to understand if the integration of fine-grained normalizations is feasible and worthwhile. In [11] a first analysis of syntactic alternations in the RTE corpus was carried out. It resulted that the most frequent alternation (24% of all entailment pairs) was simple *apposition*, e.g. $H$:[the cat, killer of the mouse], $T$:[the cat is the killer of the mouse]. Our system properly treated such cases as a generic alternation.

On the other hand, it could be useful to investigate into the use of specific grammars, such as the Inversion Transuction Grammars [31], to better formalize and handle such normalization phenomena in a real operational framework.

## 4 A Distance Feature Space for Textual Entailment

The syntactic similarity measure we proposed in the previous section is a possible detector of entailment between $T$ and $H$ (once the acceptability threshold has been set) while it is also a good base to build the *distance feature space*.

A simple distance feature space may be defined with one only feature: the similarity between $H$ and $T$ with the value $\mathcal{E}(\mathcal{XDG}_T, \mathcal{XDG}_H)$. In this case, the target of the learning problem would be setting the threshold. However, as it emerges from the previous sections, this trivial distance feature space suffers the limits of the measure $\mathcal{E}(\mathcal{XDG}_T, \mathcal{XDG}_H)$. The measure depends on many parameters ($\alpha$, $\beta$, $\gamma$, and $\delta$) that should just be empirically evaluated. To better use the machine learning algorithms, some more complex distance feature spaces not depending on these parameters could be useful.

The limitation of the $\mathcal{E}(\mathcal{XDG}_T, \mathcal{XDG}_H)$ measure relates to several aspects. One major problem is that similarity rules used in the measure may be incomplete to detect the entailment between $T$ and $H$. We will tackle this problem by using the notion of distance feature space. The point is to study sentences in $H$ not as a whole but as a collection of pieces (i.e., constituents, such as [the cat], [devours], and [the mouse]) and the distance feature space should represent how these pieces are covered by fragments of the text $T$. We will not use a bag-of-word space as it would suffer the previously discussed limits (Sec. 2).

To set up a distance feature space we have to solve a major problem. As we cannot rely on the bag-of-word model, each feature should represent the distance of a specific portion of the sentence $H$ with respect to the most similar portion in $T$. Such kind of distance feature space can be defined only if the structure of the sentence $H$ is stable and known in advance. We then focus our attention to the sentence structure `Subject-Verb-Object`. This structure can represent a fact that is the typical target of the textual entailment recognition task (see [7]) even if the verb has some more arguments to better define the fact. If the target structure of $H$ is defined, the distance feature space can be easily settled: every feature can represent the distance of each relevant element of $H$ (i.e. the *Subject*, the *Verb*, and the *Object*) to the most similar element in $T$. We will refer to $S$ as the *Subject*, to $V$ as the main *Verb*, and to $O$ as the *Object*. We will call $\mathcal{G}$ the basic distance feature space. The mnemonic feature names and the way to compute their values are then described in Fig. 2 where the functions $ss(c_h, c_t)$, $s(c_h, c_t)$, and $sm(c_h, c_t)$ are those defined in Sec. 3. Moreover, $c_h^S$, $c_h^V$, and $c_h^O$ are respectively the subject, the verb, and the object constituents of the hypothesis $H$. $c_t^S$, $c_t^V$, and $c_t^O$ are the constituents of the text $T$ most similar to respectively $c_h^S$, $c_h^V$, and $c_h^O$. This similarity is evaluated using $ss(c_h, c_t)$.

We also enriched the distance feature space with further information:

– a set of features $\mathcal{A}$ related to the percent of commonly anchored dependencies both in $H$ and in $T$, i.e.:

$$\mathcal{A} = \{\frac{|\cup_{c_h \in C_H} D_H(c_h)|}{|D_H|}, \frac{|\cup_{c_t \in C_T} D_T(c_t)|}{|D_T|}\}$$

| feature name | value |
|---|---|
| $S_{sm}$ | $sm_{np}(c_h^S, c_t^S)$ |
| $S_{simsub}$ | $\frac{\sum_{sh \in S(c_h^S)} \max_{s_t \in S(c_t^S)} s(s_h, s_t)}{\lvert S(c_h^S) \rvert}$ |
| $S_{ss}$ | $ss(c_h^S, c_t^S)$ |
| $V_{sm}$ | $sm_{vp}(c_h^V, c_t^V)$ |
| $V_{ss}$ | $ss(c_h^V, c_t^V)$ |
| $O_{sm}$ | $sm_{np}(c_h^O, c_t^O)$ |
| $O_{simsub}$ | $\frac{\sum_{sh \in S(c_h^O)} \max_{s_t \in S(c_t^O)} s(s_h, s_t)}{\lvert S(c_h^O) \rvert}$ |
| $O_{ss}$ | $ss(c_h^O, c_t^O)$ |

**Fig. 2.** The $\mathcal{G}$ distance feature space

– a set of features $\mathcal{T}$ related to the textual entailment subtasks (CD, MT, etc.).

Lastly, as simple feature spaces can work better than complex ones, we also used a less complex feature set $\mathcal{L}$. This should represent the distance at the lexical level in a bag-of-word fashion without any syntactic or semantic information. We then used two features: the percent of $H$ tokens and of $H$ lemmas that are in common with $T$. As we will see this is the baseline model.

The application of the machine learning algorithm to the distance feature space can be also seen as a method of empirically estimating the parameters $\alpha$, $\gamma$, and $\delta$ of the overall $\mathcal{E}(\mathcal{XDG}_T, \mathcal{XDG}_H)$ measure. However, what the algorithm should do is something more than using this distance feature space $\alpha$, $\gamma$, and $\delta$ are related to the specific bit of text, that is *Subject*, *Verb*, and *Object*.

## 5 Experimental Evaluation

The RTE challenge has been the first test to evaluate our approach and to verify its performances. The data set used for the competition was formed by three sets of entailment pairs: a *First development set*, composed by 287 annotated pairs, a *Second development set*, composed by 280 annotated pairs, and a *Test set*, composed by 800 non annotated pairs. Participating systems were evaluated over the test set: a prediction value ($True$ and $False$) and an associated degree of confidence on the prediction $c \in [0, 1]$ have been provided for each pair. Two measures were used for evaluation: *accuracy* (fraction of correct responses) and the *confidence-weighted score (cws)* as defined in [7].

We performed the experiments on our distance feature space using SVM-light [17]. The experiments have been organised as follows: firstly, we investigated the performances of the different feature spaces using examples in the two development sets and we choose the more promising feature space; secondly, we trained

the classifier on the chosen feature space using all the examples in the two development sets and we evaluated it on the *Test set*. Due to the maximal number of examples available for training, 567, the use of a distance feature space is perfectly justifiable. This number is enormously far from the examples methods like the ones based on the *Distributional Hypothesis* can have (see Sec. 2).

In order to better understand the effectiveness and the value of the SVM approach, we compare its performance with those obtained by a second *Rule-Based system* we presented at the RTE challenge. The Rule-Based approach simply applies the similarity measure $\mathcal{E}(\mathcal{XDG}_T, \mathcal{XDG}_H)$ on the entailment pairs. Parameters ($\alpha$, $\beta$, $\gamma$, and $\delta$) were manually tuned on the training set.

| | D1 | D3 | D4 | D5 | D6 |
|---|---|---|---|---|---|
| $\mathcal{L}$ | 51.16(±3.98) | - | - | - | - |
| $\mathcal{L}$,$\mathcal{T}$,$\mathcal{G}$ $\beta = 0.5$ | - | 55.28(±2.44) | 56.14(±2.51) | 56.40(±2.71) | 56.72(±2.92) |
| $\mathcal{L}$,$\mathcal{T}$,$\mathcal{G}$ $\beta = 1$ | - | 56.37(±2.45) | 57.14(±2.94) | 57.37(±3.45) | 57.12(±3.56) |
| $\mathcal{L}$,$\mathcal{T}$,$\mathcal{G}$,$\mathcal{A}$ $\beta = 1$ | - | - | 57.20(±3.01) | 57.42(±3.36) | 57.12(±3.38) |

**Table 1.** Preliminary accuracy analysis on the two development sets with SVM

The preliminary experiments have been carried out using the two development sets as source for an n-fold cross validation. We performed a 3-fold cross-validation repeated 10 times. Each number we report for these experiments is then given by 30 different runs. In Tab. 1, we report the mean accuracy and its standard deviation of SVM in the different feature spaces. Each row shows a different pair feature space and a value for the parameter $\beta$ when necessary: the baseline lexical distance feature space $\mathcal{L}$, the distance feature space $\mathcal{L}$,$\mathcal{T}$,$\mathcal{G}$ with $\beta = 0.5$ and $\beta = 1$, and, finally, the distance feature space with some more structural features $\mathcal{L}$,$\mathcal{T}$,$\mathcal{G}$,$\mathcal{A}$. The columns represent the degrees of the polynomial kernels used in SVM.

The results of the preliminary investigations suggest that every distance feature space is statistically significantly better than the lexical distance feature space $\mathcal{L}$. However a clear understanding of which feature space is better among all the others is not completely clear as they do not statistically differ. The higher mean is reached by the space $\mathcal{L}$,$\mathcal{T}$,$\mathcal{G}$,$\mathcal{A}$ with degree of the polynomial kernel equal to 5. Even if this is not statistically different from the other means that are around 57% of accuracy (all these performances can have represent the same statistical population), we choose this feature space to run the experiment on the competition.

The results over the competition *Test Set* are presented in Tab. 2. Results are divided in two tables: an overall analysis of the results and an analysis according to the different tasks. For comparison purposes, in Tab. 3 results of the Rule-Based system are also reported.

Not surprisingly, the overall results are only slightly above the chance threshold, in line with those obtained by other systems presented at RTE challenge. As

| *Measure* | *Result* |
|---|---|
| cws | 0.5591 |
| accuracy | 0.5182 |
| precision | 0.5532 |
| recall | 0.1950 |
| f | 0.2884 |

| TASK | *cws* | *accuracy* |
|---|---|---|
| CD | 0.7174 | 0.6443 |
| IE | 0.4632 | 0.4917 |
| MT | 0.4961 | 0.4790 |
| QA | 0.4571 | 0.4574 |
| RC | 0.5898 | 0.5214 |
| PP | 0.5768 | 0.5000 |
| IR | 0.4882 | 0.4889 |

**Table 2.** Competition results with SVM approach

| *Measure* | *Result* |
|---|---|
| cws | 0.5574 |
| accuracy | 0.5245 |
| precision | 0.5265 |
| recall | 0.4975 |
| f | 0.5116 |

| TASK | *cws* | *accuracy* |
|---|---|---|
| CD | 0.8381 | 0.7651 |
| IE | 0.4559 | 0.4667 |
| MT | 0.5914 | 0.5210 |
| QA | 0.4408 | 0.3953 |
| RC | 0.5167 | 0.4857 |
| PP | 0.5583 | 0.5400 |
| IR | 0.4405 | 0.4444 |

**Table 3.** Competition results with Rule-Based approach

stated in the introduction, Textual Entailment recognition is a fairly new task that encompasses many different NLP areas and issues (lexical semantics, syntactic and semantic analysis, etc.). Therefore, as every new demanding challenge in NLP, Textual Entailment needs to be investigated and handled carefully: the RTE challenge 2005 has been a first step in the study and the understanding of the linguistic phenomenon in its whole and most general definition. Indeed, the early experimental evidences obtained by the systems presented at the challenge are all still far from being satisfactory. Many interesting and very different approaches were presented, ranging from statistical methods to Rule-Based systems, operating at different level of analysis (lexical, syntactic, semantic, pragmatic). The variety of the approaches reveals both the intrinsically complex nature of the task and the early stage of analysis reached so far.

With regard to the results of our SVM system, two aspects are interesting to notice. Firstly, overall results are roughly in line with those obtained by the Rule-Based System in Tab. 3: the only surprising difference is in the level of recall, that is much higher for the Rule-Based. The very low level of recall achieved by SVM has to be further carefully analysed, in order to find a better trade-off between recall and precision. Moreover, the higher recall achieved by the Rule-Based is probably due also to the manual tuning process, that allowed a better set-up of the system.

Secondly, examining the results on the specific tasks, both the SVM and the Rule-Based approaches showed roughly the same performance on all tasks, apart

from CD. Indeed, on the *Comparable Document* (CD) task SVM achieved cws 0.7174, while Rule-Based 0.8381. These results are similar to those obtained by the other RTE systems. The CD task is thus in general easier than the others. In fact, most of the CD pairs in the corpus reveal the same linguistic behaviours: entailment is mainly characterized by simple lexical and syntactic variations that can be easily captured by syntactic rules and lexical-semantic analysis. Therefore, the use of simple word distance metrics between $T$ and $H$ together with a shallow syntactic and semantic analysis appears to be well suited for the CD task. Our system metrics were able to capture this kind of phenomenon adequately. In the easiest case CD pairs could be successfully recognized by simple lexical-semantic hints, as in:

$T$: [A Union Pacific freight train hit five people.]

$H$: [A Union Pacific freight train struck five people.]

or by syntactic normalization:

$T$: [Ghazi Yawar, a Sunni Muslim who lived for years in Saudi Arabia, has been picked as president of Iraq...]

$H$: [Yawer is a Sunni Muslim.]

Complex entailments, mixing syntactic and semantic variation were still captured by our system:

$T$: [Last July, a 12-year-old boy in Nagasaki - a city just north of Sasebo - was accused of kidnapping, molesting and killing a 4-year-old by shoving him off the roof of a car garage.]

$H$: [Last year a 12-year-old boy in Nagasaki was accused of murdering a four-year-old boy by pushing him off a roof.]

Notwithstanding, the CD dataset contains also cases of complex entailments that need at least logical reasoning to be correctly handled, as in:

$T$: [Each hour spent in a car was associated with a 6 percent increase in the likelihood of obesity and each half-mile walked per day reduced those odds by nearly 5 percent, the researchers found.]

$H$: [The more driving you do means you are going to weigh more – the more walking means you are going to weigh less.]

Such cases could not be grasped by our systems.

Besides CD, the partly disappointing performances on all other tasks are due to the more complex nature of the entailment, that often requires world knowledge and some kind of reasoning, such in:

$T$: [On Feb . 1 , 1945 , the Polish government made Warsaw its capital , and an office for urban reconstruction was set up ]

$H$: [Warsaw remained Poland's capital after the war .]

As a consequence of the little knowledge we still have of the linguistic phenomena underlying Textual Entailment, not only syntactic normalizations must be better studied, but also other NLP area should be investigated for hints and suggestions for solutions. Indeed, as underlined in the introduction, only 49% of entailment pairs can be captured by syntax. Other resources and reasoning tools should then be needed to cope with the problem: generic and verb lexical resources, world and domain knowledge, logical reasoning and many other issues should be better investigated.

## 6 Conclusions

Textual Entailment recognition is far from being a resolved problem and, as any other complex NLP problem, it may be possible to significantly improve results by applying machine learning techniques. In this paper we introduced the *distance feature space* that, in our opinion, could overcome the problem of a limited number of examples given for training. Results are far from being satisfactory but we believe that this is a promising way to use robust machine learning models in this very difficult problem.

## References


1. Chierchia, G., McConnell-Ginet, S.: Meaning and Grammar: An introduction to Semantics. MIT press, Cambridge, MA (2001)
2. Dagan, I., Glickman, O.: Probabilistic textual entailment: Generic applied modeling of language variability. In: Proceedings of the Workshop on Learning Methods for Text Understanding and Mining, Grenoble, France (2004)
3. Basili, R., Moschitti, A., Pazienza, M.T.: Empirical investigation of fast text categorization over linguistic features. In: Proceedings of the 15th European Conference on Artificial Intelligence (ECAI 2002), Lyon, France (2002)
4. Joachims, T.: Learning to Classify Text using Support Vector Machines: Methods, Theory, and Algorithms. Kluwer Academic Publishers (2002)
5. Glickman, O., Dagan, I.: A probabilistic setting and lexical coocurrence model for textual entailment. In: Proceedings of the ACL-Workshop on Empirical Modeling of Semantic Equivalence and Entailment, Ann Arbor, Michigan (2005)
6. Corley, C., Mihalcea, R.: Measuring the semantic similarity of texts. In: Proceedings of the ACL-Workshop on Empirical Modeling of Semantic Equivalence and Entailment, Ann Arbor, Michigan (2005)
7. Dagan, I., Glickman, O., Magnini, B.: The pascal recognising textual entailment challenge. In: PASCAL Challenges Workshop, Southampton, U.K (2005)
8. Miller, G.A.: WordNet: A lexical database for English. Communications of the ACM **38** (1995) 39–41
9. Resnik, P.: Using information content to evaluate semantic similarity. In: Proceedings of the 14th International Joint Conference on Artificial Intelligence, Montreal, Canada (1995)
10. Lin, D.: An information-theoretic definition of similarity. In: Proceedings of the 15th International Conference on Machine Learning, Madison, WI (1998)
11. Vanderwende, L., Coughlin, D., Dolan, B.: What syntax can contribute in entailment task. In: Proceedings of the 1st Pascal Challenge Workshop, Southampton, UK (2005)
12. Pazienza, M.T., Pennacchiotti, M., Zanzotto, F.M.: A linguistic inspection of textual entailment. In Bandini, S., Manzoni, S., eds.: AI*IA 2005: Advances in Artificial Intelligence. Volume LNAI 3673., Milan, Italy, Springer-Verlag (2005)
13. Raina, R., Haghighi, A., Cox, C., Finkel, J., Michels, J., Toutanova, K., MacCartney, B., de Marneffe, M.C., Manning, C.D., Ng, A.Y.: Robust textual inference using diverse knowledge sources. In: Proceedings of the 1st Pascal Challenge Workshop, Southampton, UK (2005)

14. Kouylekov, M., Magnini, B.: Tree edit distance for textual entailment. In: Proceedings of the International Conference Recent Advances of Natural Language Processing (RANLP-2005), Borovets, Bulgaria (2005)
15. Lin, D.: Dependency-based evaluation of minipar. In: Proceedings of the Workshop on Evaluation of Parsing Systems at LREC-98, Granada, Spain (1998)
16. Proceedings of the Seventh Message Understanding Conference (MUC-7), Virginia USA, Morgan Kaufmann (1998)
17. Joachims, T.: Making large-scale svm learning practical. In Schlkopf, B., Burges, C., Smola, A., eds.: Advances in Kernel Methods-Support Vector Learning, MIT Press (1999)
18. Collins, M., Duffy, N.: New ranking algorithms for parsing and tagging: Kernels over discrete structures, and the voted perceptron. In: Proceedings of the ACL-02, Philadelphia, PA (2002)
19. Moschitti, A.: A study on convolution kernels for shallow semantic parsing. In: Proceedings of the ACL-04, Barcellona, Spain (2004)
20. Lin, D., Pantel, P.: DIRT, discovery of inference rules from text. In: Knowledge Discovery and Data Mining. (2001) 323–328
21. Harris, Z.: Distributional structure. In Katz, J., ed.: The Philosophy of Linguistics, New York, Oxford University Press (1985)
22. Glickman, O., Dagan, I.: Identifying lexical paraphrases from a single corpus: A case study for verbs. In: Proceedings of the International Conference Recent Advances of Natural Language Processing (RANLP-2003), Borovets, Bulgaria (2003)
23. Shearer, K., Bunke, H., Venkatesh, S., Kieronska, D.: Efficient graph mathicng for video indexing. Technical Report 1997, Department of Computer Science, Curtin University (1997)
24. Cho, C., Kim, J.: Recognizing 3-d objects by forward checking constrained tree search. PRL **13** (1992) 587–597
25. Borner, K., Pippig, E., Tammer, E.C., Coulon, C.H.: Structural similarity and adaptation. In: EWCBR. (1996) 58–75
26. Sanders, K.E., Kettler, B.P., Hendler, J.: The case for graph-structured representations. In: Proceedings of the Second International Conference on Case-based Reasoning, Springer-Verlag (1997) 245–254
27. Bunke, H.: Graph matching: Theoretical foundations, algorithms, and applications. In: Vision Interface 2000, Montreal, Springer-Verlag (2000) 82–88
28. Bunke, H., Shearer, K.: A graph distance metric based on the maximal common subgraph. Pattern Recogn. Lett. **19** (1998) 255–259
29. Basili, R., Zanzotto, F.M.: Parsing engineering and empirical robustness. Natural Language Engineering **8/2-3** (2002)
30. Pazienza, M.T., Pennacchiotti, M., Zanzotto, F.M.: Identifying relational concept lexicalisations by using general linguistic knowledge. In: ECAI. (2004) 1071–1072
31. Wu, D.: Stochastic inversion transduction grammars, with application to segmentation, bracketing, and alignment of parallel corpora. In: Computational Linguistics. Volume 23. (1997) 207–223