Similarity Is *Not* Entailment — Jointly Learning Similarity Transformations for Textual Entailment

Ken-ichi Yokote
velleykt@iis.u-tokyo.ac.jp

Danushka Bollegala
danushka@iba.t.u-tokyo.ac.jp

Mitsuru Ishizuka
ishizuka@i.u-tokyo.ac.jp

The University of Tokyo
7-3-1, Hongo, Bunkyo-ku, Tokyo, 113-8656, Japan.

Abstract

Predicting entailment between two given texts is an important task upon which the performance of numerous NLP tasks depend on such as question answering, text summarization, and information extraction. The degree to which two texts are similar has been used extensively as a key feature in much previous work in predicting entailment. However, using similarity scores directly, without proper transformations, results in suboptimal performance. Given a set of lexical similarity measures, we propose a method that jointly learns both (a) a set of non-linear transformation functions for those similarity measures and, (b) the optimal non-linear combination of those transformation functions to predict textual entailment. Our method consistently outperforms numerous baselines, reporting a micro-averaged F-score of 46.48 on the RTE-7 benchmark dataset. The proposed method is ranked 2-nd among 33 entailment systems participated in RTE-7, demonstrating its competitiveness over numerous other entailment approaches. Although our method is statistically comparable to the current state-of-the-art, we require less external knowledge resources.

Introduction

Inferring entailment relations between natural language texts is a difficult but an important problem (Dagan and Glickman 2004). Given a text T, the objective of textual entailment is to check whether a hypothesis H can be inferred from T. If H can be inferred from T, then we say T *entails* H. For example, consider the two sentences shown below.

(1) **T:** *All animals must eat to live.*

(2) **H:** *All wild animals must eat to live.*

The second sentence can be inferred from the information provided in the first because *animals* is a superset of *wild animals*. Consequently, sentence (1) entails sentence (2).

The ability to recognize textual entailment is a fundamental requirement in many natural language processing tasks such as question answering, text summarization, and information extraction. For example, if we can infer that *"alcohol affects blood pressure"* from the fact that *alcohol reduces blood pressure"*, then we can use this result to answer the question, *"What affects blood pressure?"*. Indeed, recognizing entailment bears similarities to Turing's famous test to assess whether machines can think, as access to different sources of knowledge and the ability to draw inferences seem to be among the primary ingredients for an intelligent system (Bos and Markert 2005).

Detecting entailment relations between two texts often requires knowledge that is not explicitly encoded in the two texts (LoBue and Yates 2011). In sentences (1)-(2), the knowledge that *wild animals* are a subset of *animals* is an important piece of knowledge that an entailment system must have to accurately predict the entailment. Existing approaches to textual entailment use numerous knowledge resources such as the WordNet (Tatu and Moldovan 2005), the FrameNet (Aharon, Szpektor, and Dagan 2010), and the Web (Glickman, Dagan, and Koppel 2005) and measure the degree to which T is similar to H. A high degree of similarity between T and H sometimes indicates an entailment relation between T and H. For example, sentences (1)-(2) share many words in common. Moreover, using an ontology such as the WordNet, we can compute the taxonomic similarity between *animals* and *wild animals*. Consequently, a high degree of similarity can be observed between (1) and (2), implying that (1) entails (2).

However, a high degree of similarity between two texts *does not* always guarantee an entailment. For example, consider sentences (3)-(4) selected from the RTE-7 dataset.

(3) **T:** *Fannie Mae's accounting has been under investigation by the Justice Department and the SEC, and it has become the subject of investor lawsuits.*

(4) **H:** *Fannie Mae is a big company.*

Here, the information contained in (3) is insufficient to determine the size of the company. Consequently, in the RTE-7 dataset (3) does not entail (4). However, the words *Fannie Mae* are common to both (3) and (4), and the word *company* in (4) is similar to the words *department*, *investor*, *lawsuit*, and *accounting* in (3). Consequently, popular lexical similarity measures such as edge-counting in the WordNet (Leacock and Chodorow 1998) (similarity score = 0.39) and depth of the lowest common subsumer in the WordNet (Wu and Palmer 1994) (similarity score = 0.33) return high similarity scores between (3) and (4). This example shows that high similarity scores do not necessarily guarantee an entailment, thus similarity scores alone must not be used for predicting textual entailment.

Given a set of pairwise lexical similarity measures, we propose a method to learn non-linear transformation functions for those measures to accurately predict textual entailment between two texts. Specifically, we represent a sentence pair, (T, H), using a feature vector in which, each feature is computed using a different pairwise lexical similarity measure, transformed by a similarity transformation function. We do not assume any specific properties of the similarity measures. Therefore, the proposed method can be used with a wide-range of similarity measures. Our feature representation considers not only the information contained in T and H, but also use the original documents from which T and H are extracted as an additional background knowledge source. Moreover, our feature representation has the desirable property that the dimensionality of the feature vectors is independent of the length of T or H, and is determined only by the number of similarity measures used. We propose a supervised method to jointly learn both the similarity transformation functions, as well as optimal combination of features using a set of labeled sentence pairs.

Experimental results on the RTE-7 dataset show that the use of similarity transformation functions consistently improves the performance over using raw similarity scores. Our method achieves a micro-averaged F-score of 46.48, which is ranked 2-nd among 33 different textual entailment systems, indicating the competitiveness of the proposed method. Moreover, our results are statistically comparable to the current state-of-the-art, although we require less external knowledge sources.

From Similarity to Entailment

Representing Sentence Pairs

Let us assume that we are required to determine whether T entails H, for two sentences T and H. First, we represent the sentence pair (T, H) using a feature vector. For this purpose, we use a set, $\mathcal{S} = \{s_1, \ldots, s_L\}$, of pairwise lexical similarity measures, s_i. We do not assume any specific properties of individual similarity measures, except that their similarity scores are in the range $[0, 1]$. Let us denote the sets of words contained in H and T respectively by $\mathcal{H} = \{h_1, \ldots, h_N\}$ and $\mathcal{T} = \{t_1, \ldots, t_M\}$. Here, we use a bag-of-words boolean representation that ignores multiple occurrences of a word in a sentence. Next, we create a word-similarity matrix, **A**, as shown in Figure 1, in which we arrange similarity measures in rows and the words in $\mathcal{H}$ followed by the words in $\mathcal{T}$ in columns. The (i, j) element of **A** is denoted by $\mathbf{A}_{(i,j)}$, and is computed as follows:

$$\mathbf{A}_{(i,j)} = \begin{cases} \max\limits_{k=1,\ldots,M} s_i(h_j, t_k) & \forall j = 1, \ldots, N \\ \max\limits_{k=1,\ldots,N} s_i(h_k, t_{j-N}) & \forall j = N+1, \ldots, N+M \end{cases} \quad (1)$$

where, in each row i we compute the maximum similarity between a particular word in $\mathcal{H}$ and all the words in $\mathcal{T}$ using the similarity measure s_i to compute the elements corresponding to the words that appear in $\mathcal{H}$. Similarly, each word in $\mathcal{T}$ is compared against all the words in $\mathcal{H}$ using s_i to compute the elements in the i-th row corresponding to the words in $\mathcal{T}$. Note that a word in $\mathcal{H}$ needs not be similar to all the words in $\mathcal{T}$. For example, in sentences (1)-(2) in our first example, the *animals* in T needs to be similar only to *wild animals* in H, to infer entailment between T and H.

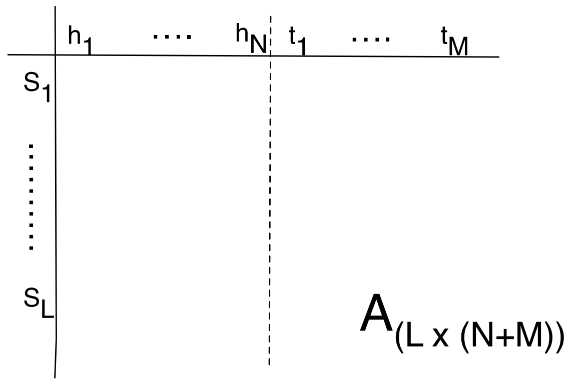

Figure 1: Constructing a word-similarity matrix **A** using a set of similarity measures $\mathcal{S}$ for sentences T and H.

The word-similarity matrix **A** is agnostic to the salience of the words in H and T. For example, it would consider both the words *animals* and *eat* in (1)-(2) with equal importance when computing the similarity scores. However, the word *animals* is more important when inferring the entailment between (1) and (2) than the word *eat*. Determining the salience of words with regard to their contribution towards overall entailment is a challenging task that requires additional background knowledge (LoBue and Yates 2011). The subset of words in $\mathcal{H}$ that is important when inferring an entailment between H and T is different from that when inferring an entailment between H and another sentence T'. For example, consider sentence (5).

(5) **T':** *All wild animals must eat what ever food they can find to sustain their lives in the jungle.*

Here, the word *live* in H becomes more important than the phrase *wild animals* when inferring an entailment between H and T' because, *live* can be inferred from the phrase *to sustain their lives* in T'. To incorporate this notion of salience of words in to our feature representation, we construct an $(N + M)$-dimensional *salience vector*, $\boldsymbol{v}$, in which the i-th element denotes the importance of the word h_i (for $i = 1, \ldots, N$), or t_i (for $i = (N + 1), \ldots, (N + M)$). For simplicity, we use the inverse document frequency (IDF) (Salton and Buckley 1983) for words h_i and t_i as their salience scores. We use the original documents provided in the RTE-7 dataset from which a particular sentence has been selected, for computing IDF. Because documents are organized by topics in RTE-7, this enables us to compute salience scores that are relevant to the topic which T and H belong.

We multiply the salience vector, $\boldsymbol{v}$, with the word-similarity matrix, **A**, to compute the feature vector, $\boldsymbol{\phi}(T, H)$, corresponding to the sentence pair, (T, H), as follows:

$$\boldsymbol{\phi}(T, H) = \mathbf{A}\boldsymbol{v}. \quad (2)$$

The dimensionality of $\boldsymbol{\phi}(T, H)$ is equal to the number of similarity measures used (i.e. L), and is independent of the number of words that appear in $\mathcal{T}$ or $\mathcal{H}$. This property of the feature representation is important for the joint learning procedure for similarity transformation functions discussed later in the paper. In particular, the i-th element, $[\boldsymbol{\phi}(T, H)]_{(i)}$, in the feature vector $\boldsymbol{\phi}(T, H)$ involves only similarity scores produced by the i-th similarity measure, s_i, and is given by the sum of two terms,

$$[\boldsymbol{\phi}(T,H)]_{(i)} = \sum_{j=1}^{N} \max_{k=1,\ldots,M} s_i(h_j, t_k)\mathrm{IDF}(h_j) + \sum_{j=N+1}^{N+M} \max_{k=1,\ldots,N} s_i(h_k, t_{j-N})\mathrm{IDF}(t_{j-N}). \quad (3)$$

Here, $\mathrm{IDF}(x)$ denotes the inverse document frequency of a word x.

Although we limit our discussion in this paper to pairwise lexical similarity measures, our feature vector representation can be easily extended to include string-based (Malakasiotis and Androutsopoulos 2007; Zanzotto and Moschitti 2006) and syntactic tree-based (Mehdad, Moschitti, and Zanzotto 2010; Wang and Neumann 2007b) similarity functions, which have been shown to be useful for predicting textual entailment.

Similarity Transformation Functions

As we already illustrated by an example (sentences (3)-(4)), a high degree of lexical similarity does not always imply an entailment between two sentences. The level to which a particular similarity score contributes towards the final entailment decision depends both on the similarity measure as well as its score. To convert the similarity scores produced by a particular similarity measure into confidence scores that can be used to predict the entailment of H by T, we propose *similarity transformation functions*. A similarity transformation function is defined as a mapping between the similarity score $s_i(h, t)$ produced by a particular similarity measure s_i for two words $h \in \mathcal{H}$, $t \in \mathcal{T}$, and a confidence score that indicates whether T would entail H. Any function that can map the range $[0, 1]$ of similarity scores to the range $[0, 1]$ of confidence scores can be used for this purpose. In this paper, we use *logistic sigmoid* as our preferred transformation function. Logistic sigmoid functions are often used as activation functions in neural networks (Bishop 2006).

Specifically, we define the similarity transformation function, σ_i, corresponding to the similarity measure s_i as follows:

$$\sigma_i(s_i(h,t), \boldsymbol{\alpha}_i) = \frac{\alpha_{i1}}{1 + \exp(-\lambda(s_i(h,t) - \alpha_{i2}))}. \quad (4)$$

Here, $\boldsymbol{\alpha}_i = (\alpha_{i1}, \alpha_{i2})^\top$ is a positive-valued parameter vector that determines the step (α_{i1}) and the cut-off (α_{i2}) of the logistic sigmoid function. The scaling factor λ determines the sharpness of the step, and is set common to all similarity transformation functions using development data as we describe later. The parameter vector $\boldsymbol{\alpha}_i$ is learnt using labeled training data by the joint learning procedure described in the next Section. Figure 2 illustrates transformation functions learnt by our method for several similarity measures.

Joint Learning of Transformations and Entailment

Following previous work on recognizing textual entailment, we model this problem as a binary classification task in which, given two sentences T and H, we must decide whether T entails H. Let us consider a labeled dataset $\mathcal{D} = \{((T_q, H_q), y_q)\}_{q=1}^{Q}$, of Q instances, in which $y_q \in \{-1, 1\}$ indicates whether T_q entails H_q. If T_q entails H_q, then $y_q = 1$, otherwise $y_q = -1$.

There are two important factors we must consider when learning to recognize textual entailment. First, the optimal form of each similarity transformation function remains unknown. It is natural to expect some level of a correlation between different similarity measures. For example, similarity measures that use WordNet share the same WordNet taxonomy. Therefore, when determining the optimal values of parameter vectors $\boldsymbol{\alpha}_i$ we must consider all similarity transformation functions simultaneously. Second, the optimal combination of features given by Equation 5 that maximizes the prediction accuracy of entailment is unknown. Different features are useful to different degrees for predicting entailment. We propose a method to *jointly* learn both (a) similarity transformation functions, and (b) the optimal combination of the features, that maximizes the accuracy of the entailment prediction task. The joint learning approach enables us to consider the overall contribution of both similarity transformation functions and the feature combination towards our ultimate goal of predicting textual entailment.

Let us denote a feature vector that results after performing similarity transformations by $\boldsymbol{\phi\prime}(T, H)$. Note that this vector is given by Equation 2, in which we use the word-similarity matrix, $\mathbf{A}$, after performing the corresponding transformations σ_i on each similarity measure s_i. Specifically, the i-th element of $\boldsymbol{\phi\prime}(T, H)$ is given by,

$$[\boldsymbol{\phi\prime}(T,H)]_{(i)} = \sum_{j=1}^{N} \max_{k=1,\ldots,M} \sigma_i(s_i(h_j, t_k), \boldsymbol{\alpha}_i)\mathrm{IDF}(h_j) + \sum_{j=N+1}^{N+M} \max_{k=1,\ldots,N} \sigma_i(s_i(h_k, t_{j-N}), \boldsymbol{\alpha}_i)\mathrm{IDF}(t_{j-N}). \quad (5)$$

Let us consider the linear combination of those transformed features given by an L-dimensional weight vector $\boldsymbol{w}$. Using a large margin binary classification approach, similar to that used in Support Vector Machines (Vapnik 1998), we derive the following optimization problem to minimize the hinge loss.

$$\min_{\boldsymbol{w}, \boldsymbol{\alpha}_i} \boldsymbol{w}^\top \boldsymbol{w} + \sum_{i=1}^{L} \mu_i \boldsymbol{\alpha}_i{}^\top \boldsymbol{\alpha}_i + \nu \sum_{q=1}^{Q} \xi_q \quad (6)$$

$$\text{s.t.} \quad y_q \boldsymbol{w}^\top \boldsymbol{\phi\prime}(T_q, H_q) \geq 1 - \xi_q \quad \forall_{q=1,\ldots,Q} \quad (7)$$

$$\xi_q \geq 0 \quad \forall_{q=1,\ldots,Q} \quad (8)$$

The objective function (Equation 6) that we aim to minimize w.r.t. the weight vector $\boldsymbol{w}$ and the parameter vectors $\boldsymbol{\alpha}_i$ consists of three terms: the $L2$ norm of the weight vector $\boldsymbol{w}$, an $L2$ regularizer for each parameter vector with regularization coefficient μ_i, and the slack penalty ξ_q for each margin violation. Note that $\xi_q = 0$, if (T_q, H_q) is correctly classified (i.e. satisfies Inequality 7). The cost parameter, ν, determines the trade-off between using a soft margin vs. margin maximization. To simplify the optimization problem, we set all μ_is equal to a common regularization coefficient, μ. Both μ and ν are hyperparameters in our model that are set using development data as described later in our Experiments.

Although the similarity transformation functions σ_i defined in Equation 4 (logistic sigmoids) are convex w.r.t. $\boldsymbol{\alpha}_i$, and each element in $\boldsymbol{\phi}\prime(T, H)$ can be written as a positively weighted sum of those similarity transformation functions, inequality constraint 7 can become non-convex if the weight vector $\boldsymbol{w}$ contains negative values. Consequently, we use Sequential Convex Programming (SCP) (Zillober 1993; Boyd and Vandenberghe 2004) to solve the above-mentioned optimization problem by using a trust region approximation to the non-convex inequality constraint 7. Moreover, by considering the dual form of the Lagrangian, we derive a kernalized version of the optimization problem, which enables us to study non-linear combinations of features as shown later in our experiments.

Dataset and Evaluation Measures

We used the dataset created for the Seventh Recognizing Textual Entailment (RTE-7) (Bentivogil et al. 2011) challenge in our experiments. RTE-7 is the most recently organized workshop in the RTE series, hence by using this dataset we can compare our method against the latest state-of-the-art results. In RTE-7 dataset, for each sentence H, a maximum of 100 candidate sentences T are listed. Moreover, a set of documents related to a particular topic is provided for each H that can be used as additional background knowledge for detecting entailment. However, this background knowledge alone, without T, does not entail H. We used the official train and test splits in the RTE-7 dataset in our experiments. The train portion of the RTE-7 dataset contains 284 Hs, and for each H a maximum of 100 candidate Ts are provided. In total, there are $21,420$ sentence annotations in the train portion of the RTE-7 dataset out of which 1136 are positive (entailment) judgements. The test portion contains 269 Hs and a maximum 100 candidate Ts for each H, amounting to $22,426$ total sentence annotations out of which, 1308 are positive instances. Following the official RTE-7 guidelines, we report micro-averaged F-scores in our experiments.

Experiments and Results

As a preprocessing step, we remove stop words and perform word lemmatization using NLTK[1] for all Ts, Hs, and background documents in the RTE dataset. We use the training data provided in RTE-7 to train the proposed method. Specifically, we combine Ts listed for an H to construct (T, H) pairs, and label those as positive (T entails H) or negative (T does not entail H) using the annotations provided in the RTE-7 dataset. Next, we construct the word-similarity matrix **A** for each (T, H) pair using six pairwise lexical similarity measures described next that have been used frequently in previous work in textual entailment.

Path: The **Path** measure computes the shortest path between two words in the WordNet taxonomy and returns the reciprocal of the number of edges along this shortest path as the similarity between the two words.

WP: To measure the similarity between two words Wu and Palmer (1994) proposed a measure that considers the depth in a taxonomy of the two words and their maximally specific superclass.

Res: The more information two concepts share, the more similar they are. Resnik (1995) computes the similarity between two words as the maximum information content of all words that subsume those two words in a taxonomy. He computes the information content of a word from its frequency in a large corpus.

Lin: This measures computes the similarity between two words using the Similarity Theorem proposed in (Lin 1998). Specifically, the similarity between two words is computed as twice the information content of the largest common subsumer of those two words, divided by the sum of the information contents of each word.

JC: Jiang and Conrath (1997) combined both edge-counts as well as information content values using the WordNet 'is-a' hierarchy to compute the similarity between two words.

LC: This is the semantic similarity measure proposed in (Leacock and Chodorow 1998). It counts up the number of edges between the word senses in the 'is-a' hierarchy of the WordNet, and scales this value by the maximum depth of the WordNet's 'is-a' hierarchy.

We used the implementations of those similarity measures provided in (Pedersen 2004). We hold-out a randomly selected set of 100 (T, H) pairs from the RTE-7 training data as development data. We tune λ (set to 10000), μ (set to 0.5), ν (set to 1) and kernel parameters such that we obtain the highest micro-averaged F-scores on this development data. All evaluations are conducted with those parameter values on the official RTE-7 test set using micro-averaged F-score as the evaluation measure.

In Table 1, we measure the performance under four settings as described next.

Raw Similarity Measures: We mark a (T, H) pair as positive if $[\boldsymbol{\phi}(T, H)]_i \geq \theta_i$, for each individual similarity measure s_i. The threshold θ_i is tuned such that the highest micro-averaged F-score is obtained on the training dataset. This baseline method demonstrates the level of performance that we would obtain if we use raw similarity scores, without any transformations. The first six rows in Table 1 shows the performance of this baseline approach for each similarity measure.

Learning the Transformations only: We use the proposed method to learn only the transformation functions for each similarity measure, without combining the features.

[1]`http://www.nltk.org/`

Table 1: Performance of similarity transformations and joint learning of feature weights. Micro-averaged F-scores are shown for different methods when $\mathcal{H}$, $\mathcal{T}$ and both $\mathcal{H}$ and $\mathcal{T}$ are used to create the word-similarity matrix $\mathbf{A}$.

Methods	$\mathcal{H}$ only	$\mathcal{T}$ only	$\mathcal{H}+\mathcal{T}$
Path	32.36	12.78	16.21
WP	21.67	11.41	12.65
Res	36.76	14.66	20.46
Lin	20.75	14.69	20.70
JC	**36.91**	13.85	20.82
LC	20.75	11.28	12.18
Path+Trans	44.72	26.92	35.23
WP+Trans	36.85	17.79	24.96
Res+Trans	44.86	27.83	36.10
Lin+Trans	**45.28**	28.18	36.41
JC+Trans	45.06	28.17	36.41
LC+Trans	44.88	35.83	28.00
Comb (Linear)	44.96	28.79	36.91
Comb (Quadratic)	46.06	28.15	34.34
Comb (Cubic)	45.93	24.81	32.94
Comb (RBF)	**46.19**	29.16	33.76
Comb (Sigmoid)	45.42	20.84	36.15
Trans + Comb (Linear)	45.68	28.72	36.54
Trans + Comb (Quadratic)	46.19	28.23	35.27
Trans + Comb (Cubic)	45.94	25.76	33.74
Trans + Comb (RBF)	46.35	28.53	34.73
Trans + Comb (Sigmoid)	**46.48**	27.40	37.28

Specifically, $\mathbf{A}$ is created using only s_i and the optimization problem 6 is solved for $\boldsymbol{\alpha}_i$ by fixing $w_i = 1$. This baseline shows the effect of transforming similarity scores as opposed to using raw similarities. In Table 1, we denote this approach by **+Trans** for each similarity measure.

Learning the Combinations without Transformations: To study the effect of integrating different raw similarity scores, we use the feature vectors $\boldsymbol{\phi}(T, H)$ with the proposed method. Optimization problem 6 reduces to the typical SVM learning under this setting. Results using different kernel functions (within brackets) are shown in Table 1 (denoted by **Comb**).

Joint Learning of Combinations and Transformations: This is the full method proposed in this paper, in which we jointly learn both similarity transformations and the optimal feature combinations. In Table 1, **Trans + Comb** corresponds to the results obtained via joint learning for different kernels (shown within brackets).

To study the effect of using information in H vs. T towards entailment detection, we construct three versions of the matrix $\mathbf{A}$ for each case in Table 1: (a) using only the words in $\mathcal{H}$ ($\mathbf{A} \in \mathbb{R}^{L\times N}$), (b) using only the words in $\mathcal{T}$ ($\mathbf{A} \in \mathbb{R}^{L\times M}$), and (c) both ($\mathbf{A} \in \mathbb{R}^{L\times(N+M)}$).

For each method compared in Table 1, we see that the performance reported by $\mathcal{H}$ only is greater than that of $\mathcal{T}$ only. This result shows that the information in H is more important than the information in T when deciding whether T entails H. Although $\mathcal{H}+\mathcal{T}$ improves the performance over $\mathcal{T}$ only for each method, it is still lower than the corresponding $\mathcal{H}$ only result. This is because when we construct the columns of $\mathbf{A}$ using words in both $\mathcal{H}$ and $\mathcal{T}$, we lose the direction of the entailment. Consequently, we focus our attention to $\mathcal{H}$ only results for the remainder of this paper.

Table 2: Performance comparison on RTE-7 dataset using micro-averaged F-score.

Method	F-score	Method	F-score
IKOMA	48.00	FBK	41.90
Proposed	46.48	TE-IITB	30.78
U-TOKYO	45.15	JU-CSE	30.47
BUPT	44.99	ICL	29.73
CELI	44.10	UAIC	27.85
DFKI	43.41	SJTU	23.31
BIU	42.34	SINAI	14.72

Table 1 shows that the transformed (**+Trans**) version of a similarity measure always improves over its non-transformed (raw) counterpart, emphasizing the importance of similarity transformations proposed in the paper. In particular, the performance of **Lin + Trans** (45.28) as a standalone similarity measure is remarkable. All the combinations of raw similarity measures (**Comb**) learnt using different kernels functions, outperform the performance of any one of the constituent raw similarity measures. However, we obtain the best results for this task by jointly learning both transformations as well as feature combinations (**Trans + Comb**) using our proposed method. The improvements shown by the joint learning is consistent across a wide range of kernel functions. In particular, the highest micro-averaged F-score (46.48) is reported by the sigmoid kernel. We believe that the superior performance shown by the sigmoid kernel in the joint learning setting can be attributable to the fact that our transformation functions themselves are logistic sigmoids. Figure 2 shows the transformation functions learnt with the sigmoid kernel for each similarity measure. From Figure 2, we see that the step (α_{i1}) and the cut-off (α_{i2}) learnt using the proposed method differ considerably for each similarity measure. This result supports our proposal to learn different transformations for different similarity measures.

In Table 2, we compare our method against all the systems that participated in RTE-7 using micro-averaged F-scores. We do not implement those methods by ourselves, and report the official results published. 13 teams submitted 33 systems to RTE-7. Due to the limited availability of space, we show only the results for the best system submitted by each team in Table 2. For further details of each individual system refer the (Bentivogil et al. 2011). Our method is ranked 2-nd among all 33 systems submitted to RTE-7. Moreover, the difference in performance between our method and IKOMA, the best system in RTE-7, is not statistically significant according to an ANalysis of Variations (ANOVA) performed with a post Tukey Honest Significant Differences (HSD) test. However, unlike the proposed method, which only uses WordNet, IKOMA is a knowledge intensive method that requires numerous external resources including WordNet, Cat-

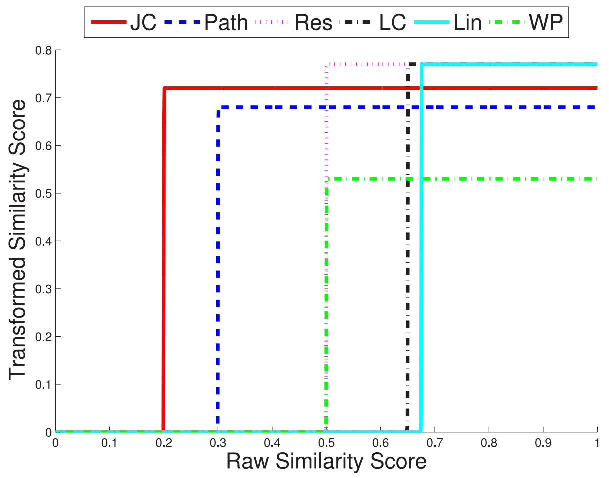

Figure 2: Similarity transformation functions learnt by the proposed method with the sigmoid kernel.

Var (Habash and Dorr 2003), and an acronym list.

Related Work

Textual entailment recognition arises in numerous natural language processing tasks such as question answering, information retrieval, and text summarization. Consequently, numerous workshops that specifically study this problem have been organized (Callison-Burch et al. 2009; Sekine et al. 2009). The Recognizing Textual Entailment (RTE) challenges (Bentivogil et al. 2011; Dagan and Glickman 2004), is a notable venue for the research in textual entailment. Next, we discuss the major approaches proposed for this problem. For a detailed survey refer (Androutsopoulos and Malakasiotis 2010).

If T and H contain some word sequences in common, then it might be a clue that T entails H. Following this intuition, several entailment recognition methods that compare T and H using surface string similarity measures have been proposed (Burchardt et al. 2009; Malakasiotis and Androutsopoulos 2007). These methods depend on the accuracy of finding a correct alignment of words between T and H. Although accurate word alignment methods have been proposed for statistical machine translation, they often perform poorly on textual entailment tasks, because T and H are often of very different lengths, they do not necessarily convey the same information, and textual entailment training datasets are much smaller than those used in statistical machine translation (MacCartney, Galley, and Manning 2008).

As an alternative to comparing T and H using their surface strings, entailment recognition methods that compare the dependency trees of T and H have been proposed (Iftene and Balahur-Dobrescu 2007; Zanzotto, Pennacchiotti, and Moschitti 2009; Wang and Neumann 2007a; 2007b). For example, if H's parse tree is highly similar to one or more of the subtrees in T's parse tree, then this might indicate an entailment between T and H. Because dependency trees are less sensitive to the relative word ordering in sentences, they can overcome some of the disfluencies associated with the surface string matching approaches discussed above. However, in practice dependency tree-based approaches do not necessarily outperform surface string-based methods in entailment detection tasks because of parse errors.

Bos and Markert (2005) model the problem of recognizing textual entailment as a logical inference problem. They use model building, a technique popularly used in automatic reasoning, and integrate it with a shallow word overlap measure. Although their method reports high precision scores on the RTE test set, it suffers in terms of recall due to a general lack of appropriate background knowledge. Learning entailment rules from natural language texts is a promising research direction to extract the necessary background knowledge for the inference process (Haghighi, Ng, and Manning 2005; Berant, Dagan, and Goldberger 2010).

Textual entailment methods that model both T and H as bags-of-words and use numerous semantic similarity measures such as WordNet-based similarity measures and distributional similarity measures, have been proposed (Zanzotto and Moschitti 2006; Geffet and Dagan 2004; Mirkin, Dagan, and Geffet 2006; Geffet and Dagan 2005). By using similarity measures that use a taxonomy such as the WordNet, it is possible to perform a soft matching between T and H using synonyms, hypernyms and meronyms. However, as we showed experimentally, using raw similarity scores alone results in suboptimal performance, and often requires some form of a transformation beforehand. Moreover, binary classifiers such as Support Vector Machines (Vapnik 1998) with customized kernels (Zanzotto and Dell'Arciprete 2009), have been used to combine different similarity measures to predict entailment or not-entailment for a given pair of sentences (T, H).

Our proposed method is different from all the above-mentioned approaches in that we simultaneously learn both a set of transformation functions as well as the optimal combination of features for predicting textual entailment. As shown experimentally on the RTE-7 evaluation dataset, both similarity transformations and joint learning of the optimal feature combination improve the performance in textual entailment recognition, achieving results comparable to that of the current state-of-the-art for this task.

Conclusion

We proposed a supervised method to jointly learn both similarity transformation functions and the optimal feature combination for textual entailment using a labeled set of sentence pairs. Our experimental results show that both similarity transformation and feature combination consistently improves numerous similarity measures, achieving results comparable to the current state-of-the-art. In future, we plan to incorporate surface string-based and dependency tree-based similarity measures into our method to further improve the textual entailment prediction.

References

Aharon, R. B.; Szpektor, I.; and Dagan, I. 2010. Generating entailment rules from framenet. In *ACL'10*, 241 – 246.

Androutsopoulos, I., and Malakasiotis, P. 2010. A survey of paraphrasing and textual entailment methods. *Journal of Artificial Intelligence Research* 38:135 – 187.

Bentivogil, L.; Clark, P.; Dagan, I.; Dang, H. T.; and Giampiccolo, D. 2011. The seventh pascal recognizing textual entailment challenge. In *RTE-7*.

Berant, J.; Dagan, I.; and Goldberger, J. 2010. Global learning of focused entailment graphs. In *ACL'10*, 1220 – 1229.

Bishop, C. M. 2006. *Pattern Recognition and Machine Learning*. Springer.

Bos, J., and Markert, K. 2005. Recognizing textual entailment with logical inference. In *EMNLP'05*, 628 – 635.

Boyd, S., and Vandenberghe, L. 2004. *Convex Optimization*. Cambridge University Press.

Burchardt, A.; Pennacchiotti, M.; Thater, S.; and Pinkel, M. 2009. Assessing the impact of frame semantics on textual entailment. *Natural Language Engineering* 15(4):527 – 550.

Callison-Burch, C.; Dagan, I.; Manning, C. D.; Pennacchiotti, M.; and Zanzotto, F. M. 2009. Textinfer'09. In *ACL-IJCNLP'09 Workshop on Applied Textual Inference*.

Dagan, I., and Glickman, O. 2004. Probabilistic textual entailment: Generic applied modeling of language variability. In *PASCAL Workshop on Text Understanding and Mining*.

Geffet, M., and Dagan, I. 2004. Feature vector quality and distributional similarity. In *COLING'04*, 247 – 253.

Geffet, M., and Dagan, I. 2005. The distributional inclusion hypothesis and lexical entailment. In *ACL'05*, 107 – 114.

Glickman, O.; Dagan, I.; and Koppel, M. 2005. Web based probabilistic textual entailment. In *1st RTE Workshop*.

Habash, N., and Dorr, B. 2003. A categorical variation database for english. In *HLT-NAACL'03*, 17 – 23.

Haghighi, A.; Ng, A. Y.; and Manning, C. D. 2005. Robust textual inference via graph matching. In *HLT/EMNLP'05*, 387 – 394.

Iftene, A., and Balahur-Dobrescu, A. 2007. Hypothesis transformation and semantic variability rules used in recognizing textual entailment. In *ACL-PASCAL Workshop on Textual Entailment and Paraphrasing*.

Jiang, J. J., and Conrath, D. W. 1997. Semantic similarity based on corpus statistics and lexical taxonomy. In *10th Intl. Conf. Research on Computational Linguistics (ROCLING)*.

Leacock, C., and Chodorow, M. 1998. Combining local context and wordnet similarity for word sense disambiguation. *WordNet: An Electronic Lexical Database* 49:265–283.

Lin, D. 1998. An information-theoretic definition of similarity. In *ICML'98*, 296 – 304.

LoBue, P., and Yates, A. 2011. Types of common-sense knowledge needed for recognizing textual entailment. In *ACL'11*, 329 – 334.

MacCartney, B.; Galley, M.; and Manning, C. D. 2008. A phrase-based alignment model for natural language inference. In *EMNLP'08*, 802 – 811.

Malakasiotis, P., and Androutsopoulos, I. 2007. Learning textual entailment using svms and string similarity measures. In *ACL'07 Workshop on Textual Entailment and Paraphrasing*, 42 – 47.

Mehdad, Y.; Moschitti, A.; and Zanzotto, F. M. 2010. Syntactic/semantic structures for textual entailment recognition. In *HLT-NAACL'10*, 1020 – 1028.

Mirkin, S.; Dagan, I.; and Geffet, M. 2006. Integrating pattern-based and distributional similarity methods for lexical entailment acqusition. In *COLING/ACL'06*, 579 – 586.

Pedersen, T. 2004. Wordnet::similarity - measuring the relatedness of concepts. In *HLT-NAACL'04 (Demos)*, 267 – 270.

Resnik, P. 1995. Using information content to evaluate semantic similarity in a taxonomy. In *IJCAI'95*.

Salton, G., and Buckley, C. 1983. *Introduction to Modern Information Retreival*. McGraw-Hill Book Company.

Sekine, S.; Inui, K.; Dagan, I.; Dolan, B.; Giampiccolo, D.; and Magnini, B. 2009. Workshop proceedings. In *ACL-PASCAL Workshop on Textual Entailment and Paraphrasing*.

Tatu, M., and Moldovan, D. 2005. A semantic approach to recognizing textual entailment. In *HLT/EMNLP'05*, 371 – 378.

Vapnik, V. 1998. *Statistical Learning Theory*. Wiley, Chichester, GB.

Wang, R., and Neumann, G. 2007a. Recognizing textual entailment using a subsequence kernel method. In *AAAI'07*, 937 – 945.

Wang, R., and Neumann, G. 2007b. Recognizing textual entailment using sentence similarity based on dependency tree skeletons. In *ACL'07 Workshop on Textual Entailment and Paraphrasing*, 36 – 41.

Wu, Z., and Palmer, M. 1994. Verb semantics and lexical selection. In *ACL'94*, 133 – 138.

Zanzotto, F. M., and Dell'Arciprete, L. 2009. Efficient kernels for sentence pair classification. In *EMNLP'09*, 91 – 100.

Zanzotto, F. M., and Moschitti, A. 2006. Automatic learning of textual entailments with cross-pair similarities. In *ACL 2006*, 401–408.

Zanzotto, F. M.; Pennacchiotti, M.; and Moschitti, A. 2009. A machine learning approach to textual entailment recognition. *Natural Language Engineering* 15(4):551 – 582.

Zillober, C. 1993. Sequential convex programming in theory and praxis. *Structural Optimization* 79 – 86.