Contextual Preferences

Idan Szpektor, Ido Dagan, Roy Bar-Haim
Department of Computer Science
Bar-Ilan University
Ramat Gan, Israel
{szpekti,dagan,barhair}@cs.biu.ac.il

Jacob Goldberger
School of Engineering
Bar-Ilan University
Ramat Gan, Israel
goldbej@eng.biu.ac.il

Abstract

The validity of semantic inferences depends on the contexts in which they are applied. We propose a generic framework for handling contextual considerations within applied inference, termed *Contextual Preferences*. This framework defines the various context-aware components needed for inference and their relationships. Contextual preferences extend and generalize previous notions, such as selectional preferences, while experiments show that the extended framework allows improving inference quality on real application data.

1 Introduction

Applied semantic inference is typically concerned with inferring a target meaning from a given text. For example, to answer "*Who wrote Idomeneo?*", Question Answering (QA) systems need to infer the target meaning 'Mozart wrote Idomeneo' from a given text "*Mozart composed Idomeneo*". Following common Textual Entailment terminology (Giampiccolo et al., 2007), we denote the target meaning by *h* (for *hypothesis*) and the given text by *t*.

A typical applied inference operation is *matching*. Sometimes, h can be directly matched in t (in the example above, if the given sentence would be literally "*Mozart wrote Idomeneo*"). Generally, the target meaning can be expressed in t in many different ways. Indirect matching is then needed, using inference knowledge that may be captured through rules, termed here *entailment rules*. In our example, 'Mozart wrote Idomeneo' can be inferred using the rule 'X compose $Y \rightarrow X$ write Y'. Recently, several algorithms were proposed for automatically learning entailment rules and paraphrases (viewed as bi-directional entailment rules) (Lin and Pantel, 2001; Ravichandran and Hovy, 2002; Shinyama et al., 2002; Szpektor et al., 2004; Sekine, 2005).

A common practice is to try matching the structure of h, or of the left-hand-side of a rule r, within t. However, context should be considered to allow valid matching. For example, suppose that to find acquisitions of companies we specify the target *template hypothesis* (a hypothesis with variables) 'X acquire Y'. This h should not be matched in "*children acquire language quickly*", because in this context Y is not a company. Similarly, the rule 'X charge $Y \rightarrow X$ accuse Y' should not be applied to "*This store charged my account*", since the assumed sense of 'charge' in the rule is different than its sense in the text. Thus, the intended contexts for h and r and the context within the given t should be properly matched to verify valid inference.

Context matching at inference time was often approached in an application-specific manner (Harabagiu et al., 2003; Patwardhan and Riloff, 2007). Recently, some generic methods were proposed to handle context-sensitive inference (Dagan et al., 2006; Pantel et al., 2007; Downey et al., 2007; Connor and Roth, 2007), but these usually treat only a single aspect of context matching (see Section 6). We propose a comprehensive framework for handling various contextual considerations, termed *Contextual Preferences*. It extends and generalizes previous work, defining the needed contextual components and their relationships. We also present and implement concrete representation models and unsupervised matching methods for these components. While our presentation focuses on semantic inference using lexical-syntactic structures, the proposed framework and models seem suitable for other common types of representations as well.

We applied our models to a test set derived from the ACE 2005 event detection task, a standard Information Extraction (IE) benchmark. We show the benefits of our extended framework for textual inference and present component-wise analysis of the results. To the best of our knowledge, these are also the first unsupervised results for event argument extraction in the ACE 2005 dataset.

2 Contextual Preferences

2.1 Notation

As mentioned above, we follow the generic Textual Entailment (TE) setting, testing whether a target meaning hypothesis h can be inferred from a given text t. We allow h to be either a text or a *template*, a text fragment with variables. For example, "*The stock rose 8%*" entails an instantiation of the template hypothesis 'X gain Y'. Typically, h represents an information need requested in some application, such as a target predicate in IE.

In this paper, we focus on parse-based lexical-syntactic representation of texts and hypotheses, and on the basic inference operation of *matching*. Following common practice (de Salvo Braz et al., 2005; Romano et al., 2006; Bar-Haim et al., 2007), h is syntactically matched in t if it can be embedded in t's parse tree. For template hypotheses, the matching induces a mapping between h's variables and their instantiation in t.

Matching h in t can be performed either directly or indirectly using entailment rules. An *entailment rule* r: '$LHS \rightarrow RHS$' is a directional entailment relation between two templates. h is matched in t using r if LHS is matched in t and h matches RHS. In the example above, r: 'X rise $Y \rightarrow X$ gain Y' allows us to entail 'X gain Y', with "stock" and "8%" instantiating h's variables. We denote $vars(z)$ the set of variables of z, where z is a template or a rule.

2.2 Motivation

When matching considers only the structure of hypotheses, texts and rules it may result in incorrect inference due to contextual mismatches. For example, an IE system may identify mentions of public demonstrations using the hypothesis h: 'X demonstrate'. However, h should not be matched in "*Engineers demonstrated the new system*", due to a mismatch between the intended sense of 'demonstrate' in h and its sense in t. Similarly, when looking for physical attack mentions using the hypothesis 'X attack Y', we should not utilize the rule r: 'X accuse $Y \rightarrow X$ attack Y', due to a mismatch between a verbal attack in r and an intended physical attack in h. Finally, r: 'X produce $Y \rightarrow X$ lay Y' (applicable when X refers to poultry and Y to eggs) should not be matched in t: "*Bugatti produce the fastest cars*", due to a mismatch between the meanings of 'produce' in r and t. Overall, such incorrect inferences may be avoided by considering contextual information for t, h and r during their matching process.

2.3 The Contextual Preferences Framework

We propose the *Contextual Preferences* (CP) framework for addressing context at inference time. In this framework, the representation of an object z, where z may be a text, a template or an entailment rule, is enriched with contextual information denoted $cp(z)$. This information helps constraining or disambiguating the meaning of z, and is used to validate proper matching between pairs of objects.

We consider two components within $cp(z)$: (a) a representation for the global ("topical") context in which z typically occurs, denoted $cp_g(z)$; (b) a representation for the preferences and constraints ("hard" preferences) on the possible terms that can instantiate variables within z, denoted $cp_v(z)$. For example, cp_v('X produce $Y \rightarrow X$ lay Y') may specify that X's instantiations should be similar to "chicken" or "duck".

Contextual Preferences are used when entailment is assessed between a text t and a hypothesis h, either directly or by utilizing an entailment-rule r. On top of structural matching, we now require that the Contextual Preferences of the participants in the inference will also match. When h is directly matched in t, we require that each component in $cp(h)$ will be matched with its counterpart in $cp(t)$. When r is utilized, we additionally require that $cp(r)$ will be matched with both $cp(t)$ and $cp(h)$. Figure 1 summarizes the matching relationships between the CP

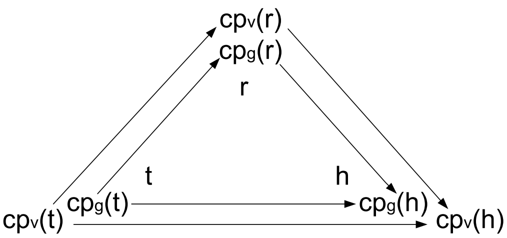

Figure 1: The directional matching relationships between a hypothesis (h), an entailment rule (r) and a text (t) in the Contextual Preferences framework.

components of h, t and r.

Like Textual Entailment inference, Contextual Preferences matching is directional. When matching h with t we require that the global context preferences specified by $cp_g(h)$ would subsume those induced by $cp_g(t)$, and that the instantiations of h's variables in t would adhere to the preferences in $cp_v(h)$ (since t should entail h, but not necessarily vice versa). For example, if the preferred global context of a hypothesis is sports, it would match a text that discusses the more specific topic of basketball.

To implement the CP framework, concrete models are needed for each component, specifying its representation, how it is constructed, and an appropriate matching procedure. Section 3 describes the specific CP models that were implemented in this paper.

The CP framework provides a generic view of contextual modeling in applied semantic inference. Mapping from a specific application to the generic framework follows the mappings assumed in the Textual Entailment paradigm. For example, in QA the hypothesis to be proved corresponds to the affirmative template derived from the question (e.g. h: '*X* invented the PC' for "*Who invented the PC?*"). Thus, $cp_g(h)$ can be constructed with respect to the question's focus while $cp_v(h)$ may be generated from the expected answer type (Moldovan et al., 2000; Harabagiu et al., 2003). Construction of hypotheses' CP for IE is demonstrated in Section 4.

3 Contextual Preferences Models

This section presents the current models that we implemented for the various components of the CP framework. For each component type we describe its representation, how it is constructed, and a corresponding unsupervised match score. Finally, the different component scores are combined to yield an overall match score, which is used in our experiments to rank inference instances by the likelihood of their validity. Our goal in this paper is to cover the entire scope of the CP framework by including specific models that were proposed in previous work, where available, and elsewhere propose initial models to complete the CP scope.

3.1 Contextual Preferences for Global Context

To represent the global context of an object z we utilize Latent Semantic Analysis (LSA) (Deerwester et al., 1990), a well-known method for representing the contextual-usage of words based on corpus statistics. We use LSA analysis of the BNC corpus[1], in which every term is represented by a normalized vector of the top 100 SVD dimensions, as described in (Gliozzo, 2005).

To construct $cp_g(z)$ we first collect a set of terms that are representative for the preferred general context of z. Then, the (single) vector which is the sum of the LSA vectors of the representative terms becomes the representation of $cp_g(z)$. This LSA vector captures the "average" typical contexts in which the representative terms occur.

The set of representative terms for a text t consists of all the nouns and verbs in it, represented by their lemma and part of speech. For a rule r: '$LHS \rightarrow RHS$', the representative terms are the words appearing in LHS and in RHS. For example, the representative terms for 'X divorce $Y \rightarrow X$ marry Y' are {*divorce:v, marry:v*}. As mentioned earlier, construction of hypotheses and their contextual preferences depends on the application at hand. In our experiments these are defined manually, as described in Section 4, derived from the manual definitions of target meanings in the IE data.

The score of matching the cp_g components of two objects, denoted by $m_g(\cdot, \cdot)$, is the Cosine similarity of their LSA vectors. Negative values are set to 0.

3.2 Contextual Preferences for Variables

3.2.1 Representation

For comparison with prior work, we follow (Pantel et al., 2007) and represent preferences for vari-

[1] http://www.natcorp.ox.ac.uk/

able instantiations using a distributional approach, and in addition incorporate a standard specification of named-entity types. Thus, cp_v is represented by two lists. The first list, denoted $cp_{v:e}$, contains examples for valid instantiations of that variable. For example, $cp_{v:e}(X \text{ kill } Y \rightarrow Y \text{ die of } X)$ may be [X: {*snakebite, disease*}, Y: {*man, patient*}]. The second list, denoted $cp_{v:n}$, contains the variable's preferred named-entity types (if any). For example, $cp_{v:n}(X \text{ born in } Y)$ may be [X: {*Person*}, Y: {*Location*}]. We denote $cp_{v:e}(z)[j]$ and $cp_{v:n}(z)[j]$ as the lists for a specific variable j of the object z.

For a text t, in which a template p is matched, the preference $cp_{v:e}(t)$ for each template variable is simply its instantiation in t. For example, when 'X eat Y' is matched in t: "*Many Americans eat fish regularly*", we construct $cp_{v:e}(t)$ = [X: {*Many Americans*}, Y: {*fish*}]. Similarly, $cp_{v:n}(t)$ for each variable is the named-entity type of its instantiation in t (if it is a named entity). We identify entity types using the default Lingpipe[2] Named-Entity Recognizer (NER), which recognizes the types *Location*, *Person* and *Organization*. In the above example, $cp_{v:n}(t)[X]$ would be {*Person*}.

For a rule r: $LHS \rightarrow RHS$, we automatically add to $cp_{v:e}(r)$ all the variable instantiations that were found common for both LHS and RHS in a corpus (see Section 4), as in (Pantel et al., 2007; Pennacchiotti et al., 2007). To construct $cp_{v:n}(r)$, we currently use a simple approach where each individual term in $cp_{v:e}(r)$ is analyzed by the NER system, and its type (if any) is added to $cp_{v:n}(r)$.

For a template hypothesis, we currently represent $cp_v(h)$ only by its list of preferred named-entity types, $cp_{v:n}$. Similarly to $cp_g(h)$, the preferred types for each template variable were adapted from those defined in our IE data (see Section 4).

To allow compatible comparisons with previous work (see Sections 5 and 6), we utilize in this paper only $cp_{v:e}$ when matching between $cp_v(r)$ and $cp_v(t)$, as only this representation was examined in prior work on context-sensitive rule applications. $cp_{v:n}$ is utilized for context matches involving $cp_v(h)$. We denote the score of matching two cp_v components by $m_v(\cdot, \cdot)$.

3.2.2 Matching $cp_{v:e}$

Our primary matching method is based on replicating the best-performing method reported in (Pantel et al., 2007), which utilizes the CBC distributional word clustering algorithm (Pantel, 2003). In short, this method extends each $cp_{v:e}$ list with CBC clusters that contain at least one term in the list, scoring them according to their "relevancy". The score of matching two $cp_{v:e}$ lists, denoted here $S_{CBC}(\cdot, \cdot)$, is the score of the highest scoring member that appears in both lists.

We applied the final binary match score presented in (Pantel et al., 2007), denoted here $binaryCBC$: $m_{v:e}(r,t)$ is 1 if $S_{CBC}(r,t)$ is above a threshold and 0 otherwise. As a more natural ranking method, we also utilize S_{CBC} directly, denoted $rankedCBC$, having $m_{v:e}(r,t) = S_{CBC}(r,t)$.

In addition, we tried a simpler method that directly compares the terms in two $cp_{v:e}$ lists, utilizing the commonly-used term similarity metric of (Lin, 1998a). This method, denoted LIN, uses the same raw distributional data as CBC but computes only pair-wise similarities, without any clustering phase. We calculated the scores of the 1000 most similar terms for every term in the Reuters RVC1 corpus[3]. Then, a directional similarity of term a to term b, $s(a,b)$, is set to be their similarity score if a is in b's 1000 most similar terms and 0 otherwise. The final score of matching r with t is determined by a nearest-neighbor approach, as the score of the most similar pair of terms in the corresponding two lists of the same variable: $m_{v:e}(r,t) = \max_{j \in vars(r)}[\max_{a \in cp_{v:e}(t)[j], b \in cp_{v:e}(r)[j]}[s(a,b)]]$.

3.2.3 Matching $cp_{v:n}$

We use a simple scoring mechanism for comparing between two named-entity types a and b, $s(a,b)$: 1 for identical types and 0.8 otherwise.

A variable j has a single preferred entity type in $cp_{v:n}(t)[j]$, the type of its instantiation in t. However, it can have several preferred types for h. When matching h with t, j's match score is that of its highest scoring type, and the final score is the product of all variable scores: $m_{v:n}(h,t) = \prod_{j \in vars(h)}(\max_{a \in cp_{v:n}(h)[j]}[s(a, cp_{v:n}(t)[j])])$.

Variable j may also have several types in r, the

[2] http://www.alias-i.com/lingpipe/

[3] http://about.reuters.com/researchandstandards/corpus/

types of the common arguments in $cp_{v:e}(r)$. When matching h with r, $s(a, cp_{v:n}(t)[j])$ is replaced with the average score for a and each type in $cp_{v:n}(r)[j]$.

3.3 Overall Score for a Match

A final score for a given match, denoted *allCP*, is obtained by the product of all six matching scores of the various CP components (multiplying by 1 if a component score is missing). The six scores are the results of matching any of the two components of h, t and r: $m_g(h, t)$, $m_v(h, t)$, $m_g(h, r)$, $m_v(h, r)$, $m_g(r, t)$ and $m_v(r, t)$ (as specified above, $m_v(r, t)$ is based on matching $cp_{v:e}$ while $m_v(h, r)$ and $m_v(h, t)$ are based on matching $cp_{v:n}$). We use $rankedCBC$ for calculating $m_v(r, t)$.

Unlike previous work (e.g. (Pantel et al., 2007)), we also utilize the *prior* score of a rule r, which is provided by the rule-learning algorithm (see next section). We denote by *allCP+pr* the final match score obtained by the product of the *allCP* score with the prior score of the matched rule.

4 Experimental Settings

Evaluating the contribution of Contextual Preferences models requires: (a) a sample of test hypotheses, and (b) a corresponding corpus that contains sentences which entail these hypotheses, where all hypothesis matches (either direct or via rules) are annotated. We found that the available event mention annotations in the ACE 2005 training set[4] provide a useful test set that meets these generic criteria, with the added value of a standard real-world dataset.

The ACE annotation includes 33 types of events, for which all event mentions are annotated in the corpus. The annotation of each mention includes the instantiated arguments for the predicates, which represent the participants in the event, as well as general attributes such as time and place. ACE guidelines specify for each event type its possible arguments, where all arguments are optional. Each argument is associated with a semantic role and a list of possible named-entity types. For instance, an *Injure* event may have the arguments {*Agent, Victim, Instrument, Time, Place*}, where *Victim* should be a person.

For each event type we manually created a small set of template hypotheses that correspond to the given event predicate, and specified the appropriate semantic roles for each variable. We considered only binary hypotheses, due to the type of available entailment rules (see below). For *Injure,* the set of hypotheses included *'A injure V'* and *'injure V in T'* where *role(A)=*{*Agent, Instrument*}, *role(V)=*{*Victim*}, and *role(T)=*{*Time, Place*}. Thus, correct match of an argument corresponds to correct role identification. The templates were represented as Minipar (Lin, 1998b) dependency parse-trees.

The Contextual Preferences for *h* were constructed manually: the named-entity types for $cp_{v:n}(h)$ were set by adapting the entity types given in the guidelines to the types supported by the Lingpipe NER (described in Section 3.2). $cp_g(h)$ was generated from a short list of nouns and verbs that were extracted from the verbal event definition in the ACE guidelines. For *Injure*, this list included {*injure:v, injury:n, wound:v*}. This assumes that when writing down an event definition the user would also specify such representative keywords.

Entailment-rules for a given *h* (rules in which *RHS* is equal to *h*) were learned automatically by the *DIRT* algorithm (Lin and Pantel, 2001), which also produces a quality score for each rule. We implemented a canonized version of *DIRT* (Szpektor and Dagan, 2007) on the Reuters corpus parsed by Minipar. Each rule's arguments for $cp_v(r)$ were also collected from this corpus.

We assessed the CP framework by its ability to correctly rank, for each predicate (event), all the candidate entailing mentions that are found for it in the test corpus. Such ranking evaluation is suitable for unsupervised settings, with a perfect ranking placing all correct mentions before any incorrect ones. The candidate mentions are found in the parsed test corpus by matching the specified event hypotheses, either directly or via the given set of entailment rules, using a syntactic matcher similar to the one in (Szpektor and Dagan, 2007). Finally, the mentions are ranked by their match scores, as described in Section 3.3. As detailed in the next section, those candidate mentions which are also annotated as mentions of the same event in ACE are considered correct.

The evaluation aims to assess the correctness of inferring a target semantic meaning, which is de-

[4] http://projects.ldc.upenn.edu/ace/

noted by a specific predicate. Therefore, we eliminated four ACE event types that correspond to multiple distinct predicates. For instance, the *Transfer-Money* event refers to both *donating* and *lending* money, which are not distinguished by the ACE annotation. We also omitted three events with less than 10 mentions and two events for which the given set of learned rules could not match any mention. We were left with 24 event types for evaluation, which amount to 4085 event mentions in the dataset. Out of these, our binary templates can correctly match only mentions with at least two arguments, which appear 2076 times in the dataset.

Comparing with previous evaluation methodologies, in (Szpektor et al., 2007; Pantel et al., 2007) proper context matching was evaluated by post-hoc judgment of a sample of rule applications for a sample of rules. Such annotation needs to be repeated each time the set of rules is changed. In addition, since the corpus annotation is not exhaustive, recall could not be computed. By contrast, we use a standard real-world dataset, in which all mentions are annotated. This allows immediate comparison of different rule sets and matching methods, without requiring any additional (post-hoc) annotation.

5 Results and Analysis

We experimented with three rule setups over the ACE dataset, in order to measure the contribution of the CP framework. In the first setup no rules are used, applying only direct matches of template hypotheses to identify event mentions. In the other two setups we also utilized *DIRT*'s top 50 or 100 rules for each hypothesis.

A match is considered correct when all matched arguments are extracted correctly according to their annotated event roles. This main measurement is denoted *All*. As an additional measurement, denoted *Any*, we consider a match as correct if at least one argument is extracted correctly.

Once event matches are extracted, we first measure for each event its Recall, the number of correct mentions identified out of all annotated event mentions[5] and Precision, the number of correct matches out of all extracted candidate matches. These figures quantify the baseline performance of the DIRT rule set used. To assess our ranking quality, we measure for each event the commonly used Average Precision (AP) measure (Voorhees and Harmann, 1998), which is the area under the non-interpolated recall-precision curve, while considering for each setup all correct extracted matches as 100% Recall. Overall, we report *Mean Average Precision* (*MAP*), macro average *Precision* and macro average *Recall* over the ACE events. Tables 1 and 2 summarize the main results of our experiments. As far as we know, these are the first published unsupervised results for identifying event arguments in the ACE 2005 dataset.

Examining Recall, we see that it increases substantially when rules are applied: by more than 100% for the top 50 rules, and by about 150% for the top 100, showing the benefit of entailment-rules to covering language variability. The difference between *All* and *Any* results shows that about 65% of the rules that correctly match one argument also match correctly both arguments.

We use two baselines for measuring the CP ranking contribution: Precision, which corresponds to the expected MAP of random ranking, and MAP of ranking using the *prior* rule score provided by *DIRT*. Without rules, the baseline *All* Precision is 34.1%, showing that even the manually constructed hypotheses, which correspond directly to the event predicate, extract event mentions with limited accuracy when context is ignored. When rules are applied, Precision is very low. But ranking is considerably improved using only the prior score (from 1.4% to 22.7% for 50 rules), showing that the prior is an informative indicator for valid matches.

Our main result is that the *allCP* and *allCP+pr* methods rank matches statistically significantly better than the baselines in all setups (according to the Wilcoxon double-sided signed-ranks test at the level of 0.01 (Wilcoxon, 1945)). In the *All* setup, ranking is improved by 70% for direct matching (Table 1). When entailment-rules are also utilized, prior-only ranking is improved by about 35% and 50% when using *allCP* and *allCP+pr*, respectively (Table 2). Figure 2 presents the average Recall-Precision curve of the '50 Rules, All' setup for applying *allCP* or *allCP+pr*, compared to prior-only ranking baseline (other setups behave similarly). The improvement in ranking is evident: the drop in precision is signif-

[5] For Recall, we ignored mentions with less than two arguments, as they cannot be correctly matched by binary templates.

	R	P	MAP (%)		
	(%)	(%)	cp_v	cp_g	*allCP*
All	14.0	34.1	46.5	52.2	60.2
Any	21.8	66.0	72.2	80.5	84.1

Table 1: Recall (R), Precision (P) and Mean Average Precision (MAP) when only matching template hypotheses directly.

	#	R	P	MAP (%)		
	Rules	(%)	(%)	prior	*allCP*	*allCP+pr*
All	50	29.6	1.4	22.7	30.6	34.1
	100	34.9	0.7	20.5	26.3	30.2
Any	50	46.5	3.5	41.2	43.7	48.6
	100	52.9	1.8	35.5	35.1	40.8

Table 2: Recall (R), Precision (P) and Mean Average Precision (MAP) when also using rules for matching.

icantly slower when CP is used. The behavior of CP with and without the prior is largely the same up to 50% Recall, but later on our implemented CP models are noisier and should be combined with the prior rule score.

Templates are incorrectly matched for several reasons. First, there are context mismatches which are not scored sufficiently low by our models. Another main cause is incorrect learned rules in which LHS and RHS are topically related, e.g. 'X convict $Y \rightarrow X$ arrest Y', or rules that are used in the wrong entailment direction, e.g. 'X marry $Y \rightarrow X$ divorce Y' (*DIRT* does not learn rule direction). As such rules do correspond to plausible contexts of the hypothesis, their matches obtain relatively high CP scores. In addition, some incorrect matches are caused by our syntactic matcher, which currently does not handle certain phenomena such as co-reference, modality or negation, and due to Minipar parse errors.

5.1 Component Analysis

Table 3 displays the contribution of different CP components to ranking, when adding only that component's match score to the baselines, and under ablation tests, when using all CP component scores except the tested component, with or without the prior.

As it turns out, matching h with t (i.e. $cp(h,t)$, which combines $cp_g(h,t)$ and $cp_v(h,t)$) is most useful. With our current models, using only $cp(h,t)$ along with the prior, while ignoring $cp(r)$, achieves

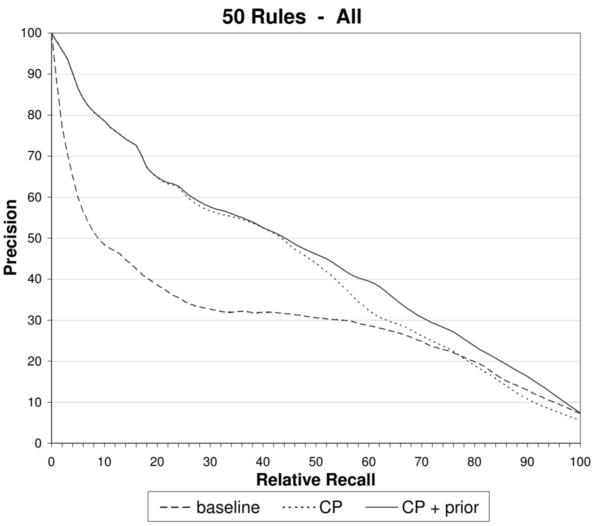

Figure 2: Recall-Precision curves for ranking using: (a) only the prior (baseline); (b) *allCP*; (c) *allCP+pr*.

the highest score in the table. The strong impact of matching h and t's preferences is also evident in Table 1, where ranking based on either cp_g or cp_v substantially improves precision, while their combination provides the best ranking. These results indicate that the two CP components capture complementary information and both are needed to assess the correctness of a match.

When ignoring the prior rule score, $cp(r,t)$ is the major contributor over the baseline Precision. For $cp_v(r,t)$, this is in synch with the result in (Pantel et al., 2007), which is based on this single model without utilizing prior rule scores. On the other hand, $cp_v(r,t)$ does not improve the ranking when the prior is used, suggesting that this contextual model for the rule's variables is not stronger than the context-insensitive prior rule score. Furthermore, relative to this $cp_v(r,t)$ model from (Pantel et al., 2007), our combined *allCP* model, with or without the prior (first row of Table 2), obtains statistically significantly better ranking (at the level of 0.01).

Comparing between the algorithms for matching $cp_{v:e}$ (Section 3.2.2) we found that while $rankedCBC$ is statistically significantly better than $binaryCBC$, $rankedCBC$ and LIN generally achieve the same results. When considering the tradeoffs between the two, LIN is based on a much simpler learning algorithm while CBC's output is more compact and allows faster CP matches.

	Addition To		Ablation From	
	P	prior	*allCP*	*allCP+pr*
Baseline	1.4	22.7	30.6	34.1
$cp_g(h,t)$	*10.4	*35.4	32.4	33.7
$cp_v(h,t)$	*11.0	29.9	27.6	32.9
$cp(h,t)$	*8.9	***37.5**	28.6	30.0
$cp_g(r,t)$	*4.2	*30.6	**32.5**	35.4
$cp_v(r,t)$	*21.7	21.9	*12.9	33.6
$cp(r,t)$	*26.0	*29.6	*17.9	**36.8**
$cp_g(h,r)$	*8.1	22.4	31.9	34.3
$cp_v(h,r)$	*10.7	22.7	*27.9	34.4
$cp(h,r)$	*16.5	22.4	*29.2	34.4
$cp_g(h,r,t)$	*7.7	*30.2	*27.5	*29.2
$cp_v(h,r,t)$	***27.5**	29.2	*7.7	30.2

* Indicates statistically significant changes compared to the baseline, according to the Wilcoxon test at the level of 0.01.

Table 3: MAP(%), under the '50 rules, All' setup, when adding component match scores to Precision (P) or prior-only MAP baselines, and when ranking with *allCP* or *allCP+pr* methods but ignoring that component scores.

Currently, some models do not improve the results when the prior is used. Yet, we would like to further weaken the dependency on the prior score, since it is biased towards frequent contexts. We aim to properly identify also infrequent contexts (or meanings) at inference time, which may be achieved by better CP models. More generally, when used on top of all other components, some of the models slightly degrade performance, as can be seen by those figures in the ablation tests which are higher than the corresponding baseline. However, due to their different roles, each of the matching components might capture some unique preferences. For example, $cp(h,r)$ should be useful to filter out rules that don't match the intended meaning of the given h. Overall, this suggests that future research for better models should aim to obtain a marginal improvement by each component.

6 Related Work

Context sensitive inference was mainly investigated in an application-dependent manner. For example, (Harabagiu et al., 2003) describe techniques for identifying the question focus and the answer type in QA. (Patwardhan and Riloff, 2007) propose a supervised approach for IE, in which relevant text regions for a target relation are identified prior to applying extraction rules.

Recently, the need for context-aware inference was raised (Szpektor et al., 2007). (Pantel et al., 2007) propose to learn the preferred instantiations of rule variables, termed Inferential Selectional Preferences (ISP). Their clustering-based model is the one we implemented for $m_v(r,t)$. A similar approach is taken in (Pennacchiotti et al., 2007), where LSA similarity is used to compare between the preferred variable instantiations for a rule and their instantiations in the matched text. (Downey et al., 2007) use HMM-based similarity for the same purpose. All these methods are analogous to matching $cp_v(r)$ with $cp_v(t)$ in the CP framework.

(Dagan et al., 2006; Connor and Roth, 2007) proposed generic approaches for identifying valid applications of lexical rules by classifying the surrounding global context of a word as valid or not for that rule. These approaches are analogous to matching $cp_g(r)$ with $cp_g(t)$ in our framework.

7 Conclusions

We presented the Contextual Preferences (CP) framework for assessing the validity of inferences in context. CP enriches the representation of textual objects with typical contextual information that constrains or disambiguates their meaning, and provides matching functions that compare the preferences of objects involved in the inference. Experiments with our implemented CP models, over real-world IE data, show significant improvements relative to baselines and some previous work.

In future research we plan to investigate improved models for representing and matching CP, and to extend the experiments to additional applied datasets. We also plan to apply the framework to lexical inference rules, for which it seems directly applicable.

Acknowledgements

The authors would like to thank Alfio Massimiliano Gliozzo for valuable discussions. This work was partially supported by ISF grant 1095/05, the IST Programme of the European Community under the PASCAL Network of Excellence IST-2002-506778, the NEGEV project (www.negev-initiative.org) and the FBK-irst/Bar-Ilan University collaboration.

References

Roy Bar-Haim, Ido Dagan, Iddo Greental, and Eyal Shnarch. 2007. Semantic inference at the lexical-syntactic level. In *Proceedings of AAAI*.

Michael Connor and Dan Roth. 2007. Context sensitive paraphrasing with a global unsupervised classifier. In *Proceedings of the European Conference on Machine Learning (ECML)*.

Ido Dagan, Oren Glickman, Alfio Gliozzo, Efrat Marmorshtein, and Carlo Strapparava. 2006. Direct word sense matching for lexical substitution. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of ACL*.

Rodrigo de Salvo Braz, Roxana Girju, Vasin Punyakanok, Dan Roth, and Mark Sammons. 2005. An inference model for semantic entailment in natural language. In *Proceedings of the National Conference on Artificial Intelligence (AAAI)*.

Scott C. Deerwester, Susan T. Dumais, Thomas K. Landauer, George W. Furnas, and Richard A. Harshman. 1990. Indexing by latent semantic analysis. *Journal of the American Society of Information Science*, 41(6):391–407.

Doug Downey, Stefan Schoenmackers, and Oren Etzioni. 2007. Sparse information extraction: Unsupervised language models to the rescue. In *Proceedings of the 45th Annual Meeting of ACL*.

Danilo Giampiccolo, Bernardo Magnini, Ido Dagan, and Bill Dolan. 2007. The third pascal recognizing textual entailment challenge. In *Proceedings of the ACL-PASCAL Workshop on Textual Entailment and Paraphrasing*.

Alfio Massimiliano Gliozzo. 2005. *Semantic Domains in Computational Linguistics*. Ph.D. thesis. Advisor-Carlo Strapparava.

Sanda M. Harabagiu, Steven J. Maiorano, and Marius A. Paşca. 2003. Open-domain textual question answering techniques. *Nat. Lang. Eng.*, 9(3):231–267.

Dekang Lin and Patrick Pantel. 2001. Discovery of inference rules for question answering. In *Natural Language Engineering*, volume 7(4), pages 343–360.

Dekang Lin. 1998a. Automatic retrieval and clustering of similar words. In *Proceedings of COLING-ACL*.

Dekang Lin. 1998b. Dependency-based evaluation of minipar. In *Proceedings of the Workshop on Evaluation of Parsing Systems at LREC*.

Dan Moldovan, Sanda Harabagiu, Marius Pasca, Rada Mihalcea, Roxana Girju, Richard Goodrum, and Vasile Rus. 2000. The structure and performance of an open-domain question answering system. In *Proceedings of the 38th Annual Meeting of ACL*.

Patrick Pantel, Rahul Bhagat, Bonaventura Coppola, Timothy Chklovski, and Eduard Hovy. 2007. ISP: Learning inferential selectional preferences. In *Human Language Technologies 2007: The Conference of NAACL; Proceedings of the Main Conference*.

Patrick Andre Pantel. 2003. *Clustering by committee*. Ph.D. thesis. Advisor-Dekang Lin.

Siddharth Patwardhan and Ellen Riloff. 2007. Effective information extraction with semantic affinity patterns and relevant regions. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*.

Marco Pennacchiotti, Roberto Basili, Diego De Cao, and Paolo Marocco. 2007. Learning selectional preferences for entailment or paraphrasing rules. In *Proceedings of RANLP*.

Deepak Ravichandran and Eduard Hovy. 2002. Learning surface text patterns for a question answering system. In *Proceedings of the 40th Annual Meeting of ACL*.

Lorenza Romano, Milen Kouylekov, Idan Szpektor, Ido Dagan, and Alberto Lavelli. 2006. Investigating a generic paraphrase-based approach for relation extraction. In *Proceedings of the 11th Conference of the EACL*.

Satoshi Sekine. 2005. Automatic paraphrase discovery based on context and keywords between ne pairs. In *Proceedings of IWP*.

Yusuke Shinyama, Satoshi Sekine, Sudo Kiyoshi, and Ralph Grishman. 2002. Automatic paraphrase acquisition from news articles. In *Proceedings of Human Language Technology Conference*.

Idan Szpektor and Ido Dagan. 2007. Learning canonical forms of entailment rules. In *Proceedings of RANLP*.

Idan Szpektor, Hristo Tanev, Ido Dagan, and Bonaventura Coppola. 2004. Scaling web-based acquisition of entailment relations. In *Proceedings of EMNLP 2004*, pages 41–48, Barcelona, Spain.

Idan Szpektor, Eyal Shnarch, and Ido Dagan. 2007. Instance-based evaluation of entailment rule acquisition. In *Proceedings of the 45th Annual Meeting of ACL*.

Ellen M. Voorhees and Donna Harmann. 1998. Overview of the seventh text retrieval conference (trec–7). In *The Seventh Text Retrieval Conference*.

Frank Wilcoxon. 1945. Individual comparisons by ranking methods. *Biometrics Bulletin*, 1(6):80–83.