

Classification-based Contextual Preferences

Shachar Mirkin, Ido Dagan, Lili Kotlerman

Bar-Ilan University
Ramat Gan, Israel

{mirkins, dagan, davidol}@cs.biu.ac.il

Idan Szpektor

Yahoo! Research
Haifa, Israel

idan@yahoo-inc.com

Abstract

This paper addresses context matching in textual inference. We formulate the task under the *Contextual Preferences* framework which broadly captures contextual aspects of inference. We propose a generic classification-based scheme under this framework which coherently attends to context matching in inference and may be employed in any inference-based task. As a test bed for our scheme we use the Name-based Text Categorization (TC) task. We define an integration of Contextual Preferences into the TC setting and present a concrete self-supervised model which instantiates the generic scheme and is applied to address context matching in the TC task. Experiments on standard TC datasets show that our approach outperforms the state of the art in context modeling for Name-based TC.

1 Introduction

Textual inference is prevalent in text understanding applications. For example, in Question Answering (QA) the expected answer should be inferred from retrieved passages, and in Information Extraction (IE) the meaning of the target event is inferred from its mention in the text.

Lexical inferences make a substantial part of the inference process. In such cases, a target term is inferred from text expressions based on either one of two types of lexical matches: (i) a *direct match* of the target term in the text. For instance, the IE event injure may be detected by finding the word *injure* in the text; (ii) an *indirect match*, through a term that implies the meaning of the target term, e.g. inferring *injure* from *hurt*.

In either case, due to word ambiguity, it is necessary to validate that the context of the match conforms with the intended meaning of the target term before carrying out an inference operation based on this match. For example, “*You hurt my feelings*” constitutes an invalid context for the injure event as *hurt* in this text does not refer to a physical injury. Similarly, inferring the protest-related event demonstrate based on *demo* is deemed invalid although *demo* implies the meaning of the word *demonstrate* in other contexts, e.g., concerning *software demonstration*.

Although seemingly equivalent, a closer look reveals that the above two examples correspond to two distinct contextual mismatch situations. While the match of *hurt* is invalid for injure in the particular given context, an inference based on *demo* is invalid for the protest demonstrate event in any context.

Thus, several types of context matching are involved in textual inference. While most prior work addressed only specific context matching scenarios, Szpektor et al. (2008) presented a broader view, proposing a generic framework for context matching in inference, termed Contextual Preferences (CP). CP specifies the types of context matching that need to be considered in inference, allowing a model of choice to be applied for validating each type of match. Szpektor et al. applied CP to an IE task using different models to validate each type of context match.

In this work we adopt CP as our context matching framework and propose a novel classification-based scheme which provides unified modeling for CP. We represent typical contexts of the textual objects that participate in inference using classifiers; at inference time, each match is assessed by the respective classifiers which determine its contextual validity.

As a test bed we applied our scheme to the task

of Name-based Text Categorization. This is an unsupervised setting of TC where the only input given is the category name, and in which context validation is of high importance. We instantiate the scheme with a novel self-supervised model and apply it to the TC task. We suggest a method for integrating any CP-based context matching model into TC and use it to combine the context matching scores generated by our model. Results on two standard TC datasets show that our approach outperforms the state of the art context model for this task and suggest applying this scheme to additional inference-based applications.

2 Background

2.1 Context matching in inference

Word ambiguity has been traditionally addressed through Word Sense Disambiguation (WSD) (Navigli, 2009). The WSD task requires selecting the meaning of a target term from amongst a predefined set of senses, based on sense-inventories such as WordNet (Fellbaum, 1998).

An alternative approach eliminates the reliance on such inventories. Instead of explicit sense identification, a direct sense-match between terms is pursued (Dagan et al., 2006). *Lexical substitution* (McCarthy and Navigli, 2009) is probably the most commonly known task that follows this approach. *Context matching* is a generalization of lexical substitution, which seeks a match between terms in context, not necessarily for the purpose of substitution. For instance, the word *played* in “*U2 played their first-ever concert in Russia*” contextually matches *music*, although *music* cannot substitute *played* in this context. The context matching task, therefore, is to determine (by quantifying or giving a binary decision) the validity of a match between two terms in context.

In Section 1 we informally presented two cases of contextual mismatches. A comprehensive view of context matching types is provided by the Contextual Preferences framework (Szpektor et al., 2008). CP is phrased in terms of the Textual Entailment (TE) paradigm (Dagan et al., 2009). In TE, a *text* t entails a textual *hypothesis* h if the meaning of h can be inferred from t . Formulating the IE example from Section 1 within TE, h may be the name of the target event, *injure*, and t is a text segment from which h can be inferred. A direct match occurs when a term

in h is identical to a term in t . An inference based on an indirect match is viewed as the application of a *lexical entailment rule*, r , such as ‘*hurt* \Rightarrow *injure*’, where the entailing left-hand side (LHS) of the rule (*hurt*) is matched in the text, while the entailed right-hand side (RHS), *injure*, is matched in the hypothesis.

Hence, three *inference objects* take part in inference operations: t , h and r . Most prior work addressed only specific contextual matches between these objects. For example, Harabagiu et al. (2003) matched the contexts of t and h for QA (answer and question, respectively); Barak et al. (2009) matched t and h (document and category) in TC, while other works, including those applying lexical substitution, typically validated the context match between t and r (Kauchak and Barzilay, 2006; Dagan et al., 2006; Pantel et al., 2007; Connor and Roth, 2007).

In comparison, in the CP framework, all possible contextual matches among t , h and r are considered: $t - h$, $t - r$ and $r - h$. The three context matches are depicted in Figure 1 (left). In CP, the representation of each inference object is enriched with contextual information which is used to characterize its valid contexts. Such information may be the words of the event description in IE, corpus instances based on which a rule was learned, or an annotation of relevant WordNet senses in Name-based TC. For example, a category name *hockey* may be assigned with the sense number corresponding to *ice hockey*, but not to *field hockey*, in order to designate information that limits the valid contexts of the category to the former among the two meanings of the name.

Before an inference operation is performed, the context representations of each pair among the participating objects should be *matched* by a context model in order to assess the contextual validity of the operation. Along with the context representation and the specific context matching models, the way context model decisions are combined needs to be specified in a concrete implementation of the CP framework.

2.2 Context matching models

Several approaches were taken in prior work to model context matching, mostly within the scope of learning *selectional preferences* of templated lexical-syntactic rules (e.g. ‘ $X \xleftarrow{subj} hit \xrightarrow{obj} Y$ ’ \Rightarrow ‘ $X \xleftarrow{subj} attack \xrightarrow{obj} Y$ ’).

Pantel et al. (2007) and Szpektor et al. (2008) represented the context of such rules as the intersection of preferences of the rule’s LHS and RHS, namely the observed argument instantiations or their semantic classes. A rule is deemed applicable to a given text if the argument instantiations in the text are similar to the selectional preferences of the rule. To overcome sparseness, other works represented context in latent space. Pennacchiotti et al. (2007) and Szpektor et al. (2008) measured the similarity between the Latent Semantic Analysis (LSA) (Deerwester et al., 1990) representations of matched contexts. Dinu and Lapata (2010) used Latent Dirichlet Allocation (LDA) (Blei et al., 2003) to model templates’ latent senses, determining rule applicability based on the similarity between the two sides of the rule when instantiated by the context, while Ritter et al. (2010) used LDA to model argument classes, considering a rule valid for a given argument instantiation if its instantiated templates are drawn from the same hidden topic.

A different approach is provided by classification-based models which learn classifiers for inference objects. A classifier is trained based on positive and negative examples which represent valid or invalid contexts of the object; from those, features characterizing the context are extracted, e.g. words in a window around the target term or syntactic links with it. Given a new context, the classifier assesses its validity with respect to the learned classification model.

Classifiers in prior work were applied to determine rule applicability in a given context ($t - r$). Training a classifier for word paraphrasing, Kauchak and Barzilay (2006) used occurrences of the rule’s RHS as positive context examples, and randomly picked negative examples. A similar approach was applied by Dagan et al. (2006), which used a single-class SVM to avoid selecting negative examples. In both works, a resulting classifier represents a word with all its senses intermixed. Clearly, this poses no problem for monosemous words, but is biased towards the more common senses of polysemous words. Indeed, Dagan et al. (2006) report a negative correlation between the degree of polysemy of a word and the performance of its classifier. Connor and Roth (2007) used per-rule classifiers to produce a noisy training set for learning a global classifier for verb substitution.

In this work we follow the classification-based approach which seems appealing for several reasons.

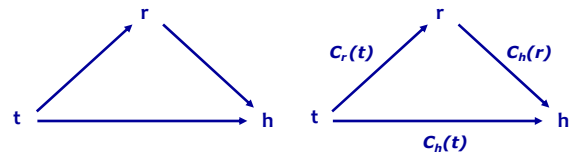


Figure 1: Left: An illustration of the CP relationships as in (Szpektor et al., 2008), with arrows indicating the context matching direction; Right: The application of classifiers to the tested contexts under our scheme.

First, it allows seamlessly integrating various types of information via classifiers’ features; unlike some of the above models, it is not inherently dependent on the type of rules that are utilized and easily accommodates to both lexical and lexical-syntactic rules through the choice of features. In addition, it does not rely on a predefined similarity measure and provides flexibility in terms of model’s parameters. Finally, this approach captures the notion of directionality which is fundamental in textual inference, and is therefore better suited to applied inference than previously proposed symmetric context models.

In comparison to prior classification-based models, our approach addresses all three context matches specified by CP, rather than only the rule-text match. It is not limited to substitutable terms or even to terms with the same part of speech. In addition, we avoid learning a classifier for all senses combined, but rather learn it for the specific intended meaning.

2.3 Name-based Text Categorization

Name-based TC (Gliozzo et al., 2009) is an unsupervised setting of Text Categorization in which the only input provided is the category name, e.g. trade, ‘mergers and acquisitions’ or guns. When category names are ambiguous, e.g. space, categories are not well defined; thus, auxiliary information is expected to accompany the name for disambiguation, such as a list of relevant senses or a category description.

Typically, unsupervised TC consists of two steps. First, an unsupervised method is applied to an unlabeled corpus, automatically labeling some of the documents to categories. Then, the labeled documents from the first step are used to train a supervised TC classifier which is used to label any document in the test set (Gliozzo et al., 2009; Downey and Etzioni, 2009; Barak et al., 2009).

In this work we focus on the above unsupervised step. Gliozzo et al. (2009) addressed this task by representing both documents and categories by LSA vectors which implicitly capture contextual similarities between terms. Each document was then assigned to the most similar category based on cosine similarity between the LSA vectors. Barak et al. (2009) required an occurrence of a term entailing the category name (or the category name itself) in order to regard the category as a candidate for the document. To assess the contextual validity of the match, they used LSA document-category similarity as in (Gliozzo et al., 2009). For example, to classify a document into the category *medicine*, at least one lexical entailment rule, e.g. ‘*drug* \Rightarrow *medicine*’, should be matched in the document. Then, the validity of *drug* for *medicine* in the matched document is assessed by the LSA context model. In this work we adopt Barak et al.’s requirement for a match for the category in the document, but address context matching in an entirely different way.

Name-based TC provides a convenient setting for evaluating context matching approaches for two main reasons. First, all types of context matchings are realized in this application (see Section 3); second, as the hypothesis consists of a single term or a few terms, the TC gold standard annotation corresponds quite directly to the context matching task for lexical inferences; in other applications where longer hypotheses are involved, context matching performance may be masked by other factors.

3 Contextual Matches in TC

Within Name-based TC, the Textual Entailment terminology is mapped as follows: h is a term denoting the category name (e.g. *merger* or *acquisition*); t is a matched term in the document to be categorized from which h may be inferred; and a match refers to an occurrence in the document of either h (direct match) or the LHS of an entailment rule r whose RHS is a category name (indirect match).¹

Under the CP view, a context model needs to address the following three context matching cases within a TC setting.

$t - h$: Assessing the validity of a match in the document with respect to the category’s intended meaning.

¹Note that t and h both refer here to individual terms.

For example, the occurrence of the category name space (in the sense of *outer space*) in “*the server ran out of disk space*” does not indicate a space-related text, and should be dismissed by the context model.

$t - r$: This case refers to a rule match in the document. A context model should ensure that the meaning of a match is compatible with that of the rule. For example, ‘*alien* \Rightarrow *space*’ is a valid rule for the space category. Yet, it should not be applied to “*The US welcomes a large number of aliens every year*”, since *alien* in this sentence has a different meaning than the intended meaning of the rule.

$r - h$: The match between the intended meanings of the category name and the RHS of the rule. For instance, the rule ‘*room* \Rightarrow *space*’ is not suitable at all for the (outer) space category.

4 A Classification-based Scheme for CP

Szpektor et al. (2008) introduced a vector-space model to implement CP, in which the text t , the rule r and the hypothesis h share the same contextual representation. However, in CP, r , h and t have non-symmetric roles: the context of t should be tested as valid for r and h and not vice versa, and the context of r should be validated for h and not the other way around. This stems from the need to consider directionality in context matching. For instance, a text about *football* typically constitutes a valid context for the more general *sports* context, but not vice versa. Indeed, directionality may be captured in vector-space models by using a directional similarity measure (Kotlerman et al., 2010), but only symmetric measures were used in context matching work so far.

Based on this distinction between the inference objects’ roles, we present a novel scheme that uses two types of classifiers to represent context:

C_h : A classifier that identifies valid contexts for h . It tests contexts of t (for $t - h$ matching) or r (for $r - h$ matching), assigning them scores $C_h(t)$ and $C_h(r)$, respectively.

C_r : A classifier that identifies valid contexts for applying the rule r . It tests the context of t , assigning it a score $C_r(t)$.

Figure 1 (right) shows the classifiers scores which are assigned to each of the matching types.

Hence, h always acts as the *classifying object*, t is always the *classified object*, while r acts as both. Context matching is quantified by the degree by which the classified object represents a valid context for the classifying object in a given inference scenario.

In comparison to the CP implementation in (Szpektor et al., 2008), our approach uses a unified model which captures directionality in context matching.

To instantiate the scheme, one needs to define the way training examples are obtained and processed. This may be done within supervised classification, where labeled examples are provided, or – as we do in this work – using self-supervised classifiers which obtain training examples automatically. We present such an instantiation in Section 5, where a classifier is trained for each category and each rule. When more complex hypotheses are involved, C_h classifiers can be trained separately for each relevant part of the hypothesis, using the rest for disambiguation.

A combination of the three model scores provides a final context matching score. In Section 6 we suggest a way to combine the actual classification scores as part of the integration of CP into TC, but other combinations are plausible. In particular, binary classifications (valid vs. invalid) may be used as filters. That is, the context is classified as valid only if all relevant models classify it as such.

5 A Self-supervised Context Model

We now turn to demonstrate how our classification-based scheme may be implemented. The model below is exemplified on Name-based TC, but may be applicable to other tasks, with few changes.

5.1 Training-set generation

Our implementation is self-supervised as we want to integrate it within the unsupervised TC setting. That is, the classifiers automatically obtain training examples for the classifying object (a category or a rule) without relying on labeled documents.

We obtain examples by querying the TC training corpus with automatically-generated search queries. The difficulty lies in correctly constructing queries that will retrieve documents representing either valid or invalid contexts for the classifying object. To this end, we retrieve examples through a gradual process in which the most accurate (least ambiguous) query

is used first and less accurate queries follow, until the designated number of examples is acquired.

5.1.1 Obtaining positive examples

To acquire positive training examples, we construct queries which are comprised of two main clauses. The first contains the *seeds*, terms which characterize the classifying object. Primarily, these are the category name or the LHS of the rule. The second consists of *context words* which are used when the seeds are polysemous, and are intended to assist disambiguation. When context words are used, at least one seed and at least one context word must be matched to retrieve a document. For example, given the highly ambiguous category name *space*, we first construct the query using only the monosemous term *outer space*; if the number of retrieved documents does not meet the requirement, a second query may be constructed: (“*outer space*” OR *space*) AND (*infinite* OR *science* OR ...).

To generate a rule classifier C_r , we retrieve positive examples as follows. If the LHS term is monosemous according to WordNet², we first query using this term alone (e.g. *decrypt*), and add its monosemous synonyms and hyponyms if more examples are required (e.g. *decrypt* OR *decode*). If the LHS is polysemous, we carry out Procedure 1. Intuitively, this procedure tries to minimize ambiguity by using monosemous terms as much as possible; when polysemous terms must be used, it tries to ensure there are monosemous terms to disambiguate them. Note that entailment directionality is maintained throughout the process, as seeds are only expanded with more specific (entailing) terms, while context words are only expanded with more general (entailed) terms.

Procedure 1 : Retrieval of C_r positive examples

Apply sequentially until sufficient examples are obtained:

- 1: Set the LHS as seed and the RHS’s monosemous synonyms, hypernyms and derivations as context words.
 - 2: Add monosemous synonyms and hyponyms of the LHS to the seeds.
 - 3: As in 2, but use polysemous terms as well.
 - 4: Add polysemous context words.
-

Positive examples for category classifiers (C_h) are obtained through a similar procedure as for rule clas-

²Terms not in WordNet are assumed monosemous.

sifiers. If the category is part of a hierarchy, we also use the name of the parent category (e.g. *sport* for *rec.sport.hockey*) as a context word.

5.1.2 Obtaining negative examples

Negative examples are even more challenging to acquire. In prior work negative examples were selected randomly (Kauchak and Barzilay, 2006; Connor and Roth, 2007). We follow this method, but also attempt to identify negative examples that are semantically similar to the positive ones in order to improve the discriminative power of the classifier (Smith and Eisner, 2005). We do that by applying a similar procedure which uses cohyponyms of the seeds, e.g. *baseball* for *hockey* or *islam* for *christianity*. Cohyponymy is a non-entailing relation; hence, by using it we expect to obtain semantically-related, yet invalid contexts. If not enough negative examples are retrieved using cohyponyms, we select the remaining required examples randomly.

As the distribution of positive and negative examples in the data is unknown, we set the ratio of negative to positive examples as a parameter of the model, as in (Bergsma et al., 2008).

5.1.3 Insufficient examples

When the number of training examples for a rule or a category is below a certain minimum, the resulting classifier is expected to be of poor quality. This usually happens for positive examples in any of the following two cases: (i) the seed is rare in the training set; (ii) the desired sense of the seed is rarely found in the training set, and unwanted senses were filtered by our retrieval query. For instance, *nazarene* does not occur at all in the training set, and the classifier corresponding to the rule '*nazarene* \Rightarrow *christian*' cannot be generated. On the other hand, *cone* does appear in the corpus but not in the astrophysical sense the rule '*cone* \Rightarrow *space*' refers to. In such cases we refrain from generating the classifier and use instead a default score of 0 for each classified object. The idea is that rare terms will also occur infrequently in the test set, while cases where the term is found in the corpus, but in a different sense than the desired one, will be blocked.

5.1.4 Feature extraction

We extract global and local lexical features that are standard in WSD work. Global features include all

the terms in the document or in the sentence in which a match was found. Local features are extracted around matches of seeds which comprised the query that retrieved the document. These features include the terms in a window around the match, and the noun, verb, adjective and adverb nearest to the match in either direction. For randomly sampled negative examples, where no matched query terms exist, we randomly select terms in the document as "matches" for local feature extraction. If more than one match of the same term is found in a document, we assume one-sense-per-discourse (Gale et al., 1992) and jointly extract features for all matches of the term.

5.2 Applying the classifiers

During inference, for each direct match in a document, the corresponding C_h is applied. For an indirect match, the respective C_r is also applied.

In addition, C_h is applied to the matched rules. Unlike t , a rule is not represented by a single text. Therefore, to test a rule's match with the category, we randomly sample from the training set documents containing the rule's LHS. We apply C_h to each sampled example and compute the ratio of positive classifications. The result is a score indicating the domain-specific probability of the rule to be applicable to the category, and may be interpreted as an in-domain *prior*. For instance, the rule '*check* \Rightarrow *hockey*' is assigned a score of 0.05, since the sense of *check* as a hockey defense technique is rare in the corpus. On the other hand, non ambiguous rules, e.g. '*warship* \Rightarrow *ship*' are assigned a high probability (1.0), and so are rules whose LHS is ambiguous but its dominant sense in the training corpus is the same one the rule refers to, e.g. '*margin* \Rightarrow *earnings*'(0.85).

We do not assign negative classifier scores to invalid matches but rather set them to zero instead. The reason is that an invalid context only indicates that the term cannot be used for entailing the category name, but not that the document itself is irrelevant.

6 CP for Text Categorization

CP may be employed in any inference-based task, but the integration with each task is somewhat different and needs to be specified. Below we present a methodology for integrating CP into Name-based Text Categorization.

As in (Barak et al., 2009) (*Barak09* below), we represent documents and categories by term-vectors in the following way: a document vector contains the document terms; a category vector contains two sets of terms: \mathcal{C} , the terms denoting the category name, and \mathcal{E} , their entailing terms. For example, *oil* is added to the vector of the category *crude* by the rule ‘*oil* \Rightarrow *crude*’ (i.e. *crude* \in \mathcal{C} and *oil* \in \mathcal{E}).

Barak09 assigned equal values of 1 to all vector entries. We suggest integrating a CP-based context model into TC by re-weighting the terms in the vectors, prior to determining the final document-category categorization score through vector similarity. Given a category c , with term vector C , and a document d with term vector D , the model re-weights vector entries of matching terms (i.e., terms in $C \cap D$), based on the validity of the context match. Valid matches should be assigned with higher scores than invalid ones, leading to higher overall vector similarity for documents with valid matches for the given category. Non-matching terms are ignored as their weights are canceled out in the subsequent vector product.

Specifically, the model assigns a new weight $w_D(u)$ to a matching term u in the document vector D based on the model’s assessment of: (a) $t - h$, the context match between the (match in the) document and the category; and (if an indirect match) (b) $t - r$, the context match between the document and the rule ‘ $u \Rightarrow c_i$ ’, where $c_i \in \mathcal{C}$. The model also sets a new weight $w_C(v)$ to a term v in the category vector C based on the context match for $r - h$, between the rule ‘ $v \Rightarrow c_j$ ’ ($c_j \in \mathcal{C}$) and the category. For instance, using our context matching scheme in TC, $w_D(u)$ is set to $C_h(u)$ or $\frac{C_h(u)+C_r(u)}{2}$ for direct and indirect matches, respectively; $w_C(v)$ is left as 1 if $v \in \mathcal{C}$ and set to $C_h(v)$ when $v \in \mathcal{E}$.

Barak09 assigned a single global context score to a document-category pair using the LSA representations of their vectors. In our approach, however, we consider the actual matches from the three different views, hence the re-weighting of the vector entries using three model scores.

7 Experimental Setting

7.1 Datasets and knowledge resources

Following (Gliozzo et al., 2009) and (Barak et al., 2009), we evaluated our method on two standard TC

datasets: *Reuters-10* and *20-Newsgroups*.

The *Reuters-10* (R10, for short) is a sub-corpus of the Reuters-21578 collection³, constructed from the ten most frequent categories in the Reuters taxonomy. We used the Apte split of the Reuters-21578 collection, often used in TC tasks. The top 10 categories include about 9,000 documents, split into training (70%) and test (30%) sets. The *20-Newsgroups* (20NG) corpus is a collection of newsgroup postings gathered from twenty different categories from the Usenet Newsgroups hierarchy⁴. We used the “by-date” version of the corpus, which contains approximately 20,000 documents partitioned (nearly) evenly across the categories and divided in advance to training (60%) and test (40%) sets.

As in (Gliozzo et al., 2009; Barak et al., 2009), we adjusted non-standard category names (e.g. *forsale* was renamed to *sale*) and manually specified for each category its relevant WordNet senses. The sense tagging properly defines the categories, and is expected to accompany such hypotheses. Other types of information may be used for this purpose, e.g. words from category descriptions, if such exist.

We applied standard preprocessing (sentence splitting, tokenization, lemmatization and part of speech tagging) to all documents in the datasets. All terms, including those denoting category names and rules, are represented by their lemma and part of speech.

As sources for lexical entailment rules we used WordNet 3.0 (synonyms, hyponyms, derivations and meronyms) and a Wikipedia-derived rule-base (Shnarch et al., 2009). Unlike *Barak09* we did not limit the rules extracted from WordNet to the most frequent senses and used all rule types from the Wikipedia-based resource.

7.2 Self-supervised model tuning

Tuning of the self-supervised context model’s parameters (number of training examples, negative to positive ratio, feature set and the way negative examples are obtained) was performed over development sets sampled from the training sets. Based on this tuning, some parameters varied between the datasets and between classifier types (C_h vs. C_r). For example,

³<http://kdd.ics.uci.edu/databases/reuters21578/reuters21578.html>

⁴<http://people.csail.mit.edu/jrennie/20Newsgroups/>

selection of negative examples based on cohyponyms was found useful for C_r classifiers in R10, while random examples were used in the rest of the cases.

We used SVM^{perf} (Joachims, 2006) with a linear kernel and binary feature weighting.

For querying the corpus we used the Lucene search engine⁵ in its default setting. Up to 150 positive examples were retrieved for each classifier, with 5 examples set as the required minimum. This resulted in generating 100% of the hypothesis classifiers for both datasets and 95% and 70% of the rule classifiers for R10 and 20NG, respectively.

We computed $C_h(r)$ scores based on up to 20 sampled instances. If less than 2 examples were found in the training set, we assigned an “unknown” context match probability of 0.5, since a rare LHS occurrence does not indicate anything about its meaning in the corpus. Such cases constituted 2% (R10) and 11% (20NG) of the utilized rules.

7.3 Baseline models

To provide a more meaningful comparison with prior work, we focus on the first unsupervised step in the typical Name-based TC flow, without the subsequent supervised training. Our goal is to improve the accuracy of this first step, and we therefore compare our context model’s performance to two unsupervised methods used by *Barak09*.

The first baseline, denoted $Barak_{no-cxt}$, is the cosine similarity score between the document and category vectors where all terms are equally weighted to a score of 1.⁶ This baseline shows the performance when no context model is employed.

The second baseline, denoted $Barak_{full}$, is a replication of the state of the art context model for Name-based TC. In this method, LSA vectors are constructed for a document by averaging the LSA vectors of its individual terms, and for a category by averaging the LSA vectors of the terms denoting its name. The categorization score of a document-category pair is set to be the product between the cosine similarity score of the LSA vectors and the score given by the above $Barak_{no-cxt}$ method. We note that LSA-based context models performed best also in (Gliozzo et al., 2009) and (Szpektor et al., 2008).

⁵<http://lucene.apache.org>

⁶Other attempted weighting schemes, such as tf-idf, did not yield better performance.

Model	<i>Reuters-10</i>			
	Accuracy	P	R	F ₁
$Barak_{no-cxt}$	73.2	63.6	77.0	69.7
$Barak_{full}$	76.3	68.0	79.2	73.2
<i>Class.-based</i>	79.3	71.8	83.6	77.2
Model	<i>20-Newsgroups</i>			
	Accuracy	P	R	F ₁
$Barak_{no-cxt}$	63.7	44.5	74.6	55.8
$Barak_{full}$	69.4	50.1	82.8	62.4
<i>Class.-based</i>	73.4	54.7	76.4	63.7

Table 1: Evaluation results.

All models were constructed based on the TC training sets, using no external corpora. The vocabulary consists of terms that appear more than once in the training set. The terms we consider include nouns, verbs, adjectives and adverbs, as well as nominal multi-word expressions.

8 Results and Analysis

Given a document, all categories for which a lexical match was found in the document are considered, and the document is classified to the highest scoring category. If all categories are assigned non-positive scores, the document is not assigned to any of them.

Based on this requirement that a document contains at least one match for the category, 4862 document-category pairs were considered for classification in R10 and 9955 pairs in 20NG. We evaluated our context model, as well as the baselines, based on the *accuracy* of these classifications, i.e. the percentage of correct decisions among the candidate document-category pairs. We also measured the models’ performance in terms of micro-averaged *precision* (P), *relative recall* (R) and F_1 . Like *Barak09*, recall is computed relative to the potential recall of the rule-set which provides the entailing terms.

Table 1 presents the evaluation results. As in *Barak09*, the LSA-based model outperforms the first baseline, supporting its usefulness as a context model. In both datasets our model outperformed the baselines in terms of accuracy. This result is statistically significant with $p < 0.01$ according to McNemar’s test (McNemar, 1947). Recall is lower for our model in 20NG but F_1 scores are higher for both datasets. These results indicate that the classification-based context model provides a favorable alternative to the

Removed	Reuters-10		20-Newsgroups	
	Accuracy	F ₁	Accuracy	F ₁
-	79.3	77.2	73.4	63.7
$C_h(t)$	76.2	72.3	71.9	61.0
$C_r(t)$	80.5	77.6	74.3	64.5
$C_h(r)$	78.4	75.7	73.1	63.4

Table 2: Ablation tests results.

state of the art LSA-based method.

Table 2 presents ablation tests of our model. In each test we measured the classification performance when one of the three classification scores is ignored. Clearly, $C_h(t)$ is the most beneficial component, and in general the category classifiers help improving overall performance. The limited performance of C_r may be related to higher ambiguity in rules relative to category names, resulting in noisier training data. In addition, the small size of the training set limits the number of training examples for rule classifiers. This problem affects C_r more than C_h since, by nature, the corpus includes more occurrences of category names. Still, C_r contributes to improved recall (this fact is not visible in Table 2).

The coverage of the utilized rule-set determines the maximal (absolute) recall that can be achieved by any model. With the rule-set we used in this experiment, the recall upper bound was 59.1% for R10 and 40.6% for 20NG. However, rule coverage affects precision as well: In many cases documents are assigned to incorrect categories because the correct category is not even a candidate as no entailing term was matched for it in the document. For instance, a document with the sentence “*For sale or trade!!! BMW R60US...*” was classified by our method to the category *forsale*, while its gold-standard category is *motorcycles*. Yet, none of the rules in our rule-set triggered *motorcycles* as a candidate category for this document. Ideally, a context model would rule out all incorrect candidate categories; in practice even a single low score for one of the competing categories results in a false positive error in such cases (in addition to the recall loss). To reduce these problems we intend to employ additional knowledge resources in future work.

Our algorithm for retrieving training examples turned out to be not sufficiently accurate, particularly for negative examples. This is a challenging task that

requires further research. Although useful for some classifier types, the use of cohyponyms may retrieve potentially positive examples as negative ones, since terms that are considered cohyponyms in WordNet are often perceived as near synonyms in common usage, e.g. *buyout* and *purchase* in the context of acquisitions. Likewise, using WordNet senses to determine ambiguity is also inaccurate. Rare or too fine-grained senses, common in WordNet, cause a term to be considered ambiguous, which in turn triggers the use of less accurate retrieval methods. For example, *auction* has a bridge-related WordNet sense which is irrelevant for our dataset, but made the term be considered ambiguous. This calls for development of other methods for determining word ambiguity, which consider the actual usage of terms in the domain rather than relying solely on WordNet.

9 Conclusions

In this paper we presented a generic classification-based scheme for comprehensively addressing context matching in textual inference scenarios. We presented a concrete implementation of the proposed scheme for Name-based TC, and showed how CP decisions can be integrated within the TC setting.

Utilizing classifiers for context matching offers several advantages. They naturally incorporate directionality and allow integrating various types of information, including ones not used in this work such as syntactic features. Our results indeed support this approach. Still, further research is required regarding issues raised by the use of multiple classifiers, scalability in particular.

Hypotheses in TC are available in advance. While also the case in other applications, it constitutes a practical challenge when hypotheses are given “on-line”, like Information Retrieval queries, since classifiers will have to be generated on the fly. We intend to address this issue in future work.

Lastly, we plan to apply the generic classification-based approach to address context matching in other inference-based applications.

Acknowledgments

This work was partially supported by the Israel Science Foundation grant 1112/08 and the NEGEV project (www.negev-initiative.org).

References

- Libby Barak, Ido Dagan, and Eyal Shnarch. 2009. Text Categorization from Category Name via Lexical Reference. In *HLT-NAACL (Short Papers)*.
- Shane Bergsma, Dekang Lin, and Randy Goebel. 2008. Discriminative Learning of Selectional Preference from Unlabeled Text. In *Proceedings of EMNLP*.
- David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2003. Latent Dirichlet Allocation. *Journal of Machine Learning Research*, 3:993–1022, March.
- Michael Connor and Dan Roth. 2007. Context Sensitive Paraphrasing with a Global Unsupervised Classifier. In *Proceedings of ECML*.
- Ido Dagan, Oren Glickman, Alfio Gliozzo, Efrat Marmorstein, and Carlo Strapparava. 2006. Direct Word Sense Matching for Lexical Substitution. In *Proceedings of COLING-ACL*.
- Ido Dagan, Bill Dolan, Bernardo Magnini, and Dan Roth. 2009. Recognizing Textual Entailment: Rational, Evaluation and Approaches. *Natural Language Engineering*, pages 15(4):1–17.
- Scott Deerwester, Scott Deerwester, Susan T. Dumais, George W. Furnas, Thomas K. Landauer, and Richard Harshman. 1990. Indexing by Latent Semantic Analysis. *Journal of the American Society for Information Science*, 41:391–407.
- Georgiana Dinu and Mirella Lapata. 2010. Topic Models for Meaning Similarity in Context. In *Proceedings of Coling 2010: Posters*.
- Doug Downey and Oren Etzioni. 2009. Look Ma, No Hands: Analyzing the Monotonic Feature Abstraction for Text Classification. In *Proceedings of NIPS*.
- Christiane Fellbaum, editor. 1998. *WordNet: An Electronic Lexical Database (Language, Speech, and Communication)*. The MIT Press.
- William A. Gale, Kenneth W. Church, and David Yarowsky. 1992. One Sense per Discourse. In *Proceedings of the workshop on Speech and Natural Language*.
- Alfio Gliozzo, Carlo Strapparava, and Ido Dagan. 2009. Improving Text Categorization Bootstrapping via Unsupervised Learning. *ACM Trans. Speech Lang. Process.*, 6:1:1–1:24, October.
- Sanda M. Harabagiu, Steven J. Maiorano, and Marius A. Paşca. 2003. Open-domain Textual Question Answering Techniques. *Natural Language Engineering*, 9:231–267, September.
- Thorsten Joachims. 2006. Training Linear SVMs in Linear Time. In *Proceedings of the ACM Conference on Knowledge Discovery and Data Mining (KDD)*.
- David Kauchak and Regina Barzilay. 2006. Paraphrasing for Automatic Evaluation. In *Proceedings of HLT-NAACL*.
- Lili Kotlerman, Ido Dagan, Idan Szpektor, and Maayan Zhitomirsky-Geffet. 2010. Directional Distributional Similarity for Lexical Inference. *Natural Language Engineering*, 16(4):359–389.
- Diana McCarthy and Roberto Navigli. 2009. The English Lexical Substitution Task. *Language Resources and Evaluation*, 43(2):139–159.
- Quinn McNemar. 1947. Note on the Sampling Error of the Difference between Correlated Proportions or Percentages. *Psychometrika*, 12(2):153–157, June.
- Roberto Navigli. 2009. Word Sense Disambiguation: A Survey. *ACM Computing Surveys*, 41(2):1–69.
- Patrick Pantel, Rahul Bhagat, Bonaventura Coppola, Timothy Chklovski, and Eduard Hovy. 2007. ISP: Learning Inferential Selectional Preferences. In *Proceedings of NAACL-HLT*.
- Marco Pennacchiotti, Roberto Basili, Diego De Cao, and Paolo Marocco. 2007. Learning Selectional Preferences for Entailment or Paraphrasing Rules. In *Proceedings of RANLP*.
- Alan Ritter, Mausam, and Oren Etzioni. 2010. A Latent Dirichlet Allocation Method for Selectional Preferences. In *Proceedings of ACL*.
- Eyal Shnarch, Libby Barak, and Ido Dagan. 2009. Extracting Lexical Reference Rules from Wikipedia. In *Proceedings of IJCNLP-ACL*.
- Noah A. Smith and Jason Eisner. 2005. Contrastive Estimation: Training Log-linear Models on Unlabeled Data. In *Proceedings of ACL*.
- Idan Szpektor, Ido Dagan, Roy Bar-Haim, and Jacob Goldberger. 2008. Contextual Preferences. In *Proceedings of ACL-08: HLT*.