# Context Sensitive Paraphrasing with a Global Unsupervised Classifier

Michael Connor and Dan Roth

Department of Computer Science
University of Illinois at Urbana-Champaign
connor2@uiuc.edu, danr@cs.uiuc.edu

**Abstract.** Lexical paraphrasing is an inherently context sensitive problem because a word's meaning depends on context. Most paraphrasing work finds patterns and templates that can replace other patterns or templates in **some** context, but we are attempting to make decisions for a **specific** context. In this paper we develop a global classifier that takes a word $v$ and its context, along with a candidate word $u$, and determines whether $u$ can replace $v$ in the given context while maintaining the original meaning.

We develop an unsupervised, bootstrapped, learning approach to this problem. Key to our approach is the use of a very large amount of unlabeled data to derive a reliable supervision signal that is then used to train a supervised learning algorithm. We demonstrate that our approach performs significantly better than state-of-the-art paraphrasing approaches, and generalizes well to unseen pairs of words.

## 1 Introduction

The problem of determining whether a text snippet can be written somewhat differently while maintaining its meaning has attracted a lot of attention from NLP researchers in recent years. It has clear applications in generation and summarization [1], automatic evaluation of machine translation and other machine generated text [2], and has been brought to focus recently by the body of work on Textual Entailment [3,4]. Consider, for example sentence 1(a) in Tab. 1. Does it have the meaning of sentence 1(b)?

**Table 1.** Context Sensitive Paraphrasing

| | | | | |
|---|---|---|---|---|
| 1(a) | The general *commanded* his troops | (b) | The general *spoke to* his troops | **Y** |
| 2(a) | The soloist *commanded* attention | (b) | The soloist *spoke to* attention | **N** |

There has been some work on generating rules or templates such as: 'X commanded Y' can be rewritten as 'X spoke to Y.' These rules do not specify when they should be applied or in what direction; they lack *context sensitivity.*

J.N. Kok et al. (Eds.): ECML 2007, LNAI 4701, pp. 104–115, 2007.

Consider sentence 2(a) in Tab.1. Is it still true that 'commanded' can be replaced with 'spoke to'? Alternatively one can ask: When can 'speak to' replace 'command' in the original sentence and not change the meaning of the sentence? This can be viewed as both a form of textual entailment and a weaker version of the common word sense disambiguation task [5]. If we knew the meaning of 'command' in the sentence and could compare it to the meaning of 'speak to,' we could decide if they are paraphrases here.

In this paper we develop a machine learning approach that learns *Context Sensitive Paraphrasing*: when one word can replace another in a given sentence without modifying its meaning. We address this task directly, without relying on any intermediate and possibly more difficult word sense assignment, and without attempting to compile a list of rules per word that say if and when another word or phrase can replace it. We will focus on verb paraphrasing; determining when one verb can replace another in a specific context. This limits our notion of context, but still provides a useful challenge because of the highly polysemous nature of verbs and their importance in encoding relations.

Our machine learning approach gives us a global classifier that is able to tackle this important and difficult task of context sensitive paraphrasing. The key difficulty from the machine learning perspective is how to derive a reliable supervision signal. By using domain knowledge and a large amount of data we are able to collect statistics over individual words to induce a reliable surrogate supervisory signal.

### 1.1 Related Work

One related line of inquiry is on *paraphrase generation*, namely given a sentence or phrase, generate paraphrases of that phrase that have the same or entailed meaning in *some* context. Often this would be represented as a fixed set of rules. Building these systems could require parallel or comparable corpora [6,7] which ties the systems to very specific topics. Other systems extract rules from dependency trees built over a large single corpora or the web [8,9,10]. These create more general rules, but they only say that a context *exists* where one phrase can replace another. They do not indicate *when* a rule can be applied, as we do.

A second line of work has approached the single word paraphrasing task as a sense disambiguation task. Kauchak and Barzilay [2] determine when one word can fit in a given context as part of a machine translation evaluation system by generating a separate simple classifier for every word. Dagan et al. [11] puts forward an implicit sense disambiguation task where two words are considered paraphrases if they share the specific sense for a given context. These approaches rely on a more standard word-sense word-expert approach with one classifier per word in the first case, or even one classifier per pair of words in the second. These approaches cannot classify unseen words; if they do encounter a new word they need to generate new training data and classifiers on the fly leading to scaling issues as the vocabulary increases.

Previous work employs either a single supervised rule (either through hand tuning or supervised signal present in parallel corpora) or a set of simple per-word classifiers. Our approach combines aspects of both so that we can train a supervised global classifier using a supervision signal induced from unsupervised per-word classifiers. We do not have to hand tune global classifier parameters, but are also able to apply our function to words outside of its original training set.

In Sec. 2 we present our learning problem and overall plan for context dependence. In Sec. 3 we define how we use context overlap to form rules for paraphrase decisions. A single global classifier encodes these rules for all words in Sec. 4, which is trained with bootstrapped examples from unsupervised local classifiers (sec. 4.2). Experimental results for both untrained context dependent rules and the single global classifier are presented in Sec. 5.

## 2 Formal Model

Formally, we are learning in an unsupervised way a binary function $f(S, v, u)$. $f$ is given a sentence $S$, a word or phrase $v$ in $S$, and a second word or phrase $u$. $f$ returns 1 iff replacing $v$ in $S$ with $u$ keeps the same or entailed meaning.

Looking at Tab. 1, if we set S to 1(a), $v$ to 'command' and $u$ to 'speak to', then $f(S, v, u)$ should return 1 indicating that the sentence in 1(b) is a correct paraphrase. However if $u$ and $v$ are kept the same but $S$ changed to 2(a) then $f(S, v, u)$ should return a 0. 2(b) is not a correct paraphrase; 'command' does not replace 'speak to' *in this context* $S$.

### 2.1 Definition of Context

Much of our technique relies heavily on our notion of context. Context around a word can be defined as bag of words or collocations, or can be derived from parsing information. We view each of these as different aspects of the same context, with each aspect providing different amounts of information and different drawbacks (sparsity, distribution, etc) for the final decision. For most of this work we use dependency links from Minipar [12] dependency parser to define the local context of a word in a sentence, similar to what is used by DIRT and TEASE [10,9]. Throughout this paper each algorithm will make use of a specific aspect of context we'll call $c$ which can be either subject and object of the verb, named entities that appear as subject or object, all dependency links connected to the target, all noun phrases in sentences containing the target, or all of the above. One of the goals of our system is to intelligently combine these sources of information about context in a single classifier framework.

### 2.2 Modeling Context Sensitive Paraphrasing

We consider the question of whether $u$ can replace $v$ in $S$ as composed of two simpler questions: (1) can $u$ replace $v$ in *some* context and (2) can $u$ fit in the

context of $S$. In the first case, $u$ must share some meaning with $v$. If the second condition also holds then we can theorize that $u$ will get this meaning of $v$ in $S$. We give the question 'can $u$ replace $v$' context dependence by determining if $u$ can belong in $v$'s context in $S$.

We approach the first question using the same distributional similarity assumptions that others use, namely that we can determine whether $u$ can replace $v$ in some context by seeing what contexts they both appear in in a large corpus. To use this idea to answer the second question we restrict the context comparison to those similar to the given context $S$. If two words are seen in the same contexts then they may be paraphrases, but if they are seen in many of the same contexts that are also similar to the given context then they may be paraphrases in the local context.

## 3 Statistical Paraphrase Decisions

Most word paraphrasing systems work on similar principles: they find contexts that both words, $u$ and $v$, appear in. Given enough such contexts then $u$ and $v$ are accepted as replacements. In the next two sections we define how we encode such rules and then add context sensitivity.

### 3.1 Context Insensitive Decisions

Given $u$ and $v$ the first goal is to find the sets of contexts that $u$ and $v$ have been seen with in a large corpora. $S_v^c$ is the set of type $c$ contextual features of contexts that $v$ appears in and likewise for $S_u^c$. By looking at the overlap of these two sets we can see what sort of contexts both $u$ and $v$ appear in. Looking at the example in Tab. 2, and more specifically the $S_v \cap S_u$ row, we can see some features of contexts that 'suggest' and 'propose' appear in compared to those that 'suggest' and 'indicate' both appear in. With $u =$ 'propose', the shared contexts are often related to political activity, with some politician suggesting or proposing an action or plan. On the other hand, in the contexts with $u =$ 'indicate', the sense of both words is that the subject is a signal for the object.

To make a decision based on the amount of contextual overlap we set a threshold for the overlap coefficient score:

$$Score_c(u, v) = |S_v^c \cap S_u^c| / min(|S_v^c|, |S_u^c|)$$

This score represents what proportion of contexts the more specific (seen in fewer contexts) word shares with the other. The specific threshold that we select differs between notions of context, but is constant across words. Given this threshold value we now have a simple classifier that can determine if any $u$ can replace any $v$ in some context. To be able to apply this classifier to a $u, v$ pair we only need to find their $S_u^c$ and $S_v^c$ sets by looking in a large body of text for occurrences of $u$ and $v$.

**Table 2.** Similar Context Overlap Example. Given the above sentence the goal is to determine whether 'propose' and/or 'indicate' can replace 'suggest' and retain the original meaning. The aspect of context used is the subject and object of target verb and $c$ notation has been left out. Note that each set only shows a subset of its members, unless otherwise noted.

| Sentence | | Marshall Formby of Plainview *suggested* a plan to fill by appointment future vacancies in the Legislature and Congress, eliminating the need for special elections. | |
|---|---|---|---|
| Query | | $v$ = suggest; $u$ = propose | $v$ = suggest; $u$ = indicate |
| CIP | $S_v \cap S_u$ | obj:alternative, subj:Clinton, obj:compromise, obj:solution | subj:presence, subj:history, obj:possibility, obj:need |
| CSP | Local Context | obj:plan, subj:NE:PER | |
| | $V_S$ | foil, lay out, debate, consider, endorse, propose, discuss, change, disrupt, unveil, detail, disclose | |
| | $S_S$ | obj:bid, obj:legislation, obj:approach, obj:initiative, subj:pronoun+obj:program, subj:George W. Bush, obj:project | |
| | $S_{S,v} \cap S_{S,u}$ | subj:George W. Bush, obj:way, obj:policy, obj:legislation, obj:program, obj:idea | subj:George W. Bush (only one) |

### 3.2 Adding Context Sensitive Decisions

The goal of this step of the algorithm is to restrict the set of contexts used to find overlap of $u$ and $v$ so that the overlap has some relation to the local context $S$. This process is described in more detail in Fig. 1(b). The system identifies contexts similar to $S$ and sees if $u$ and $v$ overlap in these. By restricting the set of contexts that we use to find overlap of $u$ and $v$ as in Sec. 3.1 we are attempting to see if the usage of $u$ and $v$ overlap in a specific sense which relates to $S$.

We consider a context similar to the current one if they both appear with similar verbs. If two contexts can appear with the same set of verbs then the contexts convey the same set of possible meanings. We start by finding a set of verbs typical to the given context. Currently we use a very simple definition of the local context of $v$ in $S$: the subject and object of $v$, if those exist. In many cases the subject and object restrict what verbs can be used and indicate a specific meaning for those verbs. For our example sentence in Tab. 2, the subject is a named entity which we indicate as NE:PER and the object is 'plan'. The $V_S$ of this context are verbs that indicate actions someone does with a plan, including synonyms for creating, breaking or presenting. Verb $x \in V_S$ if $x$ has been seen before with the subject and object of $v$, not necessarily all in the same sentence.

We now look for context features that these verbs appear in and are specific to the meaning of this set of verbs and context. $S_S^c$ are those context features of type $c$ that some percentage of the typical verbs are seen with (in experiments this percentage is empirically set to 25%). In Tab. 2 $S_S$ shows a set of contextual features (subjects and objects) that are similar to the given: objects that are

treated like plans and subjects that are people that do stuff with plans, which appears to be mostly politicians in our dataset.

So given a set of contextual features, $S^c_S$, from contexts similar to $S$, and contexts $S^c_u$ for which $u$ appear and $S^c_v$ for which $v$ appear, we want to restrict our attention to the set of contexts that both $u$ and $v$ appear in and that are similar to $S$: $S^c_S \cap S^c_v \cap S^c_u$. If we focus on the sets $S^c_{S,v} = S^c_S \cap S^c_v$ and $S^c_{S,u} = S^c_S \cap S^c_u$ then the intersection of these two sets is the same as the intersection of all three, and we can use the overlap score as we used above:

$$Score_c(S, v, u) = |S^c_{S,v} \cap S^c_{S,u}| / min(|S^c_{S,v}|, |S^c_{S,u}|)$$

This time we know the contexts of $v$ and $u$ are related to the given context, giving this overlap measure context sensitivity.

If we return to our example in Tab. 2 and look at the $S_{S,v} \cap S_{S,u}$ row we can see that 'propose' and 'indicate' have separated themselves for this specific context. The similar contexts of 'indicate' and 'suggest' have a very low intersection with the contexts similar to the given local context since they are used with a different sense of 'suggest'. On the other hand 'propose' can be used in the same sense as 'suggest' in this sentence, so it has a higher overlap with similar contexts. This higher overlap indicates that for this context, 'propose' can replace 'suggest', but 'indicate' cannot.

Algorithm UV-CIP
INPUT: $(S, v, u)$
OUTPUT: $f(S, v, u) \in \{0, 1\}$

Generate $S^c_u$ and $S^c_v$:
Depends on notion of context
Compute $\text{Score}_c(u, v)$
Based on $S^c_u \cap S^c_v$
Return $\text{Score}_c(u, v) > \text{Threshold}_c$

(a) Context Insensitive Paraphrase Rule

Algorithm SVU-CSP
INPUT: $(S, v, u)$
OUTPUT: $f(S, v, u) \in \{0, 1\}$

Generate $S^c_u$ and $S^c_v$
Find contexts similar to $S$: $S^c_S$
Find local context of $v$ in $S$
Determine $V_S$
Determine $S^c_S$
Find $S^c_{S,u} = S^c_S \cap S^c_u$
Find $S^c_{S,v} = S^c_S \cap S^c_v$
Determine $\text{Score}_c(S, v, u)$
Based on $S^c_{S,u} \cap S^c_{S,v}$
Return $\text{Score}_c(S, v, u) > \text{Threshold}_c$

(b) Context Sensitive Paraphrase Rule

**Fig. 1.** Statistics Based Context Sensitive Paraphrasing. Contextual overlap depends on a specific aspect of context $c$, be it the subject and object, bag of words, named entities, or all available.

## 4 Global Context Sensitive Paraphrase Classifier

Each paraphrasing rule above (context sensitive and insensitive for each contextual aspect) forms a simple classifier that can be applied to any pair of words and only has one parameter to tune, the threshold. We form our single global classifier as a linear combination of these different paraphrasing rules. We can use the overlap scores for each contextual aspect that the separate rules produce as

features for the global classifier, and thus leverage the powers of multiple notions of context and both context sensitive and insensitive information at the same time. To create training data for this single global classifier we use large amounts of untagged text to train local per-word classifiers that are able to identify new contexts where a specific word can act as a paraphrase. These local classifiers are used to tag new data to be used to train the single global classifier.

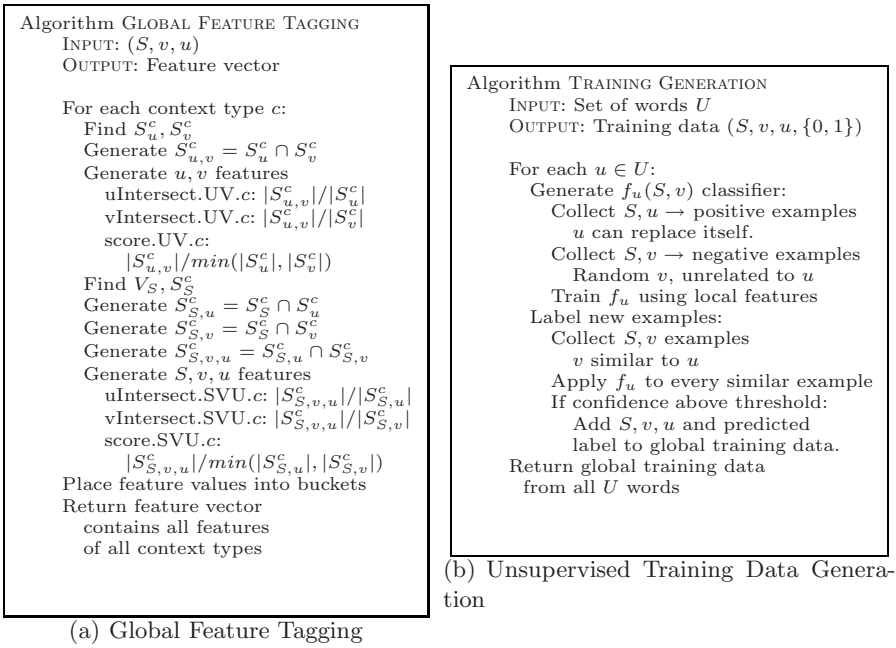
### 4.1 Shared Context Features

The flexibility of a classifier architecture allows us to incorporate additional features other than just the raw $\text{Score}_c(u, v)$ and $\text{Score}_c(S, v, u)$. For each context type $c$ we still compile the sets of similar contexts $S^c_u$, $S^c_v$, $S^c_{S,u}$, and $S^c_{S,v}$ but now we use the size of the overlaps as features. Fig. 2(a) describes this process further. We create three features for context insensitive overlap (UV features: only depend on $u$ and $v$), and three for context sensitive (SUV features: depend on $S$, $u$ and $v$). By including context insensitive features the classifier can still rely on a context independent decision when encountering a rare or malformed local context. For both UV and SUV feature types we create three features which show the direction of overlap: score, uIntersect, and vIntersect. The score feature is the same as the $\text{Score}_c(u, v)$ for UV and $\text{Score}_c(S, u, v)$ for SUV used in the statistics based rules above. The uIntersect and vIntersect actually give directionality to the feature representation. If uIntersect is close to 1 then we know that $u$ primarily appears in contexts that $v$ appears in, and thus $u$ may be a specialization of $v$, and if this is an SUV feature then we know the directionality holds in contexts similar to $S$. The classifier can now learn that its possibly more important for $v$ to specialize $u$, and can say yes for replacing A by B in a given context, but not B by A.

For any new word $w$, all we need to know about $w$ to generate features are $S^c_w$ for every $c$; contextual features of contexts that $w$ is seen with. If we can find contexts that $w$ appears in in our large corpora then we can create the $S^c_w$ sets and use these to compute overlap with other words, the only features that our classifier relies on. A separate classifier does not need to be trained on contexts that $w$ appears in, we can rely on our global classifier even if it never saw an example of $w$ during training.

### 4.2 Unsupervised Training: Bootstrapping Local Classifiers

The single global classifier requires training to be able to determine the importance of each context type in the global paraphrasing decision. To generate tagged $S, v, u$ examples for the single global classifier we employ a set of bootstrapping local classifiers similar to Kauchak and Barzilay[2] or Context-Sensitive Spelling Correction[13]. Each classifier learns what contexts one specific word can appear in. Once we have these classifiers we generate candidate examples to train the global classifier and use the local classifiers to filter and label confident examples. These examples are then tagged with global features and used to train our global binary classifier. This process is detailed in Fig. 2(b).




(a) Global Feature Tagging

(b) Unsupervised Training Data Generation

**Fig. 2.** Procedures for training data generation and tagging. Although our final global classifier is a supervised learner, we use multiple unsupervised local classifiers to generate the set of training examples. The local classifiers work on local features while the global classifier looks at features related to the contextual overlap of its input.

The key fact that allows us to build classifiers this way and train a single global classifier is that we can extract large amounts of training examples for such local classifiers from free, untagged text.

To learn what contexts a specific word $u$ can belong in, we use trivial positive and negative examples that can be collected from flat text. Trivial positive examples are of $u$ replacing itself, and trivial negatives are of $u$ replacing a random $x$ that is unrelated to $u$ (not connected in WordNet). The positive examples identify contexts that $u$ is known to fit in, and the negative examples represent contexts that $u$ likely does not, so we can eliminate irrelevant contextual features. We encode this knowledge of what contexts $u$ does and does not belong in in local $f_u$ binary classifiers.

For each word $u$ in our training set of words, we create an $f_u(S, v)$ classifier: the input is the context of $v$ in $S$ and the output is 1 or 0, depending if $u$ can replace $v$ in $S$ or not. Notice this is a local form of the global $f(S, v, u)$ function, if it were possible to train local classifiers for every possible $u$. The local $f_u$ use their implicit knowledge about the given $u$ to tag interesting examples such that the global classifier can extract patterns that hold for all words.

Our feature representation for the context of $v$ in $S$ that $f_u$ uses includes bag of words, collocations, nearest noun and verb to the left and right of target,

and named dependency links of length 1 and 2 from target from Minipar. These features are richer than the simple surrounding word and collocation features employed in [2], which allow us to get the same accuracy for local $f_u$ classifiers using many fewer examples (their local classifiers used on average 200k training examples, ours at most 30k).

Once we have trained local bootstrapping classifiers we use them to tag examples of $S, v$, where $v$ is similar to $u$ (from Lin's similarity list [14]). If the local classifiers confidently label an example, that example is added with its label to the global training set. The confidence is tuned to less than 5% error on a development set for each $f_u$. We generated 230 local classifiers, where the seed $U$ set was selected so that about half the words were seen in our test data, and half random verbs with at least 1000 occurrences in our text corpora. These local classifiers confidently tagged over 400k examples for our global classifier.

## 5 Experimental Results

### 5.1 Methodology

As a large source of text we used the 2000 New York Times section of the AQUAINT corpus. Each sentence was tagged with part of speech and named entities then parsed with Minipar. To implement both the $f_u$ and the single global classifier we used the SNoW learning architecture[15] with perceptron update rule.

Our test set goal was to select one example sentence representing each sense (and thus a different word to replace) for each of a set of verbs so that we can highlight the context sensitivity of paraphrasing. We started with an initial random selection of polysemous verbs that occur in WordNet 2.1 sense tagged data (Semcor)[16]. For each verb we selected a possible synonym for each of a coarse sense mapping (Navigli's mappings of WordNet 2.1 to coincide with ODE entries [17]) since the coarse word senses provide clearer distinctions between meanings of words than regular WordNet. We then selected one representative sentence where each coarse synonym could replace the target verb. For every $S, v$ sentence the possible $u$ were exactly the coarse synonyms for $v$, the intent being that exactly one of them would be replaceable for each sense of $v$, although this was not always the case. Two humans (native English speaking graduate students) annotated each $S, v, u$ example for whether $u$ could replace $v$ or not in $S$. The interannotator agreement was over 83% of instances, with a kappa of 0.62, corresponding to substantial agreement[18]. This kappa is comparable to our previous tagging efforts and others reported in similar efforts [19]. Overall our test set has 721 $S, v, u$ examples with 57 unique $v$ verbs and 162 unique $u$.

### 5.2 Results

The results of the global classifier on the test set is shown in Fig. 3. The varying precision/recall points were achieved by setting the SNoW confidence threshold for the global classifier and setting the score threshold for the statistical rules. As

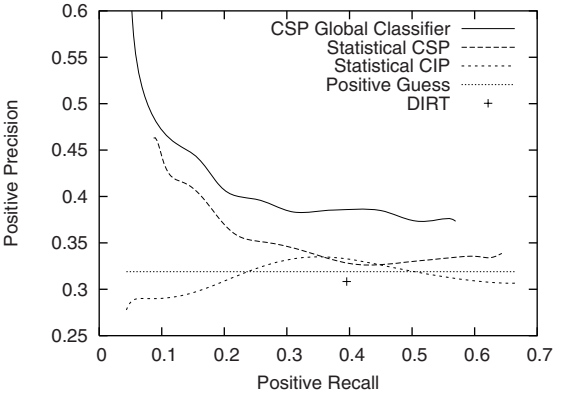
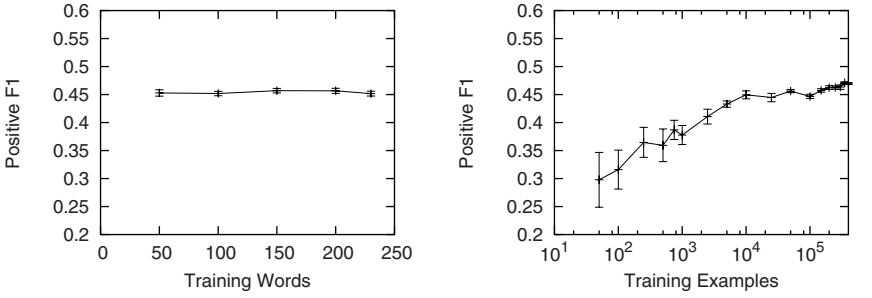



**Fig. 3.** Positive precision/recall curve for our global CSP classifier compared to statistical context sensitive and insensitive paraphrasing rules using all contextual features. Also shown is the single point of precision/recall given by current top of the line paraphrasing system DIRT on our test set. The positive guess line illustrates the precision of always guessing yes for every example: 32% of test cases are positive.

we can see the single global classifier is able to exploit both the $u, v$ and $S, v, u$ rules and different definitions of context to improve both recall and precision. As an additional comparison we include the results if we used slightly modified DIRT rules to make paraphrase judgments. Each DIRT rule specifies a dependency pattern that can be rewritten as another dependency pattern. For our verb paraphrase task we restricted the rules to those that could be interpreted as single verb rewrite rule where each pattern is a verb with two dependency links coming off of it. In a similar setup, recent experiments have shown that DIRT is the top of the line for verb paraphrasing [19].

Both classifier and statistical based rules performed better on this test set than DIRT probably because we do not rely on a fixed set of patterns (that are generated on a separate corpus). Instead we use classifiers that can be instantiated with any word encountered in the test set. During training our global classifier only saw examples generated for a subset of the words that appear in testing. The classifier can still apply to any $S, v, u$ example as long as we can collect contexts that $v$ and $u$ have been seen in. Any per-word classifier approach such as Kauchak and Barzilay could not have handled unseen examples such as these, they would need to collect examples of $u$ and generate a new classifier.

To test the ability of our classifier to learn a global hypothesis that applies across words we tested the performance of the classifier when it was trained with unsupervised data generated for a varying number of words while keeping the final number of training examples fixed. If the classifier can learn a hypothesis just as well by looking at statistics for 50 words vs. 250 then its hypothesis is not dependent on the specific words used in training, but on global statistics that exist for paraphrasing. Fig. 4(a) shows that the number of words used to




(a) Number of Words (b) Number of Examples

**Fig. 4. Generalization to unseen words:** Plot (a) shows that performance on the test data does not depend on the number of words seen in training (recall: training set is created w/o supervision). What does affect performance is the number of training examples generated (plot (b)). Each point represents the mean positive F1 over 20 different random resamplings of the training set. This plot represents one point selected from the precision/recall curve above.

generate training data has little effect on the positive F1 of the classifier, the same rules learned on 50 words is learned on 200 and beyond. On the other hand, if we look at the number of examples used to train the classifier we do see improvement. Fig 4(b) shows the results of varying the number of examples used to train the classifier, but keeping the number of words these examples are drawn from fixed. If we are only able to create accurate bootstrapping classifiers for a small set of words, this should still prove adequate to generate data to train the global classifier, as long as we can generate a lot of it.

## 6 Conclusion and Future Work

In this project we presented an approach to adding context sensitivity to a word paraphrasing system. By only looking for distributional similarity over contexts similar to the given sentence we are able to decide if one verb can replace another in the given context. Further we incorporate this approach into a classifier architecture that is able to successfully combine multiple definitions of context and context sensitive and insensitive information into a unified whole. Our machine learning approach allowed us to leverage a large amount of data regarding the local statistics of word occurrences to generate training data for a traditionally supervised global classifier in an unsupervised manner.

Our experiments indicate that it is important to have as much data per word as possible, both in terms of representation and training, so we plan to expand our knowledge sources. The eventual goal for the system is to incorporate it into a full textual entailment system and see if this context sensitive paraphrasing can benefit a larger NLP task.

## Acknowledgments

We would like to thank those who have given us comments throughout this project, especially Idan Szpektor, Kevin Small, and anonymous reviewers. This research is supported by NSF grants BCS-0620257 and ITR-IIS-0428472.

## References

1. Barzilay, R., Lee, L.: Catching the drift: Probabilistic content models, with applications to generation and summarization. In: Proceedings HLT-NAACL (2004)
2. Kauchak, D., Barzilay, R.: Paraphrasing for automatic evaluation. In: Proceedings of HLT-NAACL 2006 (2006)
3. Dagan, I., Glickman, O., Magnini, B.: The pascal recognizing textual entailment challenge. In: Proceedings of the PASCAL Challenges Workshop on Recognizing Textual Entailment (2005)
4. de Salvo Braz, R., Girju, R., Punyakanok, V., Roth, D., Sammons, M.: An inference model for semantic entailment in natural language. In: Proceedings of the National Conference on Artificial Intelligence (AAAI), pp. 1678–1679 (2005)
5. Ide, N., Veronis, J.: Word sense disambiguation: The state of the art. Computational Linguistics (1998)
6. Barzilay, R., Lee, L.: Learning to paraphrase: An unsupervised approach using multiple-sequence alignment. In: Proceedings HLT-NAACL, pp. 16–23 (2003)
7. Barzilay, R., McKeown, K.: Extracing paraphrases from a parallel corpus. In: Proceedings ACL-01 (2004)
8. Glickman, O., Dagan, I.: Identifying lexical paraphrases from a single corpus: A case study for verbs. In: Recent Advantages in Natural Language Processing (RANLP-03) (2003)
9. Szpektor, I., Tanev, H., Dagan, I., Coppola, B.: Scaling web-based acquisition of entailment relations. In: Proceedings of EMNLP 2004 (2004)
10. Lin, D., Pantel, P.: Discovery of inference rules for question answering. Natural Language Engineering 7(4), 343–360 (2001)
11. Dagan, I., Glickman, O., Gliozzo, A., Marmorshtein, E., Strapparava, C.: Direct word sense matching for lexical substitution. In: Proceedings ACL-06, pp. 449–456 (2007)
12. Lin, D.: Principal-based parsing without overgeneration. In: Proceedings of ACL-93, pp. 112–120 (1993)
13. Golding, A.R., Roth, D.: A Winnow based approach to context-sensitive spelling correction. Machine Learning 34(1-3), 107–130 (1999)
14. Lin, D.: Automatic retrieval and clustering of similar words. In: COLING-ACL-98 (1998)
15. Carlson, A., Cumby, C., Rosen, J., Roth, D.: The SNoW learning architecture. Technical Report UIUCDCS-R-99-2101, UIUC Computer Science Department (May 1999)
16. Fellbaum, C.: Wordnet: An Electronic Lexical Database. Bradford Books (1998)
17. Navigli, R.: Meaningful clustering of senses helps boost word sense disambiguation performance. In: Proceedings of COLING-ACL 2006 (2006)
18. Landis, J., Koch, G.: The measurement of observer agreement for categorical data. In: Biometrics (1977)
19. Szpektor, I., Shnarch, E., Dagan, I.: Instance-based evaluation of entailment rule acquisition. In: Proceedings of ACL 2007 (2007)