In Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics (ACL 2010)

Inducing Domain-specific Semantic Class Taggers from (Almost) Nothing

Ruihong Huang and Ellen Riloff
School of Computing
University of Utah
Salt Lake City, UT 84112
{huangrh,riloff}@cs.utah.edu

Abstract

This research explores the idea of inducing domain-specific semantic class taggers using only a domain-specific text collection and seed words. The learning process begins by inducing a classifier that only has access to contextual features, forcing it to generalize beyond the seeds. The contextual classifier then labels new instances, to expand and diversify the training set. Next, a *cross-category bootstrapping process* simultaneously trains a suite of classifiers for multiple semantic classes. The positive instances for one class are used as negative instances for the others in an iterative bootstrapping cycle. We also explore a one-semantic-class-per-discourse heuristic, and use the classifiers to dynamically create semantic features. We evaluate our approach by inducing six semantic taggers from a collection of veterinary medicine message board posts.

1 Introduction

The goal of our research is to create semantic class taggers that can assign a semantic class label to every noun phrase in a sentence. For example, consider the sentence: *"The lab mix was diagnosed with parvo and given abx"*. A semantic tagger should identify the *"the lab mix"* as an ANIMAL, *"parvo"* as a DISEASE, and *"abx"* (antibiotics) as a DRUG. Accurate semantic tagging could be beneficial for many NLP tasks, including coreference resolution and word sense disambiguation, and many NLP applications, such as event extraction systems and question answering technology.

Semantic class tagging has been the subject of previous research, primarily under the guises of *named entity recognition* (NER) and *mention detection*. Named entity recognizers perform semantic tagging on proper name noun phrases, and sometimes temporal and numeric expressions as well. The *mention detection* task was introduced in recent ACE evaluations (e.g., (ACE, 2007; ACE, 2008)) and requires systems to identify all noun phrases (proper names, nominals, and pronouns) that correspond to 5-7 semantic classes.

Despite widespread interest in semantic tagging, nearly all semantic taggers for comprehensive NP tagging still rely on supervised learning, which requires annotated data for training. A few annotated corpora exist, but they are relatively small and most were developed for broad-coverage NLP. Many domains, however, are replete with specialized terminology and jargon that cannot be adequately handled by general-purpose systems. Domains such as biology, medicine, and law are teeming with specialized vocabulary that necessitates training on domain-specific corpora.

Our research explores the idea of inducing domain-specific semantic taggers using a small set of seed words as the only form of human supervision. Given an (unannotated) collection of domain-specific text, we automatically generate training instances by labelling every instance of a seed word with its designated semantic class. We then train a classifier to do semantic tagging using these seed-based annotations, using bootstrapping to iteratively improve performance.

On the surface, this approach appears to be a contradiction. The classifier must learn how to assign different semantic tags to different instances of the same word based on context (e.g., *"lab"* may refer to an animal in one context but a laboratory in another). And yet, we plan to train the classifier using stand-alone seed words, making the assumption that every instance of the seed belongs to the same semantic class. We resolve this apparent contradiction by using semantically unambiguous seeds and by introducing an initial context-only training phase before bootstrapping begins. First, we train a *strictly contextual* classifier that only has access to contextual features and cannot see the seed. Then we apply the classifier to the corpus to automatically label new instances, and combine these new instances with the seed-based instances. This process expands and diversifies the training set to fuel subsequent bootstrapping.

Another challenge is that we want to use a *small* set of seeds to minimize the amount of human effort, and then use bootstrapping to fully exploit the domain-specific corpus. Iterative self-training, however, often has difficulty sustaining momentum or it succumbs to semantic drift (Komachi et al., 2008; McIntosh and Curran, 2009). To address these issues, we simultaneously induce a suite of classifiers for multiple semantic categories, using the positive instances of one semantic category as negative instances for the others. As bootstrapping progresses, the classifiers gradually improve themselves, and each other, over many iterations. We also explore a *one-semantic-class-per-discourse* (OSCPD) heuristic that infuses the learning process with fresh training instances, which may be substantially different from the ones seen previously, and we use the labels produced by the classifiers to dynamically create semantic features.

We evaluate our approach by creating six semantic taggers using a collection of message board posts in the domain of veterinary medicine. Our results show this approach produces high-quality semantic taggers after a sustained bootstrapping cycle that maintains good precision while steadily increasing recall over many iterations.

2 Related Work

Semantic class tagging is most closely related to *named entity recognition* (NER), *mention detection*, and *semantic lexicon induction*. NER systems (e.g., (Bikel et al., 1997; Collins and Singer, 1999; Cucerzan and Yarowsky, 1999; Fleischman and Hovy, 2002) identify proper named entities, such as people, organizations, and locations. Several bootstrapping methods for NER have been previously developed (e.g., (Collins and Singer, 1999; Niu et al., 2003)). NER systems, however, do not identify nominal NP instances (e.g., “a software manufacturer” or “the beach”), or handle semantic classes that are not associated with proper named entities (e.g., symptoms).[1] ACE mention detection systems (e.g., see (ACE, 2005; ACE, 2007; ACE, 2008)) require tagging of NPs that correspond to 5-7 general semantic classes. These systems are typically trained with supervised learning using annotated corpora, although techniques have been developed to use resources for one language to train systems for different languages (e.g., (Zitouni and Florian, 2009)).

Another line of relevant work is *semantic class induction* (e.g., (Riloff and Shepherd, 1997; Roark and Charniak, 1998; Thelen and Riloff, 2002; Ng, 2007; McIntosh and Curran, 2009), where the goal is to induce a stand-alone dictionary of words with semantic class labels. These techniques are often designed to learn specialized terminology from unannotated domain-specific texts via bootstrapping. Our work, however, focuses on classification of NP *instances* in context, so the same phrase may be assigned to different semantic classes in different contexts. Consequently, our classifier can also assign semantic class labels to pronouns.

There has also been work on extracting semantically related terms or category members from the Web (e.g., (Paşca, 2004; Etzioni et al., 2005; Kozareva et al., 2008; Carlson et al., 2009)). These techniques harvest broad-coverage semantic information from the Web using patterns and statistics, typically for the purpose of knowledge acquisition. Importantly, our goal is to classify instances in context, rather than generate lists of terms. In addition, the goal of our research is to learn specialized terms and jargon that may not be common on the Web, as well as domain-specific usages that may differ from the norm (e.g., *“mix”* and *“lab”* are usually ANIMALS in our domain).

The idea of simulataneously learning multiple semantic categories to prevent semantic drift has been explored for other tasks, such as semantic lexicon induction (Thelen and Riloff, 2002; McIntosh and Curran, 2009) and pattern learning (Yangarber, 2003). Our bootstrapping model can be viewed as a form of self-training (e.g., (Ng and Cardie, 2003; Mihalcea, 2004; McClosky et al., 2006)), and cross-category training is similar in spirit to co-training (e.g., (Blum and Mitchell, 1998; Collins and Singer, 1999; Riloff and Jones, 1999; Mueller et al., 2002; Phillips and Riloff, 2002)). But, importantly, our classifiers all use the same feature set so they do not represent independent views of the data. They do, however, offer slightly different perspectives because each is at-

[1] Some NER systems also handle specialized constructs such as dates and monetary amounts.

tempting to recognize a different semantic class.

3 Bootstrapping an Instance-based Semantic Class Tagger from Seeds

3.1 Motivation

Our goal is to create a bootstrapping model that can rapidly create semantic class taggers using just a small set of seed words and an unannotated domain-specific corpus. Our motivation comes from specialized domains that cannot be adequately handled by general-purpose NLP systems. As an example of such a domain, we have been working with a collection of message board posts in the field of veterinary medicine. Given a document, we want a semantic class tagger to label every NP with a semantic category, for example:

> [A 14yo doxy]$_{ANIMAL}$ owned by [a reputable breeder]$_{HUMAN}$ is being treated for [IBD]$_{DISEASE}$ with [pred]$_{DRUG}$.

When we began working with these texts, we were immediately confronted by a dizzying array of non-standard words and word uses. In addition to formal veterinary vocabulary (e.g., animal diseases), veterinarians often use informal, shorthand terms when posting on-line. For example, they frequently refer to breeds using "nicknames" or shortened terms (e.g., *gshep* for German shepherd, *doxy* for dachsund, *bxr* for boxer, *labx* for labrador cross). Often, they refer to animals based solely on their physical characteristics, for example *"a dlh"* (domestic long hair), *"a m/n"* (male, neutered), or *"a 2yo"* (2 year old). This phenomenon occurs with other semantic categories as well, such as drugs and medical tests (e.g., *pred* for prednisone, and *rads* for radiographs).

Nearly all semantic class taggers are trained using supervised learning with manually annotated data. However, annotated data is rarely available for specialized domains, and it is expensive to obtain because domain experts must do the annotation work. So we set out to create a bootstrapping model that can rapidly create domain-specific semantic taggers using just a few seed words and a domain-specific text collection.

Our bootstrapping model consists of two distinct phases. First, we train *strictly contextual classifiers* from the seed annotations. We then apply the classifiers to the unlabeled data to generate new annotated instances that are added to the training set. Second, we employ a *cross-category bootstrapping* process that simultaneously trains a suite of classifiers for multiple semantic categories, using the positive instances for one semantic class as negative instances for the others. This cross-category training process gives the learner sustained momentum over many bootstrapping iterations. Finally, we explore two additional enhancements: (1) a *one-semantic-class-per-discourse* heuristic to automatically generate new training instances, and (2) dynamically created semantic features produced by the classifiers themselves. In the following sections, we explain each of these steps in detail.

3.2 Phase 1: Inducing a Contextual Classifier

The main challenge that we faced was how to train an instance-based classifier using seed words as the only form of human supervision. First, the user must provide a small set of seed words that are relatively unambiguous (e.g., *"dog"* will nearly always refer to an animal in our domain). But even so, training a traditional classifier from seed-based instances would likely produce a classifier that learns to recognize the seeds but is unable to classify new examples. We needed to force the classifier to generalize beyond the seed words.

Our solution was to introduce an initial training step that induces a *strictly contextual classifier*. First, we generate training instances by automatically labeling each instance of a seed word with its designated semantic class. However, when we create feature vectors for the classifier, the seeds themselves are hidden and only contextual features are used to represent each training instance. By essentially "masking" the seed words so the classifier can only see the contexts around them, we force the classifier to generalize.

We create a suite of *strictly contextual classifiers*, one for each semantic category. Each classifier makes a binary decision as to whether a noun phrase belongs to its semantic category. We use the seed words for category C_k to generate positive training instances for the C_k classifier, and the seed words for all other categories to generate the negative training instances for C_k.

We use an in-house sentence segmenter and NP chunker to identify the base NPs in each sentence and create feature vectors that represent each constituent in the sentence as either an NP or an individual word. For each seed word, the feature vector captures a context window of 3 constituents (word or NP) to its left and 3 constituents (word or NP) to its right. Each constituent is represented with a lexical feature: for NPs, we use its head noun; for individual words, we use the word itself. The seed word, however, is discarded so that the classifier is essentially blind-folded and cannot see the seed that produced the training instance. We also create a feature for every modifier that precedes the head noun in the target NP, except for articles which are discarded. As an example, consider the following sentence:

> *Fluffy was diagnosed with FELV after a blood test showed that he tested positive.*

Suppose that *"FELV"* is a seed for the DISEASE category and *"test"* is a seed for the TEST category. Two training instances would be created, with feature vectors that look like this, where M represents a modifier inside the target NP:

was_{-3} $diagnosed_{-2}$ $with_{-1}$ $after_{1}$ $test_{2}$ $showed_{3}$ $\Rightarrow$ DISEASE

$with_{-3}$ $FELV_{-2}$ $after_{-1}$ $blood_{M}$ $showed_{1}$ $that_{2}$ he_{3} $\Rightarrow$ TEST

The contextual classifiers are then applied to the corpus to automatically label new instances. We use a confidence score to label only the instances that the classifiers are most certain about. We compute a confidence score for instance i with respect to semantic class C_k by considering both the score of the C_k classifier as well as the scores of the competing classifiers. Intuitively, we have confidence in labeling an instance as category C_k if the C_k classifier gave it a positive score, and its score is much higher than the score of any other classifier. We use the following scoring function:

$$\text{Confidence}(i,C_k) = \text{score}(i,C_k) - \max(\forall_{j \neq k}\ \text{score}(i,C_j))$$

We employ support vector machines (SVMs) (Joachims, 1999) with a linear kernel as our classifiers, using the SVMlin software (Keerthi and DeCoste, 2005). We use the value produced by the decision function (essentially the distance from the hyperplane) as the score for a classifier. We specify a threshold θ_{cf} and only assign a semantic tag C_k to an instance i if Confidence(i,C_k) $\geq \theta_{cf}$.

All instances that pass the confidence threshold are labeled and added to the training set. This process greatly enhances the diversity of the training data. In this initial learning step, the strictly contextual classifiers substantially increase the number of training instances for each semantic category, producing a more diverse mix of seed-generated instances and context-generated instances.

3.3 Phase 2: Cross-Category Bootstrapping

The next phase of the learning process is an iterative bootstrapping procedure. The key challenge was to design a bootstrapping model that would not succumb to semantic drift and would have sustained momentum to continue learning over many iterations.

Figure 1 shows the design of our *cross-category bootstrapping* model.[2] We simultaneously train a suite of binary classifiers, one for each semantic category, $C_1 \ldots C_n$. After each training cycle, all of the classifiers are applied to the remaining unlabeled instances and each classifier labels the (positive) instances that it is most confident about (i.e., the instances that it classifies with a confidence score $\geq \theta_{cf}$). The set of instances positively labeled by classifier C_k are shown as C_k^+ in Figure 1. All of the new instances produced by classifier C_k are then added to the set of positive training instances for C_k and to the set of negative training instances for all of the other classifiers.

One potential problem with this scheme is that some categories are more prolific than others, plus we are collecting negative instances from a set of competing classifiers. Consequently, this approach can produce highly imbalanced training sets. Therefore we enforced a 3:1 ratio of negatives to positives by randomly selecting a subset of the possible negative instances. We discuss this issue further in Section 4.4.

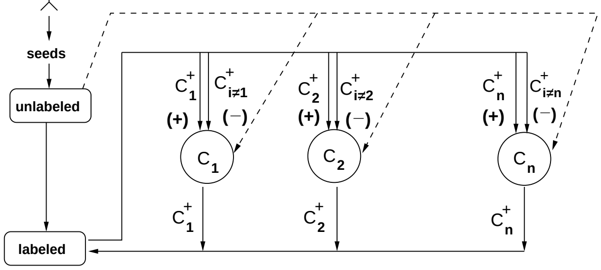

Figure 1: Cross-Category Bootstrapping

[2] For simplicity, this picture does not depict the initial contextual training step, but that can be viewed as the first iteration in this general framework.

Cross-category training has two advantages over independent self-training. First, as others have shown for pattern learning and lexicon induction (Thelen and Riloff, 2002; Yangarber, 2003; McIntosh and Curran, 2009), simultaneously training classifiers for multiple categories reduces semantic drift because each classifier is deterred from encroaching on another one's territory (i.e., claiming the instances from a competing class as its own). Second, similar in spirit to co-training[3], this approach allows each classifier to obtain new training instances from an outside source that has a slightly different perspective. While independent self-training can quickly run out of steam, cross-category training supplies each classifier with a constant stream of new (negative) instances produced by competing classifiers. In Section 4, we will show that cross-category bootstrapping performs substantially better than an independent self-training model, where each classifier is bootstrapped separately.

The feature set for these classifiers is exactly the same as described in Section 3.2, except that we add a new lexical feature that represents the head noun of the target NP (i.e., the NP that needs to be tagged). This allows the classifiers to consider the local context as well as the target word itself when making decisions.

3.4 One Semantic Class Per Discourse

We also explored the idea of using a *one semantic class per discourse (OSCPD)* heuristic to generate additional training instances during bootstrapping. Inspired by Yarowsky's *one sense per discourse* heuristic for word sense disambiguation (Yarowsky, 1995), we make the assumption that multiple instances of a word in the same discourse will nearly always correspond to the same semantic class. Since our data set consists of message board posts organized as threads, we consider all posts in the same thread to be a single discourse.

After each training step, we apply the classifiers to the unlabeled data to label some new instances. For each newly labeled instance, the OSCPD heuristic collects all instances with the same head noun in the same discourse (thread) and unilaterally labels them with the same semantic class. This heuristic serves as meta-knowledge to label instances that (potentially) occur in very different contexts, thereby infusing the bootstrapping process with "fresh" training examples.

In early experiments, we found that OSCPD can be aggressive, pulling in many new instances. If the classifier labels a word incorrectly, however, then the OSCPD heuristic will compound the error and mislabel even more instances incorrectly. Therefore we only apply this heuristic to instances that are labeled with extremely high confidence ($\theta_{cf} \geq 2.5$) and that pass a global sanity check, $gsc(w) \geq 0.2$, which ensures that a relatively high proportion of labeled instances with the same head noun have been assigned to the same semantic class. Specifically, $gsc(w) = 0.1 * \frac{w_{l/c}}{w_l} + 0.9 * \frac{w_{u/c}}{w_u}$ where w_l and w_u are the # of labeled and unlabeled instances, respectively, $w_{l/c}$ is the # of instances labeled as c, and $w_{u/c}$ is the # of unlabeled instances that receive a positive confidence score for c when given to the classifier. The intuition behind the second term is that most instances are initially unlabeled and we want to make sure that many of the unlabeled instances are likely to belong to the same semantic class (even though the classifier isn't ready to commit to them yet).

3.5 Dynamic Semantic Features

For many NLP tasks, classifiers use semantic features to represent the semantic class of words. These features are typically obtained from external resources such as Wordnet (Miller, 1990). Our bootstrapping model incrementally trains semantic class taggers, so we explored the idea of using the labels assigned by the classifiers to create enhanced feature vectors by dynamically adding semantic features. This process allows later stages of bootstrapping to directly benefit from earlier stages. For example, consider the sentence:

He started the doxy on Vetsulin today.

If *"Vetsulin"* was labeled as a DRUG in a previous bootstrapping iteration, then the feature vector representing the context around *"doxy"* can be enhanced to include an additional semantic feature identifying Vetsulin as a DRUG, which would look like this:

$$He_{-2}\ started_{-1}\ on_{1}\ Vetsulin_{2}\ DRUG_{2}\ today_{3}$$

Intuitively, the semantic features should help the classifier identify more general contextual patterns, such as *"started <X> on DRUG"*. To create semantic features, we use the semantic tags that

[3] But technically this is not co-training because our feature sets are all the same.

have been assigned to the current set of labeled instances. When a feature vector is created for a target NP, we check every noun instance in its context window to see if it has been assigned a semantic tag, and if so, then we add a semantic feature. In the early stages of bootstrapping, however, relatively few nouns will be assigned semantic tags, so these features are often missing.

3.6 Thresholds and Stopping Criterion

When new instances are automatically labeled during bootstrapping, it is critically important that most of the labels are correct or performance rapidly deteriorates. This suggests that we should only label instances in which the classifier has high confidence. On the other hand, a high threshold often yields few new instances, which can cause the bootstrapping process to sputter and halt. To balance these competing demands, we used a *sliding threshold* that begins conservatively but gradually loosens the reins as bootstrapping progresses. Initially, we set $\theta_{cf} = 2.0$, which only labels instances that the classifier is highly confident about. When fewer than min new instances can be labeled, we automatically decrease θ_{cf} by 0.2, allowing another batch of new instances to be labeled, albeit with slightly less confidence. We continue decreasing the threshold, as needed, until $\theta_{cf} < 1.0$, when we end the bootstrapping process. In Section 4, we show that this sliding threshold outperforms fixed threshold values.

4 Evaluation

4.1 Data

Our data set consists of message board posts from the Veterinary Information Network (VIN), which is a web site (www.vin.com) for professionals in veterinary medicine. Among other things, VIN hosts forums where veterinarians engage in discussions about medical issues, cases in their practices, etc. Over half of the small animal veterinarians in the U.S. and Canada use VIN. Analysis of veterinary data could not only improve pet health care, but also provide early warning signs of infectious disease outbreaks, emerging zoonotic diseases, exposures to environmental toxins, and contamination in the food chain.

We obtained over 15,000 VIN message board threads representing three topics: cardiology, endocrinology, and feline internal medicine. We did basic cleaning, removing html tags and tokenizing numbers. For training, we used 4,629 threads, consisting of 25,944 individual posts. We developed classifiers to identify six semantic categories: ANIMAL, DISEASE/SYMPTOM[4], DRUG, HUMAN, TEST, and OTHER.

The message board posts contain an abundance of veterinary terminology and jargon, so two domain experts[5] from VIN created a test set (answer key) for our evaluation. We defined annotation guidelines[6] for each semantic category and conducted an inter-annotator agreement study to measure the consistency of the two domain experts on 30 message board posts, which contained 1,473 noun phrases. The annotators achieved a relatively high κ score of .80. Each annotator then labeled an additional 35 documents, which gave us a *test set* containing 100 manually annotated message board posts. The table below shows the distribution of semantic classes in the test set.

Animal	Dis/Sym	Drug	Test	Human	Other
612	900	369	404	818	1723

To select seed words, we used the procedure proposed by Roark and Charniak (1998), ranking all of the head nouns in the training corpus by frequency and manually selecting the first 10 nouns that *unambiguously* belong to each category.[7] This process is fast, relatively objective, and guaranteed to yield high-frequency terms, which is important for bootstrapping. We used the Stanford part-of-speech tagger (Toutanova et al., 2003) to identify nouns, and our own simple rule-based NP chunker.

4.2 Baselines

To assess the difficulty of our data set and task, we evaluated several baselines. The first baseline searches for each head noun in WordNet and labels the noun as category C_k if it has a hypernym synset corresponding to that category. We manually identified the WordNet synsets that, to the best of our ability, seem to most closely correspond

[4] We used a single category for diseases and symptoms because our domain experts had difficulty distinguishing between them. A veterinary consultant explained that the same term (e.g., diabetes) may be considered a symptom in one context if it is secondary to another condition (e.g., pancreatitis) and a disease in a different context if it is the primary diagnosis.

[5] One annotator is a veterinarian and the other is a veterinary technician.

[6] The annotators were also allowed to label an NP as *POS_Error* if it was clearly misparsed. These cases were not used in the evaluation.

[7] We used 20 seeds for DIS/SYM because we merged two categories and for OTHER because it is a broad catch-all class.

Method	Animal	Dis/Sym	Drug	Test	Human	Other	Avg
BASELINES							
WordNet	32/80/46	21/81/34	25/35/29	NA	62/66/64	NA	35/66/45.8
Seeds	38/100/55	14/99/25	21/97/35	29/94/45	80/99/88	18/93/30	37/98/53.1
Supervised	67/94/78	20/88/33	24/96/39	34/90/49	79/99/88	31/91/46	45/94/60.7
Ind. Self-Train I.13	61/84/71	39/80/52	53/77/62	55/70/61	81/96/88	30/82/44	58/81/67.4
CROSS-CATEGORY BOOTSTRAPPED CLASSIFIERS							
Contextual I.1	59/77/67	33/84/47	42/80/55	49/77/59	82/93/87	33/80/47	53/82/64.3
XCategory I.45	86/71/78	57/82/67	70/78/74	73/65/69	85/92/89	46/82/59	75/78/76.1
XCat+OSCPD I.40	86/69/77	59/81/68	72/70/71	72/69/71	86/92/89	50/81/62	75/76/75.6
XCat+OSCPD+SF I.39	86/70/77	60/81/69	69/81/75	73/69/71	86/91/89	50/81/62	75/78/76.6

Table 1: Experimental results, reported as Recall/Precision/F score

to each semantic class. We do not report WordNet results for TEST because there did not seem be an appropriate synset, or for the OTHER category because that is a catch-all class. The first row of Table 1 shows the results, which are reported as Recall/Precision/F score[8]. The WordNet baseline yields low recall (21-32%) for every category except HUMAN, which confirms that many veterinary terms are not present in WordNet. The surprisingly low precision for some categories is due to atypical word uses (e.g., *patient*, *boy*, and *girl* are HUMAN in WordNet but nearly always ANIMALS in our domain), and overgeneralities (e.g., WordNet lists *calcium* as a DRUG).

The second baseline simply labels every instance of a seed with its designated semantic class. All non-seed instances remain unlabeled. As expected, the seeds produce high precision but low recall. The exception is HUMAN, where 80% of the instances match a seed word, undoubtedly because five of the ten HUMAN seeds are 1st and 2nd person pronouns, which are extremely common.

A third baseline trains semantic classifiers using supervised learning by performing 10-fold cross-validation on the test set. The feature set and classifier settings are exactly the same as with our bootstrapped classifiers.[9] Supervised learning achieves good precision but low recall for all categories except ANIMAL and HUMAN. In the next section, we present the experimental results for our bootstrapped classifiers.

4.3 Results for Bootstrapped Classifiers

The bottom section of Table 1 displays the results for our bootstrapped classifiers. The **Contextual I.1** row shows results after just the first iteration, which trains only the *strictly contextual classifiers*. The average F score improved from 53.1 for the seeds alone to 64.3 with the contextual classifiers. The next row, **XCategory I.45**, shows the results after cross-category bootstrapping, which ended after 45 iterations. (We indicate the number of iterations until bootstrapping ended using the notation **I.#**.) With cross-category bootstrapping, the average F score increased from 64.3 to 76.1. A closer inspection reveals that all of the semantic categories except HUMAN achieved large recall gains. And importantly, these recall gains were obtained with relatively little loss of precision, with the exception of TEST.

Next, we measured the impact of the *one-semantic-class-per-discourse* heuristic, shown as **XCat+OSCPD I.40**. From Table 1, it appears that OSCPD produced mixed results: recall increased by 1-4 points for DIS/SYM, DRUG, HUMAN, and OTHER, but precision was inconsistent, improving by +4 for TEST but dropping by -8 for DRUG. However, this single snapshot in time does not tell the full story. Figure 2 shows the performance of the classifiers during the course of bootstrapping. The OSCPD heuristic produced a steeper learning curve, and consistently improved performance until the last few iterations when its performance dipped. This is probably due to the fact that noise gradually increases during bootstrapping, so incorrect labels are more likely and OSCPD will compound any mistakes by the classifier. A good future strategy might be to use the OSCPD heuristic only during the early stages of bootstrapping when the classifier's decisions are most reliable.

We also evaluated the effect of dynamically created semantic features. When added to the basic XCategory system, they had almost no effect. We suspect this is because the semantic features are sparse during most of the bootstrapping process. However, the semantic features did im-

[8] We use an F(1) score, where recall and precision are equally weighted.

[9] For all of our classifiers, supervised and bootstrapped, we label all instances of the seed words first and then apply the classifiers to the unlabeled (non-seed) instances.

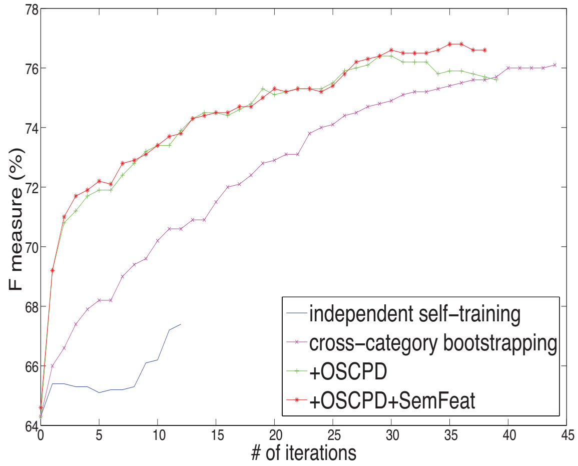

Figure 2: Average F scores after each iteration

prove performance when coupled with the OSCPD heuristic, presumably because the OSCPD heuristic aggressively labels more instances in the earlier stages of bootstrapping, increasing the prevalence of semantic class tags. The **XCat+OSCPD+SF I.39** row in Table 1 shows that the semantic features coupled with OSCPD dramatically increased the precision for DRUG, yielding the best overall F score of 76.6.

We conducted one additional experiment to assess the benefits of cross-category bootstrapping. We created an analogous suite of classifiers using self-training, where each classifier independently labels the instances that it is most confident about, adds them only to its own training set, and then retrains itself. The **Ind. Self-Train I.13** row in Table 1 shows that these classifiers achieved only 58% recall (compared to 75% for **XCategory**) and an average F score of 67.4 (compared to 76.1 for **XCategory**). One reason for the disparity is that the self-training model ended after just 13 bootstrapping cycles (**I.13**), given the same threshold values. To see if we could push it further, we lowered the confidence threshold to 0 and it continued learning through 35 iterations. Even so, its final score was 65% recall with 79% precision, which is still well below the 75% recall with 78% precision produced by the **XCategory** model. These results support our claim that cross-category bootstrapping is more effective than independently self-trained models.

Figure 3 tracks the recall and precision scores of the **XCat+OSCPD+SF** system as bootstrapping progresses. This graph shows the sustained momentum of cross-category bootstrapping: recall steadily improves while precision stays consistently high with only a slight dropoff at the end.

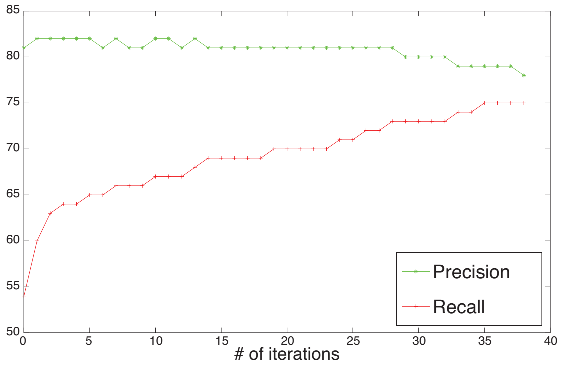

Figure 3: Recall and Precision scores during cross-category bootstrapping

4.4 Analysis

To assess the impact of corpus size, we generated a learning curve with randomly selected subsets of the training texts. Figure 4 shows the average F score of our best system using $\frac{1}{16}$, $\frac{1}{8}$, $\frac{1}{4}$, $\frac{1}{2}$, $\frac{3}{4}$, and all of the data. With just $\frac{1}{16}$th of the training set, the system has about 1,600 message board posts to use for training, which yields a similar F score (roughly 61%) as the supervised baseline that used 100 manually annotated posts via 10-fold cross-validation. So with 16 times more text, seed-based bootstrapping achieves roughly the same results as supervised learning. This result reflects the natural trade-off between supervised learning and seed-based bootstrapping. Supervised learning exploits manually annotated data, but must make do with a relatively small amount of training text because manual annotations are expensive. In contrast, seed-based bootstrapping exploits a small number of human-provided seeds, but needs a larger set of (unannotated) texts for training because the seeds produce relatively sparse annotations of the texts.

An additional advantage of seed-based bootstrapping methods is that they can easily exploit unlimited amounts of training text. For many domains, large text collections are readily available. Figure 4 shows a steady improvement in performance as the amount of training text grows. Overall, the F score improves from roughly 61% to nearly 77% simply by giving the system access to more unannotated text during bootstrapping.

We also evaluated the effectiveness of our sliding confidence threshold (Section 3.6). The table below shows the results using fixed thresholds

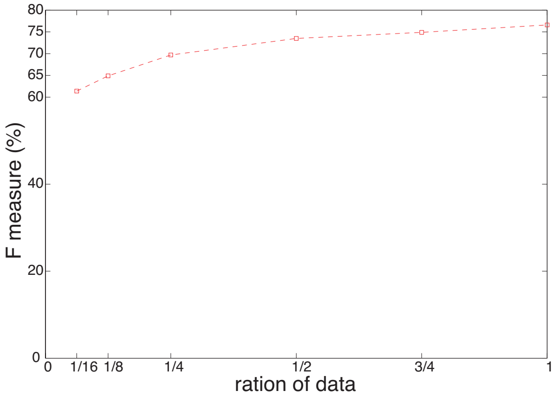

Figure 4: Learning Curve

of 1.0, 1.5, 2.0, as well as the sliding threshold (which begins at 2.0 and ends at 1.0 decreasing by 0.2 when the number of newly labeled instances falls below 3000 (i.e., $<$ 500 per category, on average). This table depicts the expected trade-off between recall and precision for the fixed thresholds, with higher thresholds producing higher precision but lower recall. The sliding threshold produces the best F score, achieving the best balance of high recall and precision.

θ_{cf}	**R/P/F**
1.0	71/77/74.1
1.5	69/81/74.7
2.0	65/82/72.4
Sliding	75/78/76.6

As mentioned in Section 3.3, we used 3 times as many negative instances as positive instances for every semantic category during bootstrapping. This ratio was based on early experiments where we needed to limit the number of negative instances per category because the cross-category framework naturally produces an extremely skewed negative/positive training set. We revisited this issue to empirically assess the impact of the negative/positive ratio on performance. The table below shows recall, precision, and F score results when we vary the ratio from 1:1 to 5:1. A 1:1 ratio seems to be too conservative, improving precision a bit but lowering recall. However the difference in performance between the other ratios is small. Our conclusion is that a 1:1 ratio is too restrictive but, in general, the cross-category bootstrapping process is relatively insensitive to the specific negative/positive ratio used. Our observation from preliminary experiments, however, is that the negative/positive ratio does need to be controlled, or else the dominant categories overwhelm the less frequent categories with negative instances.

Neg:Pos	**R/P/F**
1:1	72/79/75.2
2:1	74/78/76.1
3:1	75/78/76.6
4:1	75/77/76.0
5:1	76/77/76.4

Finally, we examined performance on gendered pronouns (he/she/him/her), which can refer to either animals or people in the veterinary domain. 84% (220/261) of the gendered pronouns were annotated as ANIMAL in the test set. Our classifier achieved 95% recall (209/220) and 90% precision (209/232) for ANIMAL and 15% recall (6/41) and 100% precision (6/6) for HUMAN. So while it failed to recognize most of the (relatively few) gendered pronouns that refer to a person, it was highly effective at identifying the ANIMAL references and it was always correct when it did assign a HUMAN tag to a pronoun.

5 Conclusions

We presented a novel technique for inducing domain-specific semantic class taggers from a handful of seed words and an unannotated text collection. Our results showed that the induced taggers achieve good performance on six semantic categories associated with the domain of veterinary medicine. Our technique allows semantic class taggers to be rapidly created for specialized domains with minimal human effort. In future work, we plan to investigate whether these semantic taggers can be used to improve other tasks.

Acknowledgments

We are very grateful to the people at the Veterinary Information Network for providing us access to their resources. Special thanks to Paul Pion, DVM and Nicky Mastin, DVM for making their data available to us, and to Sherri Lofing and Becky Lundgren, DVM for their time and expertise in creating the gold standard annotations. This research was supported in part by Department of Homeland Security Grant N0014-07-1-0152 and Air Force Contract FA8750-09-C-0172 under the DARPA Machine Reading Program.

References

ACE. 2005. NIST ACE evaluation website. In *http://www.nist.gov/speech/tests/ace/2005*.

ACE. 2007. NIST ACE evaluation website. In *http://www.nist.gov/speech/tests/ace/2007*.

ACE. 2008. NIST ACE evaluation website. In *http://www.nist.gov/speech/tests/ace/2008*.

Daniel M. Bikel, Scott Miller, Richard Schwartz, and Ralph Weischedel. 1997. Nymble: a high-performance learning name-finder. In *Proceedings of ANLP-97*, pages 194–201.

A. Blum and T. Mitchell. 1998. Combining Labeled and Unlabeled Data with Co-Training. In *Proceedings of the 11th Annual Conference on Computational Learning Theory (COLT-98)*.

Andrew Carlson, Justin Betteridge, Estevam R. Hruschka Jr., and Tom M. Mitchell. 2009. Coupling semi-supervised learning of categories and relations. In *HLT-NAACL 2009 Workshop on Semi-Supervised Learning for NLP*.

M. Collins and Y. Singer. 1999. Unsupervised Models for Named Entity Classification. In *Proceedings of the Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora (EMNLP/VLC-99)*.

S. Cucerzan and D. Yarowsky. 1999. Language Independent Named Entity Recognition Combining Morphologi cal and Contextual Evidence. In *Proceedings of the Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora (EMNLP/VLC-99)*.

O. Etzioni, M. Cafarella, D. Downey, A. Popescu, T. Shaked, S. Soderland, D. Weld, and A. Yates. 2005. Unsupervised named-entity extraction from the web: an experimental study. *Artificial Intelligence*, 165(1):91–134, June.

M.B. Fleischman and E.H. Hovy. 2002. Fine grained classification of named entities. In *Proceedings of the COLING conference*, August.

T. Joachims. 1999. Making Large-Scale Support Vector Machine Learning Practical. In A. Smola B. Schölkopf, C. Burges, editor, *Advances in Kernel Methods: Support Vector Machines*. MIT Press, Cambridge, MA.

S. Keerthi and D. DeCoste. 2005. A Modified Finite Newton Method for Fast Solution of Large Scale Linear SVMs. *Journal of Machine Learning Research*.

Mamoru Komachi, Taku Kudo, Masashi Shimbo, and Yuji Matsumoto. 2008. Graph-based analysis of semantic drift in espresso-like bootstrapping algorithms. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*.

Z. Kozareva, E. Riloff, and E. Hovy. 2008. Semantic Class Learning from the Web with Hyponym Pattern Linkage Graphs. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL-08)*.

D. McClosky, E. Charniak, and M Johnson. 2006. Effective self-training for parsing. In *HLT-NAACL-2006*.

T. McIntosh and J. Curran. 2009. Reducing Semantic Drift with Bagging and Distributional Similarity. In *Proceedings of the 47th Annual Meeting of the Association for Computational Linguistics*.

R. Mihalcea. 2004. Co-training and Self-training for Word Sense Disambiguation. In *CoNLL-2004*.

G. Miller. 1990. Wordnet: An On-line Lexical Database. *International Journal of Lexicography*, 3(4).

C. Mueller, S. Rapp, and M. Strube. 2002. Applying co-training to reference resolution. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*.

V. Ng and C. Cardie. 2003. Weakly supervised natural language learning without redundant views. In *HLT-NAACL-2003*.

V. Ng. 2007. Semantic Class Induction and Coreference Resolution. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics*.

Cheng Niu, Wei Li, Jihong Ding, and Rohini K. Srihari. 2003. A bootstrapping approach to named entity classification using successive learners. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics (ACL-03)*, pages 335–342.

M. Paşca. 2004. Acquisition of categorized named entities for web search. In *Proc. of the Thirteenth ACM International Conference on Information and Knowledge Management*, pages 137–145.

W. Phillips and E. Riloff. 2002. Exploiting Strong Syntactic Heuristics and Co-Training to Learn Semantic Lexicons. In *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing*, pages 125–132.

E. Riloff and R. Jones. 1999. Learning Dictionaries for Information Extraction by Multi-Level Bootstrapping. In *Proceedings of the Sixteenth National Conference on Artificial Intelligence*.

E. Riloff and J. Shepherd. 1997. A Corpus-Based Approach for Building Semantic Lexicons. In *Proceedings of the Second Conference on Empirical Methods in Natural Language Processing*, pages 117–124.

B. Roark and E. Charniak. 1998. Noun-phrase Co-occurrence Statistics for Semi-automatic Semantic Lexicon Construction. In *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics*, pages 1110–1116.

M. Thelen and E. Riloff. 2002. A Bootstrapping Method for Learning Semantic Lexicons Using Extraction Pa ttern Contexts. In *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing*, pages 214–221.

K. Toutanova, D. Klein, C. Manning, and Y. Singer. 2003. Feature-Rich Part-of-Speech Tagging with a Cyclic Dependency Network. In *Proceedings of HLT-NAACL 2003*.

R. Yangarber. 2003. Counter-training in the discovery of semantic patterns. In *Proceedings of the 41th Annual Meeting of the Association for Computational Linguistics*.

D. Yarowsky. 1995. Unsupervised Word Sense Disambiguation Rivaling Supervised Methods. In *Proceedings of the 33rd Annual Meeting of the Association for Computational Linguistics*.

Imed Zitouni and Radu Florian. 2009. Cross-language information propagation for arabic mention detection. *ACM Transactions on Asian Language Information Processing (TALIP)*, 8(4):1–21.