Semi-Supervised Learning of Attribute-Value Pairs from Product Descriptions

Katharina Probst, Rayid Ghani, Marko Krema, Andrew Fano
Accenture Technology Labs, Chicago, IL, USA
Yan Liu
Carnegie Mellon University, Pittsburgh, PA, USA

Abstract

We describe an approach to extract attribute-value pairs from product descriptions. This allows us to represent products as sets of such attribute-value pairs to augment product databases. Such a representation is useful for a variety of tasks where treating a product as a set of attribute-value pairs is more useful than as an atomic entity. Examples of such applications include product recommendations, product comparison, and demand forecasting. We formulate the extraction as a classification problem and use a semi-supervised algorithm (co-EM) along with (Naïve Bayes). The extraction system requires very little initial user supervision: using unlabeled data, we automatically extract an initial seed list that serves as training data for the supervised and semi-supervised classification algorithms. Finally, the extracted attributes and values are linked to form pairs using dependency information and co-location scores. We present promising results on product descriptions in two categories of sporting goods.

1 Introduction

Retailers have been collecting a large amount of sales data containing customer information and related transactions. These data warehouses also contain product information which is often very sparse and limited. Most retailers treat their products as atomic entities with very few related attributes (typically brand, size, or color). This hinders the effectiveness of many applications that businesses currently use transactional data for such as product recommendation, demand forecasting, assortment optimization, and assortment comparison. If a business could represent their products in terms of attributes and attribute values, all of the above applications could be improved significantly.

Suppose, for example, that a sporting retailer wanted to forecast sales of a specific running shoe. Typically, they would look at the sales of the same product from the same time last year and adjust that based on new information. Now suppose that the shoe is described with the following attributes: *Lightweight mesh nylon material*, *Low Profile Sole*, *Standard lacing system*. Improved forecasting is possible if the retailer is able to describe the shoe not only with a product number, but with a set of attribute-value pairs, such as *material: lightweight mesh nylon*, *sole: low profile*, *lacing system: standard*. This would enable the retailer to use data from other products having similar attributes. While many retailers have recently realized this and are working towards enriching product databases with attribute-value pairs, the work is currently being done completely manually: by looking at product descriptions that are available in an internal database or on the web or by looking at the actual physical product packaging in the store. The work presented in this paper is motivated by the need to make the process of extracting attribute-value pairs from product descriptions more efficient and cheaper by developing an interactive tool that can help human experts with this task.

The task we tackle in this paper requires a system that can process textual product descriptions and extract relevant attributes and values, and then form pairs by associating values with the attributes they describe. This is accomplished with minimal human supervision. We describe the components of our system and show experimental results on a web catalog of sporting goods products.

2 Related Work

While we are not aware of any system that addresses the task described in this paper, much research has been done on extracting information from text documents. One task that has received attention recently is that of extracting product features and their polarity from online user reviews. Liu et al. [Liu and Cheng, 2005] describe a system that focuses on extracting relevant product attributes, such as *focus* for digital cameras. These attributes are extracted by use of a rule miner, and are restricted to noun phrases. In a second phase, the system extracts polarized descriptors, e.g., *good*, *too small*, etc. [Popescu and Etzioni, 2005] describe a similar approach: they approach the task by first extracting noun phrases as candidate attributes, and then computing the mutual information between the noun phrases and salient context patterns. Our work is related in that in both cases, a product is expressed as a vector of attributes. However, our work focuses not only on attributes, but also on values, and on pairing attributes with values. Furthermore, the attributes that are extracted from user reviews are often different than the attributes of the products that retailers would mention. For example, a review might mention *photo quality* as an attribute but specifications of cameras would tend to use *megapixels* or *lens manufacturer* in the specifications.

Information extraction for filling templates, e.g., [Seymore *et al.*, 1999; Peng and McCallum, 2004], is related to the approach in this paper in that we extract certain parts of the text as relevant facts. It however differs from such tasks in several ways, notably because we do not have a definitive list of 'template slots' available. Recent work in bootstrapping for information extraction using semi-supervised learning has focused on named entity extraction [Jones, 2005; Collins and Singer, 1999; Ghani and Jones, 2002], which is related to part of the work presented here (classifying the words/phrase as attributes or values or as neither).

3 Overview of the Attribute Extraction System

Our system consists of five modules:

1. **Data Collection** from an internal database containing product information/descriptions or from the web using web crawlers and wrappers.
2. **Seed Generation**, i.e., automatically creating seed attribute-value pairs.
3. **Attribute-Value Entity Extraction** from unlabeled product descriptions. This tasks lends itself naturally to classification. We chose a supervised algorithm (Naïve Bayes) as well as a semi-supervised algorithm (co-EM).
4. **Attribute-Value Pair Relationship Extraction**, i.e., forming pairs from extracted attributes and values. We use a dependency parser (Minipar, [Lin, 1998]) as well as co-location scores.
5. **User Interaction** via active learning to allow users to correct the results as well as provide additional training data for the system to learn from.

The modular design allows us to break the problem into smaller steps, each of which can be addressed by various approaches. We only focus on tasks 1-4 in this paper and describe our approach to each of the four tasks in greater detail. We are currently investigating the user interaction phase.

4 Data and Preprocessing

For the experiments reported in this paper, we developed a web crawler that traverses retailer web sites and extracts product descriptions. We crawled the web site of a sporting goods retailer[1], concentrating on the domains of tennis and football. We believe that sporting goods is an interesting and relatively challenging domain because the attributes are not easy and straightforward to detect. For example, a baseball bat would have non-obvious attributes and values such as *aerodynamic construction, curved hitting surface*, etc. The web crawler gives us a set of product descriptions, which we use as (unlabeled) training data. Some examples are:

4 rolls white athletic tape
Extended Torsion bar
Audio/Video Input Jack
Vulcanized latex outsole construction is lightweight and flexible

It can be seen from these examples that the entries are not often full sentences. This makes the extraction task more difficult, because most of the phrases contain a number of modifiers, e.g., *cutter* being modified both by *1* and by *tape*. For this reason, there is often no definitive answer as to what the extracted attribute-value pair should be, even for humans inspecting the data. For instance, should the system extract *cutter* as an attribute with two separate values, *1* and *tape*, or should it rather extract *tape cutter* as an attribute and *1* as a value? To answer this question, it is important to keep in mind the goal of the system to express each product as a vector of attribute-value pairs, and to compare products. For this reason, it is more important that the system is consistent than which of the valid answers it gives.

The data is first tagged with parts of speech (POS) using the Brill tagger [Brill, 1995] and stemmed with the Porter stemmer [Porter, 1980]. We also replace all numbers with the unique token *#number#* and all measures (e.g., *liter, kg*) by the unique token *#measure#*.

5 Seed Generation

Once the data is collected and processed, the next step is to provide labeled seeds for the learning algorithms to learn from. The extraction algorithm is seeded in two ways: with a list of known attributes and values, as well as with an unsupervised, automated algorithm that extracts a set of seed pairs from the unlabeled data. Both of these seeding mechanisms are designed to facilitate scaling to other domains.

We also experimented with labeled training data: some of the data that we used in our experiments exhibits structural patterns that clearly indicate attribute-value pairs separated by colons, e.g., *length: 5 inches*. Such structural cues can be used to obtain labeled training data. In our experiments, however, the labeled pairs were not very useful. The results obtained with the automatically extracted pairs are comparable to the ones obtained when the given attribute-value pairs were used. In the case of comparable results, we prefer the unsupervised approach because it does not rely on the structure of the data and is domain-independent.

We use a very small amount of labeled training data in the form of generic and domain-specific value lists that are readily available. We use four lists: one each for colors, materials, countries, and units of measure. We also use a list of domain-specific (in our case, sports) values: this list consists of sports teams (such as *Pittsburgh Steelers*), and contains 82 entries.

In addition to the above lists, we automatically extract a small number of attribute-value seed pairs in an unsupervised way as described in the following. Extracting attribute-value pairs is related to the problem of phrase recognition in that both methods aim at extracting pairs of highly correlated words. There are however differences between the two problems. Consider the following two sets of phrases:

back pockets, *front pockets*, and *zip pockets*
vs.

[1]www.dickssportinggoods.com

Pittsburgh Steelers, *Chicago Bears*

The first set contains an example of an attribute with several possible values. The second set contains phrases that are not attribute-value pairs. The biggest difference between the two lists is that attributes generally have more than one possible value but do not occur with a very large number of words. For this reason, two words that always occur together and common words such as *the* are not good attribute candidates.

In order to exploit these observations, we consider all bigrams $w_i w_{i+1}$ as candidates for pairs, where w_i is a candidate value, and w_{i+1} is a candidate attribute. Although it is not always the case that the modifying value occurs (directly) before its attribute, this heuristic allows us to extract seeds with high precision. Suppose word w (in position $i+1$) occurs with n unique words $w_{1...n}$ in position i. We rank the words $w_{1...n}$ by their conditional probability $p(w_j|w), w_j \in w_{1...n}$, where the word w_j with the highest conditional probability is ranked highest.

The words w_j that have the highest conditional probability are candidates for values for the candidate attribute w. Clearly, however, not all words are good candidate attributes. We observed that attributes generally have more than one value and typically do not occur with a wide range of words. For example, frequent words such as *the* occur with many different words. This is indicated by their conditional probability mass being distributed over a large number of words. We are interested in cases where few words account for a high proportion of the probability mass. For example, both *Steelers* and *on* will not be good candidates for being attributes. *Steelers* only occurs after *Pittsburgh* so all of the conditional probability mass will be distributed on one value whereas *on* occurs with many words with the mass distributed over too many values. This intuition can be captured in two phases: in the first phase, we retain enough words w_j to account for a part $z, 0 < z < 1$ of the conditional probability mass $\sum_{j=1}^{k} p(w_j|w)$. In the experiments reported here, z was set to 0.5.

In the second phase, we compute the *cumulative* modified mutual information for all candidate attribute-value pairs. We again consider the perspective of the candidate attribute. If there are a few words that together have a high mutual information with the candidate attribute, then we are likely to have found an attribute and (some of) its values. We define the cumulative modified mutual information as follows:

Let $p(w, w_{1...k}) = \sum_{j=1}^{k} p(w, w_j)$. Then

$$cmi(w_{1...k}; w) = \log \frac{p(w, w_{1...k})}{(\lambda * \sum_{j=1}^{k} p(w_j)) * ((\lambda - 1) * p(w))}$$

λ is a user-specified parameter, where $0 < \lambda < 1$. We have experimented with several values, and have found that setting λ close to 1 yields robust results. Table 1 lists several examples of extracted attribute-value pairs.

As we can observe from the table, our unsupervised seed generation algorithm captures the intuition we described earlier and extracts high-quality seeds for training the system. We expect to refine this method in the future. Currently, not all extracted pairs are actual attribute-value pairs. One typical example of an extracted incorrect pair are first name - last name pairs, e.g., *Smith* is extracted as an attribute as it occurs as part of many phrases and fulfills our criteria (Joe Smith, Mike Smith, etc.) after many first names. Our unsupervised seed generation algorithm gets about 65% accuracy in the tennis category and about 68% accuracy in the football category. We have experimented with manually correcting the seeds by eliminating all those that were incorrect. This did not result in any improvement of the final extraction performance, leading us to conclude that our algorithm is robust to noise and is able to deal with noisy seeds.

value	**attribute**
carrying, storage	case
main, racquet	compartment
welt, side-seam, key	pocket

Table 1: Automatically extracted seed attribute-value pairs

6 Attribute and Value Extraction

After generating initial seeds, the next step is to use the seeds as labeled training data to train the learning algorithms and extract attributes and values from the unlabeled data. We formulate the extraction as a classification problem where each word or phrase can be classified as an attribute or a value (or as neither). We treat this as a supervised learning problem and use Naïve Bayes as our first approach. The initial seed training data is generated as described in the previous section and serves as labeled data which Naïve Bayes uses to train a classifier. Since our goal is to create a system that minimizes human effort required to train the system, we use semi-supervised learning to improve the performance of Naïve Bayes by exploiting large amounts of unlabeled data available for free on the Web. Gathering product descriptions (from retail websites) is a relatively cheap process using simple web crawlers. The expensive part is labeling the words in the descriptions as attributes or values. We augment the initial seeds (labeled data) with the all the unlabeled product descriptions collected in the data collection phase and use semi-supervised learning (co-EM [Nigam and Ghani, 2000] with Naïve Bayes) to improve attribute-value extraction performance. The classification algorithm is described in the sections below.

6.1 Initial labeling

The initial labeling of data items (words or phrases) is based on whether they match the labeled data. We define four classes to classify words into: *unassigned, attribute, value, or neither*. The initial label for each word defaults to *unassigned*. If the unlabeled example does not match the labeled data, this default remains as input for the classification algorithm. If the unlabeled example does match the labeled data, then we simply assign it that label. By contrast, words that appear on a stoplist are tagged as *neither*.

6.2 Naïve Bayes Classification

We apply the training seeds to the unlabeled data in order to assign labels to as many words as possible, as described in the previous section. These labeled words are then used as training data for Naïve Bayes that classifies each word or phrase in the unlabeled data as an attribute, a value, or neither.

The features used for classification are the words of each unlabeled data item, the surrounding 8 words, and their corresponding parts of speech. With this feature set, we capture not only each word, but also its context as well as the parts of speech in its context. This is similar to earlier work in extracting named entities using labeled and unlabeled data [Ghani and Jones, 2002].

6.3 co-EM for Attribute Extraction

Since labeling attributes and values is an expensive process, we would like to reduce the amount of labeled data required to train accurate classifiers. Gathering unlabeled product descriptions is cheap. This allows us to use the semi-supervised learning setting by combining small amounts of labeled data with large amounts of unlabeled data. We use the multi-view or co-training [Blum and Mitchell, 1998] setting, where each example can be described by multiple views (e.g., the word itself and the context in which it occurs). It has been shown that such multi-view algorithms often outperform a single-view EM algorithm for information extraction tasks [Jones, 2005].

The specific algorithm we use is co-EM: a multi-view semi-supervised learning algorithm, proposed by [Nigam and Ghani, 2000], that combines features from both co-training [Blum and Mitchell, 1998] and Expectation-Maximization (EM). co-EM is iterative, like EM, but uses the feature split present in the data, like co-training. The separation into feature sets we use is that of the word to be classified and the context in which it occurs. Co-EM with Naïve Bayes has been applied to classification, e.g., by [Nigam and Ghani, 2000], but so far as we are aware, not in the context of information extraction. To express the data in two views, each word is expressed in *view1* by the stemmed word itself, plus the part of speech as assigned by the Brill tagger. The *view2* for this data item is a context of window size 8, i.e. up to 4 words (plus parts of speech) before and up to 4 words (plus parts of speech) after the word or phrase in *view1*. By default, all words are processed into *view1* as single words. Phrases that are recognized with correlation scores (Yule's Q, χ^2, and pointwise mutual information) are treated as an entity and thus as a single $view1$ data item.

co-EM proceeds by initializing the *view1* classifier using the labeled data only. Then this classifier is used to probabilistically label all the unlabeled data. The context (*view2*) classifier is then trained using the original labeled data plus the unlabeled data with the labels provided by the *view1* classifier. Similarly, the *view2* classifier then relabels the data for use by the *view1* classifier, and this process iterates for a number of iterations or until the classifiers converge.

More specifically, labeling a $view2$ training example using the current $view1$ classifier is done as follows:

$$P(c_k|view2_i) \propto P(c_k) * P(view2_i|c_k)$$

if $view2_i$ does *not* match the labeled training data. Otherwise, the initial labeling is used.

$P(c_k)$ is the class probability for class c_k, which is estimated using the current labels in the other view. The word probabilities $P(view2_i|c_k)$ are estimated using the current labels in the other view as well as the number of times the $view2$ data element $view2_i$ occurs with each $view1$ data element. The opposite direction is done analogously. After co-EM is run for a pre-specified number of iterations, we assign final co-EM probability distributions to all $\langle view1_i, view2_j\rangle$ pairs as follows:

$$P(c_k|\langle view1_i, view2_j\rangle) = \frac{P(c_k|view1_i) + P(c_k|view2_j)}{2}$$

It should be noted that even words that are tagged with high probability as attributes or values are not necessarily extracted as part of an attribute-value pair in the next phase. They will generally only be extracted if they form part of a pair, as described below.

7 Finding Attribute-Value Pairs

The classification phase assigns a probability distribution over all the labels to each word (or phrase). This is not enough: often, subsequent words that are tagged with the same label should be *merged* to form an attribute or value phrase. Additionally, the system must establish *links* between attributes and their corresponding values, so as to form attribute-value pairs, especially for product descriptions that contain more than one attribute and/or value. We accomplish merging and linking using the following steps:

- **Step 1:** Link if attributes and values match a seed pair.
- **Step 2:** Merge words of the same label into phrases if their correlation scores exceed a threshold.
- **Step 3:** Link if there is a syntactic dependency from the value to the attribute (as given by Minipar [Lin, 1998]).
- **Step 4:** Link if attribute and value phrases exceed a co-location threshold.
- **Step 5:** Link if attribute and value phrases are adjacent.
- **Step 6:** Add implicit attributes material, country, color.
- **Step 7:** Extract binary attributes, i.e., attributes without values, if the attribute phrase appears frequently or if the unlabeled data item consists of only one word.

The steps described here are heuristics and by no means the only approach that can be taken to linking. We chose this order to follow the intuition that pairs should be linked first by known pairs, then by syntactic dependencies which are often indicative, and if those two fail, then they should finally be linked by co-location information as a fallback. Co-location information is clearly less indicative than syntactic dependencies, but can cover cases that dependencies do not capture.

In the process of establishing attribute-value pairs, we exclude words of certain parts of speech, namely most closed-class items such as prepositions and conjunctions.

In step 6, the system 'extracts' information that is not explicit in the data. The attributes (*country*, *color*, and/or *material*) are added to any existing attribute words for this value if the value is on the list of known countries, colors, and/or materials. Assigning attributes from known lists is an initial approach to extracting non-explicit attributes. In the future, we will explore this issue in greater details.

After all seven pair identification steps, some attribute or value phrases can remain unaffiliated. Some of them are valid attributes with binary values. For instance, the data item *Imported* is a valid attribute with the possible values *true* or *false*. We extract only those attributes that are single word data items as well as those attributes that occur frequently in the data as a phrase.

8 Evaluation

In this section, we present evaluation results for experiments performed on tennis and football categories. The tennis category contains 3194 unlabeled data items (i.e., individual phrases from the lists in the product descriptions), the football category 72825 items. Unsupervised seed extraction resulted in 169 attribute-value pairs for the tennis category and 180 pairs for football.

Table 2 shows a sample list of extracted attribute-value pairs (i.e., the output of the full system), together with the phrases that they were extracted from.

Full Example	**Attribute**	**Value**
1 1/2-inch polycotton blend tape	polycotton blend tape	1 1/2-inch
1 roll underwrap	underwrap	1 roll
1 tape cutter	tape cutter	1
Extended Torsion bar	bar	Torsion
Synthetic leather upper	#material# upper	leather
Metal ghillies	#material# ghillies	Metal
adiWear tough rubber outsole	rubber outsole	adiWear tough
Imported	Imported	#true#
Dual-density padding with Kinetofoam	padding	Dual-density
Contains 2 BIOflex concentric circle magnet	BIOflex concentric circle magnet	2
93% nylon, 7% spandex	#material#	93% nylon 7% spandex
10-second start-up time delay	start-up time delay	10-second

Table 2: Examples of extracted pairs for system run with co-EM

We ran our system in the following three settings to gauge the effectiveness of each component: 1) only using the automatically generated seeds and the generic lists ('Seeds' in the tables), 2) with the baseline Naïve Bayes classifier ('NB'), and 3) co-EM with Naïve Bayes ('coEM'). In order to make the experiments comparable, we do not vary pre-processing or seed generation, and keep the pair identification (linking) steps constant as well.

The evaluation of this task is not straightforward since people often do not agree on what the 'correct' attribute-value pair should be. Consider the example *Audio/JPEG navigation menu*. This phrase can be expressed as an attribute-value pair in multiple ways, e.g., with *navigation menu* as the attribute and *Audio/JPEG* as the value, or with *menu* as the attribute and *Audio/JPEG navigation* as the value. For this reason, we give partial credit to an automatically extracted attribute-value pair, even if it does not completely match the human annotation.

8.1 Precision and Recall

Whenever the system extracts a partially correct pair for an example that is also given by the human annotator, the pair is considered recalled. The results for this metric can be found in table 3.

	Seeds	NB	coEM
Recall Tennis	27.62	36.40	**69.87**
Recall Football	32.69	47.12	**78.85**

Table 3: Recall for *Tennis* and *Football* Categories

Recall differs greatly between system settings. More specifically, co-EM results in recalling a much larger number of pairs, whereas seed generation and Naïve Bayes yield relatively poor recall performance. Seed generation was not expected to yield high recall, as only few pairs were extracted. In addition, the results show that co-EM is more effective at extrapolating from few seed examples to achieve higher recall than Naïve Bayes.

In order to measure precision, we evaluate how many automatically extracted pairs match manual pairs completely, partially, or not at all. The percentage of pairs that are fully *or* partially correct is useful as a metric especially in the context of human post-processing: partially correct pairs are corrected faster than completely incorrect pairs. Table 4 lists results for this metric for both categories. The results show no conclusive difference between the three systems: co-EM achieves a higher percentage of fully correct pairs, while the other two result in a total higher percentage of fully or partially correct pairs.

	Seeds	NB	coEM
% fully correct Tennis	20.29	21.28	**26.60**
% full or part correct Tennis	98.56	**98.94**	96.81
% incorrect Tennis	1.44	**1.06**	3.19
% fully correct Football	15.38	14.44	**17.65**
% full or part correct Football	**96.15**	90.40	89.59
% incorrect Football	**3.85**	9.60	10.41

Table 4: Precision for *Tennis* and *Football* Categories

Currently, however, our system places higher importance on recall than on precision. Furthermore, co-EM results in reasonable levels of precision, and therefore offers a the best balance of the proposed algorithms. The F-1 measure, as shown in table 5, supports this intuition.

	Seeds	NB	coEM
F-1 Tennis	0.43	0.53	**0.81**
F-1 Football	0.49	0.62	**0.84**

Table 5: F-1 measure for *Tennis* and *Football* categories

	T nW	T W	F nW	F W
% fully correct	51.25	55.89	51.90	60.01
% flip to correct	12.08	20.14	9.62	10.25
% flip to partially correct	2.92	1.75	0.87	2.14
% partially correct	32.92	21.74	35.27	25.98
% incorrect	0.83	0.48	2.33	1.63

Table 6: *Non-weighted* and *Weighted* Precision Results for *Tennis* and *Football* Categories. 'T' stands for *tennis*, 'F' is *football*, 'nW' *non-weighted*, and 'W' is *weighted*

8.2 Precision Results for Most Frequent Data Items

As the training data contains many duplicates, it is more important to extract correct pairs for the most frequent pairs than for the less frequent ones. In this section, we report precision results for the most frequently data items. This is done by sorting the training data by frequency, and then manually inspecting the pairs that the system extracted for the most frequent 300 data items. This was done only for the system run that includes co-EM classification. We report precision results for the two categories (*tennis* and *football*) in two ways: first, we do a simple evaluation of each unique data item. Then we weight the precision results by the frequency of each sentence. In order to be consistent with the results from the previous section, we define five categories that capture very similar information to the information provided above. The five categories contain *fully correct* and *incorrect*. Another category is *Flip to correct*, meaning that the extracted pair would be *fully correct* if attribute and value labels were flipped. *Flip to partially correct* refers to pairs that would be *partially correct* if attribute and value were flipped. Finally, we define *partially correct* as before. Table 6 shows the results.

A final comment on the current performance of the system: while there is room for improvement in the results, in reality the automated system provides a vast improvement over the current 'state of the art': as we mentioned above, the real baseline, as it is being performed by businesses today, is the *manual* extraction of attribute-value pairs.

9 Conclusions and Future Work

We describe an approach to extract attribute-value pairs from product descriptions that requires very little user supervision. We start with a novel unsupervised seed generation algorithm that has high precision but limited recall. The supervised and especially the semi-supervised algorithm yield significantly increased recall with little or no decrease in precision using the automatically generated seeds as labeled data.

Future work will focus on adding an interactive step to the extraction algorithm that will allow users to correct extracted pairs as quickly and efficiently as possible. We are experimenting with different active learning algorithms to use in the interactive step to minimize the number of corrections required to improve the system. One of the main challenges with an interactive approach is to make co-EM efficient enough that the time of the user is optimized.

We believe that a powerful attribute extraction system can be useful in a wide variety of tasks, as it allows for the normalization of products as attribute-value vectors. This enables fine-grained comparison between products and can improve a variety of applications such as product recommender systems, price comparison engines, demand forecasting, assortment optimization and comparison systems.

References

[Blum and Mitchell, 1998] Avrim Blum and Tom Mitchell. Combining labeled and unlabeled data with co-training. In *COLT-98*, 1998.

[Brill, 1995] Eric Brill. Transformation-based error-driven learning and natural language processing: A case study in part of speech tagging. *Computational Linguistics*, 1995.

[Collins and Singer, 1999] M. Collins and Y. Singer. Unsupervised Models for Named Entity Classification. In *EMNLP/VLC*, 1999.

[Ghani and Jones, 2002] Rayid Ghani and Rosie Jones. A comparison of efficacy of bootstrapping algorithms for information extraction. In *LREC 2002 Workshop on Linguistic Knowledge Acquisition*, 2002.

[Jones, 2005] Rosie Jones. *Learning to Extract Entities from Labeled and Unlabeled Text.* Ph.D. Dissertation, 2005.

[Lin, 1998] Dekan Lin. Dependency-based evaluation of MINIPAR. In *Workshop on the Evaluation of Parsing Systems*, 1998.

[Liu and Cheng, 2005] Bing Liu and Minqing Hu and Junsheng Cheng. Opinion observer: Analyzing and comparing opinions on the web. In *Proceedings of WWW 2005*, 2005.

[Nigam and Ghani, 2000] Kamal Nigam and Rayid Ghani. Analyzing the effectiveness and applicability of co-training. In *Proceedings of the Ninth International Conference on Information and Knowledge Management (CIKM-2000)*, 2000.

[Peng and McCallum, 2004] Fuchun Peng and Andrew McCallum. Accurate information extraction from research papers using conditional random fields. In *HLT 2004*, 2004.

[Popescu and Etzioni, 2005] Ana-Maria Popescu and Oren Etzioni. Extracting product features and opinions from reviews. In *Proceedings of EMNLP 2005*, 2005.

[Porter, 1980] M. F. Porter. An algorithm for suffix stripping. *Program*, 14(3):130–137, 1980.

[Seymore *et al.*, 1999] Kristie Seymore, Andrew McCallum, and Roni Rosenfeld. Learning hidden markov model structure for information extraction. In *AAAI 99 Workshop on Machine Learning for Information Extraction*, 1999.