Shallow Semantic Parsing of Product Offering Titles (*for better automatic hyperlink insertion*)

Gabor Melli
VigLink Inc.
539 Bryant St.
San Francisco, CA USA
gmelli@viglink.com

ABSTRACT

With billions of database-generated pages on the Web where consumers can readily add priced product offerings to their virtual shopping cart, several opportunities will become possible once we can automatically recognize what exactly is being offered for sale on each page. We present a case study of a deployed data-driven system that first chunks individual titles into semantically classified sub-segments, and then uses this information to improve a hyperlink insertion service.

To accomplish this process, we propose an annotation structure that is general enough to apply to offering titles from most e-commerce industries while also being specific enough to identify useful semantics about each offer. To automate the parsing task we apply the best-practices approach of training a supervised conditional random fields model and discover that creating separate prediction models for some of the industries along with the use of model-ensembles achieves the best performance to date.

We further report on a real-world application of the trained parser to the task of growing a lexical dictionary of product-related terms which critically provides background knowledge to an affiliate-marketing hyperlink insertion service. On a regular basis we apply the parser to offering titles to produce a large set of labeled terms. From these candidates we select the most confidently predicted novel terms for review by crowd-sourced annotators. The agreed on terms are then added into a dictionary which significantly improves the performance of the link-insertion service. Finally, to continually improve system performance, we retrain the model in an online fashion by performing additional annotations on titles with incorrect predictions on each batch.

Categories and Subject Descriptors

H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval—*Information filtering*; I.2.7 [**Artificial Intelligence**]: Natural Language Processing—*text analysis*

KDD-2014 '14 New York, NY USA
ACM 978-1-4503-2956-9/14/08.
http://dx.doi.org/10.1145/2623330.2623343.

General Terms

Algorithms, Experimentation, Case-study

Keywords

shallow semantic parsing; automated terminology extraction; composite CRF ensembles; product offer titles; hyperlink insertion

1. INTRODUCTION

With billions of database-generated pages on the Web where consumers can readily add priced product offerings to their virtual shopping cart, several opportunities will become possible once we can automatically recognize what exactly is being offered for sale on each page. While there has been a significant amount of work reported on how to extract aggregate-level information about products (and the sentiments expressed about them) from webpage content, especially for electronic products, far less work has been reported on the task of automatically reading and recognizing all of the characteristics of any individual offering. When feasible, this capability will significantly aid in such tasks as terminology extraction and cross-seller product offering searches.

A natural place to begin is to characterize the information available within product offering *titles* such as: "*The Show (Album Version)*" and "*Pro Digital Lens Hood for VIXIA HF S10, S100 Flash Camcorders (1-year wrty from e-sekuro)*." These titles, along with price, product category, and seller name, are prevalent pieces of information available on the Web [1]. While titles are a form of unstructured text, their central role in consumer decision making and search engine rank performance, sellers are motivated to ensure that its text is rich in relevant information while also being easy to read/parse. Further, we will show, these titles can be interpreted to have structure in the form of semantic terms that refer to brands, features and a few other semantic classes, along with some syntactic terms (such as "*for*"). Titles, as with full sentences, can also be understood at a higher semantic parsing structure, but simply being able to automatically identify the structure at shallow-level can be useful to real-world applications. In this paper we present a case study of a deployed data-driven system that grows a

[1] these four data items, for example, are available in a product search such as google.com/search?tbm=shop&q=Lauren+black+dress, and also in Amazon's product API service http://www.google.com/search?q=RG_Small.html

product-related terms dictionary by parsing product offering titles in order to improve an affiliate-marketing link insertion service that inserts affiliated hyperlinks to relevant offers.[2].

Although the semantic chunking of offering titles is a simpler task than the chunking of natural language sentences, even in this constrained domain there is no preexisting formal parser available to apply to the task; nor, as we will show in the experiments section, does it appear feasible to manually develop one. Given our pragmatic ambitions, we followed the best-practices approach of training a sequence BIO tagging model [19]. Further, we pursue the approach taken in natural language processing (NLP) of first chunking a text string as a preprocessing step to additional tasks [1]. As Abney writes: "*[I begin] [with an intuition]: [when I read] [a sentence], [I read it] [a chunk] [at a time]*". For us the task of shallow semantic offering title parsing appears to also be naturally commenced with a chunking step. The offering shown earlier, for example, could be semantically segmented as follows: "*[Pro Digital] [black] [Lens Hood] [for] [Canon] [VIXIA] [HF S10], [HF S100] [Flash Camcorders] ([1-year wrty.]) [from] [e-sekuro]*.". Further, just as subsequent work in NLP added the requirement to label segments with their phrase type (e.g. NP-chunk, VP-chunk, etc.) [14], the semantic segments in a title could also be mapped to semantic classes. Given a set of semantic classes such as *product category*/`PC`, *product feature*/`PF`, etc. (which we define in section 2) the sample title can now be further annotated as follows: "[*Pro Digital*]$^{\texttt{BN}}$ [*black*]$^{\texttt{PF}}$ [*Lens Hood*]$^{\texttt{PF}}$ [*for*]$^{\texttt{FT}}$ [*Canon*]$^{\texttt{BN}}$ [*VIXIA*]$^{\texttt{PL}}$ [*HF S10*]$^{\texttt{PI}}$, [*HF S100*]$^{\texttt{PI}}$ [*Flash Camcorders*]$^{\texttt{PC}}$ ([*1-year wrty.*]$^{\texttt{OF}}$) [*from*]$^{\texttt{FT}}$ [*e-sekuro.com*]$^{\texttt{ME}}$".

In NLP a best-practices approach to perform chunking is to use supervised sequence tagging models, such as a linear chain conditional random fields (CRFs) [18]. Because state-of-the-art systems do not generally approximate expert-level performance, even with a significant amount of labeled and unlabeled data [8], most real-word applications need to prepare for error rates that are higher than those achieved by human annotators. One way to address this challenge is to associate a confidence score with each predicted chunk so that only high-confidence predictions are provided to consuming processes. In our case we apply a set of models (a model ensemble) to naturally produce a confidence score.

Given a system that can chunk offerings titles into rankable terms one can begin to apply it to real-world tasks, such as the growing of a domain-specific dictionary of categorized consumer product-related terms. Having such a terms database can be very useful in automation of many consumer e-commerce related tasks. Once a system knows that "*Canon*" is a brand (`BN`), that "*Vixia*" is a product line (`PL`), that "*flash camcorder*" is a product category (`PC`) and that "*high def*" is a product feature (`PF`), then a data-driven system can more easily infer the meaning of a phrase such as "*Canon Vixia high-def flash camcorder*". Unfortunately, updating and managing a terminological dictionary remains a time-consuming process [5]. With hundreds of thousands of product-related terms, it is essential to enhance the dictionary in a cost-effective manner. By parsing hundreds of thousands of titles, we believed, many high-quality candidate terms could be cheaply discovered and inserted into the dictionary.

The remainder of this paper is structured as follows: Section 2 defines the proposed annotation style in detail; Section 3 describes our algorithmic solution to the chunking task; Section 4 describes our solution to the addition of parsed terms into the dictionary, including the use of a crowdsourcing-based filter; Section 5 analyses the parser's intrinsic performance on manually annotated data; Section 6 analyses the return-on-investment from the added terms; and Section 7 concludes the paper with future possible directions.

2. ANNOTATING OFFERING TITLES

In this section we define our proposed annotation framework to the task of chunking product offering titles into their granular terms. Our main challenge is to define a term typing system that balances the requirement to handle titles from a diverse set of industries; to also identify helpful semantic characterization of the offer; and to provide intuitive quidelines to human annotators.

Based on prior work in the modeling of product databases and product data management [9] and through iterative experimentation, we propose that product offering titles can be exhaustively and beneficially decomposed into token substrings from the eight term types described below.

We observe in advance that any given title can reference zero or more of the eight term classes - no class is mandatory, though at least one must be present. Further, a given term can belong to more than one term type (famously the term "*apple*" can refer to a food category, a brand, a merchant, and even a scent-type product feature).

1. A *product identifying term* (`PI`) is a term that identifies any product in the offering. Electronic products, for example, are known to typically include an identifying product code, such as "*LS-606M*, "*4s*", or "*s4*" in their names. Other industries such as publishing and entertainment can use the entire title as a product identifier, such as "*Les Miserables*" (which could ambiguously refer to the book, the movie, or the recorded play by the same title). Finally, industries such as fashion typically do not include any identifying terms in their titles.

2. A *product feature term* (`PF`) is a term that refers to a property of any product mentioned in the offering. Typical product feature terms in offers can refer to size ("*short*","*14-42mm*"), capacity ("*500gb*"), color ("*mustard on black*"), style ("*coupe*"), and fragrance ("*green tea*", "*apple*").

3. A *product category term* (`PC`) is a term that refers to a group of products with some substitutability. Examples include "*hockey helmet*", "*apple*" (the edible type). This class can require some additional level of domain-expertise to recognize, for example, that a "*hockey helmet*" is not substitutable with a "*bike helmet*" while a "*red*$^{\texttt{PF}}$ *helmet*$^{\texttt{PC}}$" can be substituted for a *black*$^{\texttt{PF}}$ *helmet*$^{\texttt{PC}}$" at the category-level.

4. A *product brand term* (`BN`) is a term that refers to a manufacturer of a product. Examples include: "*Apple*" and "*Mercedes Benz*".

[2] an affiliated hyperlink is encoded with sufficient information for a seller to reward the website publisher with some commission related to the value of "introducing" the consumer to the seller. Additional information on the service can be found here http://viglink.com/products/insert

Table 1: Examples of offering titles annotated using our framework. Square brackets indicate multi-token terms

Annotated product offering title	Industry
Tripp$^{\texttt{PI}}$ *LS606M*$^{\texttt{PF}}$ [*600 watt*]$^{\texttt{PC}}$ [*Line Conditioner*]$^{\texttt{PF}}$ [*6 Outlet*]$^{\texttt{PF}}$ [*120 volt*]$^{\texttt{QQ}}$	Electronics
Samsung$^{\texttt{PF}}$ *51-Inch*$^{\texttt{PF}}$ *720p*$^{\texttt{PF}}$ *600Hz*$^{\texttt{PF}}$ *Plasma*$^{\texttt{PF}}$ *HDTV*$^{\texttt{PC}}$ (*Black*$^{\texttt{PF}}$)	Electronics
[*Wheelskins*]$^{\texttt{PF}}$ [*Genuine Deerskin*]$^{\texttt{PC}}$ [*DRIVING GLOVES*]$^{\texttt{PC}}$ - *Tan*$^{\texttt{PF}}$ ([*Size Large*]$^{\texttt{PF}}$)	Automotive
[*1990-1997*]$^{\texttt{PF}}$ [*Mazda*]$^{\texttt{BN}}$ [*Miata*]$^{\texttt{PL}}$ *AxleBack*$^{\texttt{PL}}$ [*Exhaust Bolt*]$^{\texttt{PC}}$ *on*$^{\texttt{FT}}$ *Muffler*$^{\texttt{PC}}$	Automotive
[*Cloudy with a Chance of Meatballs*]$^{\texttt{PI}}$ (*Two-Disc*$^{\texttt{PF}}$ [*Blu-ray / DVD Combo*]$^{\texttt{PF}}$)	Entertainment
[*A Pius Man : A Holy Thriller*]$^{\texttt{PI}}$ ([*The Pius Trilogy*]$^{\texttt{PL}}$) ([*PF|Volume 1*]$^{\texttt{PF}}$)	Books
[*Amazing Herbs*]$^{\texttt{BN}}$ [*Black Seed*]$^{\texttt{PF}}$ [*Cold-Pressed Oil*]$^{\texttt{PC}}$ - *32oz*$^{\texttt{PF}}$	Health & Beauty
([*180 days wrty.*]$^{\texttt{OF}}$) *ZeroLemon*$^{\texttt{BN}}$ *LG*$^{\texttt{BN}}$ *Nexus*$^{\texttt{PL}}$ *4*$^{\texttt{PI}}$ *Juicer*$^{\texttt{PL}}$ [*Battery Case*]$^{\texttt{PC}}$	Mobile
[*WORLDS ONLY*]$^{\texttt{OF}}$ *NEXUS*$^{\texttt{PL}}$ *4*$^{\texttt{PI}}$ [*BATTERY CASE*]$^{\texttt{PC}}$ (*LG-N4-BattCase-black*$^{\texttt{PI}}$)	Mobile
[*Kenneth Cole*]$^{\texttt{BN}}$ *REACTION*$^{\texttt{PL}}$ *men's*$^{\texttt{PF}}$ *Cufflinks*$^{\texttt{PC}}$, *Silver*$^{\texttt{PF}}$, [*One Size*]$^{\texttt{PF}}$	Fashion

5. A *product line term* (`PL`) is a term that refers to a narrowly defined sub-brand. The term must generally be prefixable by the brand name, such as how "*Galaxy*" can be rewritten as *Samsung Galaxy*$^{\text{PL}}$ *S4*) and "*W126*" as *1989 Mercedes Benz W126*$^{\text{PL}}$ *v8 coupe.* The product line term type also accounts for celebrities whose name is being branded with a product, such as "*Alicia Keys*$^{\text{PL}}$ *Rebook sneakers*". This category could be merged into the brand type, but we opted to separate it at this step.

6. A *merchant term* (`ME`) is a term that refers to a merchant/seller mentioned in the title, or the given offer of possibly of a bundled item, such as a warranty provider.

7. An *offering feature term* (`OF`) is a term added by the merchant to differentiate their offering from some other seller's offering of the same underlying product. Common terms include warranty periods, temporary discounts, and subjective attributes ("*lovely*", "*stylish*").

8. A *functional term* (`FT`) is a term that plays a syntactic role in the title. Examples include: "*of*", "*and*", "*without*", "*with*", and "*includes*".

Several examples of annotated product offering titles are presented in table 1.

3. AUTOMATICALLY CHUNKING TITLES

We opted to apply the best-practices approach of BIO tagging, in which a string is first tokenized, and then a sequential classification/tagging model is required to tag each token with either a `B`, `I`, or `O` depending on whether the given token either *begins* a chunk, is *inside* a chunk, or is *outside* of any chunk. When the task also requires that chunks be classified then the labels are updated encode the relevant term class. In our case, with eight term types, each token must be labeled with one of seventeen possible labels: {`O`, `B-PI`, `I-PI`, `B-PF`, `I-PF`, `B-PC`, ..., `I-FT`}. Specifically, we train a linear-chain conditional random field model on a manually annotated training dataset, as originally proposed in [18].

The remaining decision for applying a supervised model is to select the predictor features associated with each token and the size of the before/after token window. Here again, we leverage the best-practices of features reported to be helpful in NP-chunking [18] and named entity recognition (NER) [11, 15]. Indeed, most of features that we use are contained in the published baseline recognizer for the ICDM-2012 CPROD1 contest[3] [10].

For our task, which involves shorter text items and more varied patterns, we modified the `eosTokFeat.pl`[4] from the contest with three additional features. Two of the new features simply the offset of the token from the edge of the token string. This feature-type is used in sentence-centric chunking and was not used in the `CPROD1` contest. The other new feature is global feature based on the offerings industry (such as Electronics, Books, or Automotive). As seen in table 1, the annotation pattern for an offering's title can differ significantly based on their industry - book and movies can have long product identifiers. By introducing this feature we hope that the model can account for these patterns.

The new features are further described below:

1. *Previous Tokens* (`LEFTOFF`): The number of tokens into the title. For example, the fourth token in a title receives the label `4`.

2. *Remaining Tokens* (`RIGHTOFF`): The number of tokens remaining in the title. For example, the fourth-to-last token receives the label `4`.

3. *Industry* (`INDUSTRY`): The code of the titleś industry. For example, every token of a book title receives the code `BK`.

Clearly there are additional state-of-the-art techniques that we could apply, but we believed (given our Agile develop-

[3] the original program to generate those features `eosTokFeat.pl` can be found at http://kaggle.com/c/cprod1/forums/t/2287/crf-based-baseline2-published

[4] the updated featurization program and all code used in the experimental study is available at https://dropbox.com/sh/q8cyv0wfuyg0han/sXZ95rOUC8

ment mindset) that this best-practices approach should be good enough to prove the business value of the solution.

3.1 Sequential Tagging Model Ensembles

As presented, the trained sequence tagging model does not provide a confidence likelihood estimate with each predicted segment. Such an estimate can be helpful for many downstream tasks that may use title parser. For our dictionary population task, for example, where we require high-precision predictions, a confidence ranking score enables us to better expend our resources on predictions that will be more likely to be accepted into the dictionary. While there are methods to extract likelihood estimates from a trained CRF model, such as the Constrained Forward Backward algorithm (CFB) [3] that computes the total weight of all paths constrained by the states of our segments, these methods are not generally available in CRF software libraries and can increase the processing time.

An alternative approach that we investigate is to train an ensemble of models, and to report a confidence score based on the number of models that agree for each prediction [12]. Unfortunately, there is no agreed-on best-practice approach to combining the predictions of a set of sequence tagging models[5]. However, a simple voting approach can be naturally implemented by ranking predictions based on the proportion of models in agreement. If all models agree on a segment then associate a score of 1; if only four fifthś of the models agree then associate a confidence score of 0.8. In our case, we opted to use the five models trained during a five-fold cross validation analysis which reports an in-sample F_1 (our "unit test" of each iteration is that the F_1 be higher than a threshold of 50).

4. DICTIONARY UPDATES

Now that we can parse offering titles and rank the predictions, we are ready to apply it to a real-world problem. Several business processes could benefit from this capability at our organization. The first that we explore is the addition of terms into a domain-specific dictionary that is referenced by our hyperlink-insertion service. A more complete dictionary improves the chances of recognizing mentions of product related terms in text. Managing such a large dictionary however can be a time-consuming process [16], especially if a low error-rate is required. In our case errors can result in bad user experiences to the visitors of our customer's webpages. If a term such as "*Christmas*" were accidentally inserted into our dictionary as a product category (`PC`) rather than, say, of a product feature (`PF`), then the term becomes a viable candidate for hyperlink insertion when encountered in a webpage, and then be linked to some random (if still Christmassy) product. Further, the value of adding terms diminishes over time as fewer and fewer high-value terms remain and more esoteric terms are added. Given the expectation of erroneous predictions, we opted to include a manual review step but of such high-confidence terms that the time spent per term would be very low.

We start the addition process by retrieving the titles of the most popularly clicked links to seller offerings[6] because popular destinations are more likely to contain high-value terms that are more likely to result in the insertion of additional high-value links. Each week we parse a set of approximately ten-thousand titles and retain the terms with a confidence score higher than 50% (more than half the ensemble models agree on the prediction) that are not yet present in the dictionary. This step currently produces approximately two-thousand candidate terms per week, such as: {`term=`"*table saw*", `type`=`PC`, `industry`=`HG`, `score`=0.80 `quantity`=72}, ranked in descending order of their quantity (more popular terms should be entered ahead of less popular ones).

4.1 Crowd-sourced Candidate Review

Because of the requirement for low-error rate of dictionary term addition and the low marginal value per individual addition, we use a web-based crowdsourcing service to manually review each of these terms to further filter the terms that are likely to be correct[7]. These online marketplaces match *human intelligence micro-tasks* (HITs) posted by organizations with workers, and one of the areas that these services have been successfully applied to is natural language annotation [2]. A best-practice approach is to have w-workers, with typically $2 \leq w \leq 3$, perform each micro-task in order to (as with an ensemble) determine which predictions have high-confidence and to more readily identify free-riding workers. We opted for $w = 3$ workers per task.

The basic questionnaire format that we selected was for the worker to provide feedback on whether a given term was of the given offering-term type (feature, category, etc.) within a given industry (electronics, books, etc.). For example: "*Is the term two-disk a product feature in the Arts & Entertainment industry?*" We further experimented with whether to make it a binary-classification task (`TRUE` or `FALSE`) or to make it a multi-label classification one (`Definitely`, `Likely`, `Maybe`, `Unlikely`, and `No`). We opted for the multi-label approach because it resulted in slightly higher F_1, where agreement required two of the workers to select either `Definitely` or `Likely` for accepted predictions. Approximately six hundred terms currently survive this filtering step. The accepted terms are now manually reviewed by an internal annotator to our organization, but by now all of them are entered into the dictionary.

In terms of the fine-tuning of the crowdsourced solution, we tested three different levels of payments per micro-task: \$0.01, \$0.02 and \$0.05 and found that a payment of \$0.01 achieved the highest number of additional terms per investment. In the hope of detecting and rejecting free-riders (who sloppily speed through HITs) we identify and block any worker who replies differently than the two other annotators more than half of the time on a batch in which they attempt more than thirty HITs.

4.2 Active Iterative (Re)Training

As with many real-world predictive modeling applications, our situation benefited from continuous improvement of our models [17]. Rather than creating a single training dataset to train our model from, we iteratively updated the training set with titles in which the existing model performed poorly on. Every week we select approximately one-hundred titles that contain a term confidently predicted by the ensemble-based parser but that a majority of the crowdsourced workers agreed on as being an incorrect prediction. The dataset

[5] this topic appears to be an open research problem

[6] our click-log contains billions of seller URLs from which we can readily identify popular offerings

[7] http://mturk.amazon.com

produced by this process is further discussed in the next section on experiments.

5. INTRINSIC PERFORMANCE ANALYSIS

This section evaluates the performance of the shallow-semantic parser on a pre-annotated dataset, while the next section evaluates the impact of using the parser on a real-world application.

Our focus of the analysis is on ways to better structure the trained parser such that it produces more accurate predictions from the provided training data.[8]

5.1 Datasets

For the evaluation we used a dataset with 2,437 annotated product titles[9]. As in most real-world settings the annotated dataset evolved and grew over time, from a starting seed of records to test the feasibility of training a parser with useful accuracy. Afterwards, titles were added as described earlier in the section on iterative retraining[10]. Each record is categorized into one of nineteen typically high-level industries based on an internally managed mapping table (for example, "Computers" → "CE").

Table 2 shows the distribution of titles and of term types by the different industry labels[11]. The table shows that there is a clear skew in record distributions - with the categories of Consumer Electronics and Book accounting for more than half of the records (750+566), and product features (PF) dominating the types of terms annotated. From the product term distributions by industry we further notice a few outliers: Books and Arts & Entertainment are dominated by `PI` terms; while alternatively Fashion and Food records have relatively few `PI` terms.

5.2 Performance Measure

A standard measure of performance for systems that must segment and label text is the harmonic mean between the precision (P) and recall (R) of all predictions: $F_1 = \frac{2RP}{R+P}$. To calculate this measure we used the widely use evaluation tool from the CoNLL-2000 Shared Task [12].

5.3 Cross-validation (+ by Industry)

We performed a 5-fold cross-valuation study on the dataset. Table 3 reports the average F_1 performance of five CRF models trained and tested on different portions of the annoated data. The first row reports the F_1 performance against "*All*" test records ($F_1 = 57.7$) and against the records of the different product categories.

We had suspected that performance would vary significantly by the industry feature. This hypothesis was confirmed by the other performance results on this row. Notice how records from the books (BK) industry, , despite it having the second most number of records, performs particularly poorly ($F_1 = 41.2$). Given the relatively weak performance on some industries, we trained independent models with only records from each industry to see the effect of isolating the title data into clusters. As the table shows, two industries significantly improved F_1 performance despite the fewer available training records to the models: Books (from 41.2 to 82.0) and Arts & Entertainment (from 48.8 to 61.3).

At this point we could have decided to use three models: one for Books, on for Arts & Entertainment, and one based on *All* the training data.

Next, however we combined the `AE` and `BK` records into one training set, `BK&AE`, to determine whether these two categories could be treated as a single record cluster. Performance did improve but only for the `AE` records. This increase may be due to the relatively few `AE` records benefiting from the large number of `BK` records, while `BK` has sufficient records to diverge into its own model [13].

We tested on final permutation in this vein: Retaining the `BK` and `AE` training records to contribute to the predictions on the other industry records may be sub-optimal, so we further tested removing these records to create a `not BK|AE` model. This parsing model equaled the performance of the "All" model with lower performance on the Jewelry (`JW`) titles and higher performance on (`FS`) and (`AU`) titles.

Based on this analysis, we opted to use three models: `BK` then the title was known to be of that industry, `BK&AE` when the title was known to be from the `AE` industry, and *All* otherwise.[14].

5.4 Baseline Comparison

A natural baseline algorithm to the task is simply insert terms already found in an existing dictionary [15]. Given the overlap in terms (recall the many term types that a term such as "*apple*" can take) and the requirement that annotation occurs at a granular level (e.g. the token *2* may be from a product identifier, such as in *iPad* <u>*2*</u> or be a product feature). After some experimentation, we opted for an iterative labeling method that proceeds from one term type, to the next. The basic pattern was to first apply brands, then product lines, then several product features, then products, black list and offer features within the record's industry [16].

The baseline algorithm describe above achieves a 30.4 F_1 overall, with high scores on FashionFS and ElectronicsCE records (40.0 and 35.7 respectively) and nearly zero on Books and Arts & Entertainment records because the dictionary only contains famous books and movies which are not found in the annotated data.

5.5 Feature Ablation Analysis

Because most of the used predictor features have been relatively well tested in other published research and none of the features are particularly time-consuming to calculate,

[8] we used MALLET as our CRF toolkit mallet.cs.umass.edu

[9] the data can be downloaded from https://dropbox.com/sh/q8cyv0wfuyg0han/sXZ95rOUC8

[10] Because of the iterative history of the datasetś development, the records are biased towards more challenging cases.

[11] The `other` label refers to titles whose merchant category did not map to one of our existing categories; the `other` label refers to titles whose merchant did not appear to provide a categorization whatsoever

[12] available and described at http://www.cnts.ua.ac.be/conll2000/chunking/output.html

[13] ideally the sequential tagging model would automatically detect and adjust for the importance of these attribute values and so avoid this time consuming and error prone creation of a composite model

[14] a demo of the service can be found at http://www.gabormelli.com/Projects/PTPv1

[15] we used an internal dictionary with nearly five-hundred thousand terms, but readers can use the term dictionary made publicly available for the CPROD1 contest

[16] the baseline program `PTPbaseline.pl` is available in the repository

Table 2: Distribution of titles in the gold dataset by industry type

	Industry	Code	Recs.	PF	PI	PC	BN	PL	BL	OF
1	Electronics	CE	760	1,618	667	596	332	283	155	3
2	Book	BK	566	103	554	7	8	68	10	0
3	Home&Garden	HG	178	502	94	231	123	28	76	2
4	Phones&Mobile	CM	127	479	151	177	80	162	121	4
5	Hobbies&Toys	HO	122	189	90	89	65	44	41	0
6	Jewelry	JW	96	253	45	50	30	22	50	2
7	Health&Beauty	HB	90	293	33	119	58	15	61	1
8	Fashion	FS	76	190	24	67	47	25	18	0
9	Automotive	AU	63	175	43	105	68	26	30	2
10	Arts&Ent.	AE	58	44	64	3	7	12	13	0
11	Gaming	GM	55	44	41	10	9	20	10	0
12	Camera&Photo.	CP	54	181	109	80	33	21	19	0
13	Sports&Fit.	SF	54	151	22	79	45	12	35	0
14	*not avail.*	na	37	77	20	33	9	8	18	5
15	Food	FD	32	87	4	41	27	1	10	0
16	Families	FB	31	82	11	41	25	14	15	0
17	Musician	MM	19	47	9	19	15	9	15	0
18	Pets	PT	12	52	6	17	11	4	10	2
19	*other*	OT	6	25	0	12	7	1	11	0

Table 3: 5-fold cross-validated F_1 performance by training and/or testing on titles from the 10 most frequent industries in the data. The statistically significant best scores are bolded.

Training Set	Test Set Category									
training set	All	CE	BK	HG	CM	HO	JW	HB	FS	AU	AE
All	**57.7**	**64.3**	41.2	**54.6**	**58.3**	48.0	**60.3**	**51.6**	55.0	46.7	48.8
CE	47.1	63.3	6.6	46.6	48.7	34.8	35.8	34.6	36.6	42.7	27.2
BK	11.1	4.0	**82.0**	2.8	4.2	2.6	2.1	3.5	3.8	2.8	37.5
HG	34.9	41.4	7.5	50.0	28.9	32.8	36.1	32.3	32.3	32.9	28.1
CM	41.2	48.9	9.1	39.7	56.5	29.8	32.5	33.1	31.0	35.7	31.5
HO	27.3	30.4	4.1	37.3	22.3	43.5	28.7	29.6	25.7	27.7	18.9
JW	27.1	30.2	7.2	31.8	19.5	23.9	56.2	29.3	31.9	26.1	15.9
HB	27.9	30.9	4.7	35.1	23.3	29.2	27.3	45.8	28.1	24.3	23.6
FS	27.0	32.0	5.8	31.7	22.1	23.1	32.1	27.6	52.3	25.0	27.8
AU	31.1	40.0	4.8	33.9	24.8	26.8	31.4	25.0	26.7	38.2	19.2
AE	13.3	9.0	59.2	6.3	7.5	7.4	4.3	9.7	11.7	4.6	61.3
BK&AE	12.9	6.2	65.2	3.6	6.0	4.0	3.0	5.2	4.9	3.3	**65.3**
not BK|AE	55.8	**64.6**	18.1	**54.3**	**59.0**	**49.3**	58.4	**52.1**	**56.3**	**49.2**	34.9

Table 4: F_1 learning curve of the *All* parser

titles	2%	5%	20%	33%	67%	80%	90%
F_1	29.1	39.8	48.2	53.1	56.9	57.7	58.2

we did not perform a thorough analysis of which features could be safely removed from the process. We did however analyze the impact of the "global" `INDUSTRY` feature that we introduced that assist the model in detecting the different patterns that apply to some of the different industries (such as long product identifier in books). What also motivated this analysis is the fact that the Books/BK and Arts&Entertainment/AE industry titles had weak performance. Table 3 suggests that the feature is not informative. This strongly suggests that the feature is not properly informing the models of the strong role that industry categories have on annotation patterns. In the future we hope to investigate the work by (Krishnan & Manning, 2006) [7] that uses two phases to introduce global features.

5.6 Learning Curve Analysis

The learning curve in table 4, in which cross-validation is used to estimate performance at different proportions of held-out data, suggests that performance has begun to plateau at the current number of training records in the annotated dataset. Given the current feature set, many more annotated titles (possibly through self-labeling approaches) would be required to noticeably impact F_1 performance. Recently, due to this analysis, we have begun to direct our annotation effort towards titles from the industries with fewer records.

6. REAL-WORLD PERFORMANCE

Given the ability to parse product titles we can now evaluate the impact on a mission-critical service. As already described in sections 3 and 4, on a weekly basis we apply the parser to a large set of product titles and filter the resulting terms with a crowdsourced service, such that approximately four-hundred terms are added weekly to the dictionary. With each batch of added terms our hyperlink insertion service has more information about the kinds of terms that relate to consumer products and may therefore be possible candidates for the insertion of a hyperlink (so long as other business rules and optimization rules are satisfied). For example, recently the process discovered that the dictionary was missing terms such as: {"table saw", "running lights", "floor mats", "suspension system", "rear derailleur"} ∈ `PC`; {"Emotiva"} ∈ `BN`; and {"Xonar", "Quadro"} ∈ `BN`. After the addition of these terms into our dictionary the linking service can be more confident that the presence of these terms in a webpage indicate a valid hyperlinkable product-related mention. For example, when the service sees a passage such as "*... the Ryobi table saw was able to ...* " in an appropriate place on a website (based on business rule restrictions) then, just like a website publisher might do manually on their own, the statistical recognizer used by the service can substitute the plain text "*Ryobi table saw*" with an affiliated hyperlink to a page that offers that type of tables saw [17] which may earn the publisher a commission each time that a visitor to their page interested in that type of product clicked on the link and soon after decided to make a purchase from the seller at the end of the link.

But, were the significant costs spent to introduce this semi-automated term-adding capability a worthwhile expense? In terms of costs and benefits, the costs to deliver the service had several dimensions, including: the time to prototype the solution, the effort to annotate the data, the effort to program a reliable weekly production system that includes crowdsourced annotating, the effort to evaluate performance, and finally the expenses of the new temporary equipment to train and evaluate models. The benefit is the increased revenue generated by the link insertion server. This can be calculated from the cumulative value each added term. Our link-insertion service has been in operation for several years so were able to retrieve significant historical evidence of past click performance for each term prior to its insertion, and from this estimate future click behavior for the term. The term "*table saw*", for example, which, being composed of a common noun "*table*" and verb "*saw*", was likely avoided by the service prior to the addition of the term, could now be more readily inserted (though still only in appropriate textual context). One unknown was whether the increase in insertions would also result in an increase in clicks - it may be that missing terms were not valuable with respect to clicks. We modeled and extrapolated the clicks for each term based on click data before and after their insertion, and from this estimated that how many weeks it would require for the incremental value (if any) would pay for the costs and from then on return a positive investment. We determined that the effort recovered its expense within the first eleven weeks of its operation, and would also continue to deliver the incremental value to our organizations and our customers who benefit from the commission associated from the previously absent links. Finally, each new batch of terms continues to identify high-value terms though, as expected, fewer with each batch. The service could also allow our organization to create one of the most comprehensive product-related dictionaries in the market quickly and cost-effectively. Finally, our organization can now apply a new capability to other mission-critical processes.

7. RELATED WORK

Given the commercial opportunity related to understanding product offerings on the Web, there has been significant published research on the automated identification of product properties from textual information. We review several publications that most closely related to our task and solution.

7.1 (Ghani & al, 2006)

One of the first published investigations in our task by Ghani & al [4] which applied a semi-supervised approach to the task of identifying attribute-value pairs in short product-related phrases.

We include three examples of the types of strings that they sought to parse:

1. "*Extended Torsion bar*"
2. "*Imported*"
3. "*Contains 2 BIOflex concentric circle magnets*"

These text items are related but differ significantly from the product offering titles listed in table 1. These short text

[17] for example to amazon.com/dp/B0050RBHQE

items instead appear to be drawn from the lists of product specifications that are commonly found on the product offering description pages.

Next, they propose a general annotation structure that attempts to identify attribute-value pairs in the text. This approach is very general. It can be applied to almost all other domains, not only product offering related text. Their annotation framework involves the application of three labels to text segments: *attribute*(`ATTR`), *value*(`VAL`), and *neither*(`NA`), where (a `TRUE` value is inserted when no actual value is present in the text). As we can see in the annotated versions of the strings above, the attribute-value pairs can be readily extracted:

1. *Extended*$^{\mathrm{NA}}$ *Torsion*$^{\mathrm{VAL.}}$ *bar*$^{\mathrm{ATTR.}}$
2. *Imported*$^{\mathrm{ATTR.}}$ `TRUE`$^{\mathrm{VAL.}}$
3. *Contains*$^{\mathrm{NA}}$ *2*$^{\mathrm{VAL.}}$ *BIOflex concentric circle magnet*$^{\mathrm{ATTR.}}$

Unfortunately, their representation cannot be easily translated into our needs. As can be seen in their examples, sometimes chunks labeled as attributes and as values can both be mapped to product features. The representation, as (Ghani & al, 2006) mention, results in a significant disagreement between human annotators in the "correct" annotation structure for many cases. The example they offer is that of "*Audio/JPEG navigation menu*" which can be naturally annotated in three different ways, while for us the string naturally becomes a product feature (`PF`). Contrast their annotation structure to ours on their three examples:

1. *Extended*$^{\mathrm{PF}}$ *Torsion bar*$^{\mathrm{PC}}$
2. *Imported*$^{\mathrm{OF}}$
3. *Contains*$^{\mathrm{FT}}$ *2*$^{\mathrm{OF}}$ *BIOflex*$^{\mathrm{BN}}$ [*concentric circle*]$^{\mathrm{PF}}$ *magnet*$^{\mathrm{PC}}$.

Aside from the annotation structure, we notice also that their input strings are already in a succinct form relative to the long descriptive product titles in our task which often contain many values with an implicit attribute. For example, when a color is mentioned in a title the color term (say *yellow*) is almost never preceded or followed by a term such as "color".

Finally, the proposed algorithm which combines semi- supervised co-training along with expectation-maximization (EM) would involve significant programming effort and has not been successfully applied in other case studies, so we did not attempt to implement it as a baseline.

7.2 (Putthividhya & Hu, 2011)

The work that most closely approximates ours, both in terms of task and approach appears to be the one by Putthividhya & Hu [13]. They propose the use of an iterative online supervised CRF approach to the task of labeling descriptive product offering titles.

Their annotation framework involves five labels: *brand* (`B`), *garment type* (`G`), *size* (`S`), *color* (`C`), and *not applicable* (`NA`). These labels are intentionally tailored for products from the fashion industry (from eBay's "clothing and shoes" products).

Below are two sample titles that they provide:

1. *NEXT Blue Petite Bootcut jeans size 12 BNWT*[18]
2. *Paul Smith Osmo White Plimsoll Trainers - UK 6 RRP*[19] *: £100*

Followed by their reported annotations:

1. *NEXT*$^{\mathrm{B}}$ *Blue*$^{\mathrm{C}}$ *Petite*$^{\mathrm{NA}}$ *Bootcut*$^{\mathrm{NA}}$ *jeans*$^{\mathrm{G}}$ [*size 12*]$^{\mathrm{S}}$ *BNWT*$^{\mathrm{NA}}$
2. [*Paul Smith*]$^{\mathrm{B}}$ *Osmo*$^{\mathrm{NA}}$ *White*$^{\mathtt{C}}$ *Plimsoll*$^{\mathtt{NA}}$ *Trainers*$^{\mathtt{G}}$ -$^{\mathtt{NA}}$ *UK6*$^{\mathrm{S}}$ *RRP*$^{\mathrm{NA}}$ *:*$^{\mathrm{NA}}$ £$^{\mathrm{NA}}$ *100*$^{\mathrm{NA}}$

Their labels map roughly to ours as follows: `B=BN`, `G` ∈ `PC`, `S` ∈ `PF`, `C` ∈ `PF`, and `NA` ∈ `FT,PL,ME,OF`.

Our annotation of their two examples is as follows:

1. *NEXT*$^{\mathrm{BN}}$ *Blue*$^{\mathrm{PF}}$ *Petite*$^{\mathrm{PF}}$ *Bootcut*$^{\mathrm{PF}}$ *jeans*$^{\mathrm{PC}}$ [*size 12*]$^{\mathrm{PF}}$ *BNWT*$^{\mathrm{NA}}$
2. [*Paul Smith*]$^{\mathrm{BN}}$ *Osmo*$^{\mathrm{NA}}$ *White*$^{\mathtt{C}}$ *Plimsoll*$^{\mathtt{NA}}$ *Trainers*$^{\mathtt{PC}}$ - *UK6*$^{\mathrm{PF}}$ *RRP*$^{\mathrm{OF}}$: [*£100*]$^{\mathrm{OF}}$

Our proposal uses a broader term definition of product feature that encompasses their size and color labels (and is therefore applicable to other industries, but loses the granual information). We also use product line (`PL`) which can appear in Fashion products (such as in *Ralph Lauren* - ***Polo*** and *Tory Burch* - ***Reva***). Additionally we use the `Merchant`, `Offering Feature`, and `Function Term` and avoid the use of a less informative `NA`.

In terms of algorithm design, they also propose the use of a supervised solution and a bootstrapped solution that grows their gazetteer (seed list). Finally, they do not report on the real-world impact of their work.

7.3 (Köpcke & al, 2012)

Finally, Köpcke & al [6] address the focused task of identifying the code/identifiers in largely electronics product offering titles using information theoretic approaches. This task is too restrictive to aid in our more general needs.

8. CONCLUSION

In this paper we reviewed the design and impact of a system that can identify and label the low-level semantic structure of the billions of product offering titles that pervade consumer e-commerce websites. We leverage the current best-practices approach of supervised BIO-tagging via a trained linear CRF to train a predictive model, though we extend the approach to involve separate models based on industry information along with model ensembles that produces a voting-based ranking score. We apply the trained parser to many offering titles on a regular basis in order to grow a domain-specific dictionary of offering-related terms. To ensure a low error-rate, each candidate term is reviewed via a crowdsourcing service. Terms that pass this filter are added to the dictionary, while terms that are strongly rejected are used to select offering titles to be manually annotated and added to the training dataset for subsequent trained parsers. Finally, we assess the impact of the added terms to our affiliate hyperlink-insertion service and we find

[18] `BNWT` in eBay is a contraction of "*brand new with tags*"

[19] `RRP` in eBay is a contraction of "*recommended retail price*"

that the new capability has both paid for itself, and continues to add value to our organization, to our website-owning customers, and to their visitors.

Several enhancements to the deployed system are possible. More leading-edge mechanisms for sequential tagging and active learning should be considered such as `BILOU` (instead of `BIO`) tagging [15] as well as the two-phased approach in [7] to include global features.

Aside from enhancing the semantic parser's accuracy, there are several additional applications that we would like to deploy. These include 1) the ability to perform record linkage on product records to find substitutable products with high-accuracy and likely also data cleaning based on contradictions with what the product offerings titles suggest. Finally, these applications may also benefit from the ability to create a full parse tree from a title. For now, through this case-study, we have demonstrated the cost-effective feasibility of data-driven shallow semantic parsing of product offering titles (for better automatic hyperlink insertion).

9. ACKNOWLEDGMENTS

The author would like to thank the rest of the team at VigLink for fruitful discussions and aid with the annotation.

10. REFERENCES

[1] S. P. Abney. *Parsing by chunks.* 1989.

[2] C. Callison-Burch and M. Dredze. Creating speech and language data with amazon's mechanical turk. In *Proceedings of the NAACL HLT 2010 Workshop on Creating Speech and Language Data with Amazon's Mechanical Turk*, 2010.

[3] A. Culotta and A. McCallum. Confidence estimation for information extraction. In *Proceedings of HLT-NAACL 2004*, 2004.

[4] R. Ghani, K. Probst, Y. Liu, M. Krema, and A. Fano. Text mining for product attribute extraction. *ACM SIGKDD Explorations Newsletter*, 8(1), 2006.

[5] T. Gornostay. Terminology management in real use. In *Proceedings of the 5th International Conference Applied Linguistics in Science and Education*, 2010.

[6] H. Köpcke, A. Thor, S. Thomas, and E. Rahm. Tailoring entity resolution for matching product offers. In *Proceedings of the 15th International Conference on Extending Database Technology*, 2012.

[7] V. Krishnan and C. D. Manning. An effective two-stage model for exploiting non-local dependencies in named entity recognition. In *Proceedings of the 2006 COLING and ACL conferences*, 2006.

[8] X. Liu, S. Zhang, F. Wei, and M. Zhou. Recognizing named entities in tweets. In *Proceedings of the 2011 ACL conference*, 2011.

[9] G. Melli and J. McQuinn. Requirements specification using fact-oriented modeling: A case study and generalization. In *Object-Role Modeling Workshop at OTM 2008*, 2008.

[10] G. Melli and C. Romming. An overview of the cprod1 contest on consumer product recognition within user generated postings and normalization against a large product catalog. In *Data Mining Workshops (ICDMW), 2012 IEEE 12th International Conference on*, pages 861–864. IEEE, 2012.

[11] D. Nadeau and S. Sekine. A survey of named entity recognition and classification. *Lingvisticae Investigationes*, 30(1), 2007.

[12] R. Polikar. Ensemble based systems in decision making. *Circuits and Systems Magazine, IEEE*, 6(3), 2006.

[13] D. P. Putthividhya and J. Hu. Bootstrapped named entity recognition for product attribute extraction. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 1557–1567, 2011.

[14] L. A. Ramshaw and M. P. Marcus. Text chunking using transformation-based learning. In *Proceedings of the Third ACL Workshop on Very Large Corpora*, 1999.

[15] L. Ratinov and D. Roth. Design challenges and misconceptions in named entity recognition. In *Proceedings of the 2009 Computational Natural Language Learning conference*, 2009.

[16] F. Sclano and P. Velardi. Termextractor: a web application to learn the shared terminology of emergent web communities. In *Enterprise Interoperability II.* 2007.

[17] B. Settles and M. Craven. An analysis of active learning strategies for sequence labeling tasks. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, 2008.

[18] F. Sha and F. Pereira. Shallow parsing with conditional random fields. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1*, 2003.

[19] W. Wong, W. Liu, and M. Bennamoun. Ontology learning from text: A look back and into the future. *ACM Computing Surveys (CSUR)*, 44(4), 2012.