Recognizing Salient Entities in Shopping Queries

Zornitsa Kozareva, Qi Li, Ke Zhai and Weiwei Guo
Yahoo!
701 First Avenue
Sunnyvale, CA 94089
zornitsa@kozareva.com
{lqi|kzhai}@yahoo-inc.com
weiwei@cs.columbia.edu

Abstract

Over the past decade, e-Commerce has rapidly grown enabling customers to purchase products with the click of a button. But to be able to do so, one has to understand the semantics of a user query and identify that in *digital lifestyle tv*, *digital lifestyle* is a brand and *tv* is a product.

In this paper, we develop a series of structured prediction algorithms for semantic tagging of shopping queries with the *product*, *brand*, *model* and *product family* types. We model wide variety of features and show an alternative way to capture knowledge base information using embeddings. We conduct an extensive study over $37,000$ manually annotated queries and report performance of 90.92 F_1 independent of the query length.

1 Introduction

Recent study shows that yearly e-Commerce sales in the U.S. top 100 Billion (Fulgoni, 2014). This leads to substantially increased interest in building semantic taggers that can accurately recognize *product*, *brand*, *model* and *product family* types in shopping queries to better understand and match the needs of online shoppers.

Despite the necessity for semantic understanding, yet most widely used approaches for product retrieval categorize the query and the offer (Kozareva, 2015) into a shopping taxonomy and use the predicted category as a proxy for retrieving the relevant products. Unfortunately, such procedure falls short and leads to inaccurate product retrieval. Recent efforts (Manshadi and Li, 2009; Li, 2010) focused on building CRF taggers that recognize basic entity types in shopping query such as *brands*, *types* and *models*. (Li, 2010) conducted a study over 4000 shopping queries and showed promising results when huge knowledge bases are present. (Paşca and Van Durme, 2008; Kozareva et al., 2008; Kozareva and Hovy, 2010) focused on using Hearst patterns (Hearst, 1992) to learn semantic lexicons. While such methods are promising, they cannot be used to recognize all product entities in a query. In parallel to the semantic query understanding task, there have been semantic tagging efforts on the product offer side. (Putthividhya and Hu, 2011) recognize *brand*, *size* and *color* entities in eBay product offers, while (Kannan et al., 2011) recognized similar fields in Bing product catalogs.

Despite these efforts, to date there are three important questions, which have not been answered, but we address in our work. (1) *What is an alternative method when product knowledge bases are not present?* (2) *Is the performance of the semantic taggers agnostic to the query length?* (3) *Can we minimize manual feature engineering for shopping query log tagging using neural networks?*

The main contributions of the paper are:

- Building semantic tagging framework for shopping queries.
- Leveraging missing knowledge base entries through word embeddings learned on large amount of unlabeled query logs.
- Annotating $37,000$ shopping queries with *product*, *brand*, *model* and *product family* entity types.
- Conducting a comparative and efficiency study of multiple structured prediction algorithms and settings.
- Showing that long short-term memory networks reaches the best performance of 90.92 F_1 and is agnostic to query length.

Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, pages 107–111,
Berlin, Germany, August 7-12, 2016. ©2016 Association for Computational Linguistics

2 Problem Formulation and Modeling

2.1 Task Definition

We define our task as given a shopping query identify and classify all segments that are *product*, *brand*, *product family* and *model*, where:
-**Product** is generic term(s) for goods not specific to a particular manufacturer (e.g. *shirts*).
-**Brand** is the actual name of the product manufacturer (e.g. *Calvin Klein*).
-**Product Family** is a brand-specific grouping of products sharing the same product (e.g. Samsung *Galaxy*).
-**Model** is used by manufacturer to distinguish variations (e.g. for the brand Lexus has *IS* product family, which has model *200t* and *300 F Sport*).

For modeling, we denote with $\mathcal{T} = \{\perp, t_1, t_2, \ldots, t_K\}$ the whole *label space*, where $\perp$ indicates a word that is not a part of an entity and t_i stands for an entity category. The tagging models have to recognize the following types *product*, *brand*, *model*, *product family* and $\perp$ (other) using the BIO schema (Tjong Kim Sang, 2002).

We denote as $\mathbf{x} = (x_1, x_2, \ldots, x_M)$ a shopping query of length M. The objective is to find the best configuration $\hat{\mathbf{y}}$ such that:

$$\hat{\mathbf{y}} = \arg\max_{\mathbf{y}} p(\mathbf{y}|\mathbf{x}),$$

where $\mathbf{y}$=$(y_1, y_2, ..., y_N)$ ($N \leq M$) are the shopping query segments labeled with their corresponding entity category. Each segment y_i corresponds to a triple $\langle b_i, e_i, t_i \rangle$ indicating the start index b_i and end index e_i of the sequence followed by the entity category $t_i \in \mathcal{T}$. When $t_i = \perp$, the segment contains only one word.

2.2 Structured Prediction Models

To tackle the shopping tagging problem of query logs, we use Conditional Random Fields (Lafferty et al., 2001, CRF)[1], learning to search (Daumé III et al., 2009, SEARN)[2], structured perceptron (Collins, 2002, STRUCTPERCEPTRON) and a long short-term memory networks extended by CRF layer (Hochreiter and Schmidhuber, 1997; Graves, 2012, LSTM-CRF).

CRF: is a popular algorithms for sequence tagging tasks (Lafferty et al., 2001). The objective is to find the label sequence $\mathbf{y} = (y_1, ..., y_M)$ that maximizes

$$p(\mathbf{y}|\mathbf{x}) = \frac{1}{Z_{\boldsymbol{\lambda}}(\mathbf{x})} \exp\{\boldsymbol{\lambda} \cdot \boldsymbol{f}(\mathbf{y}, \mathbf{x})\},$$

where $Z_{\boldsymbol{\lambda}}(\mathbf{x})$ is the normalization factor, $\boldsymbol{\lambda}$ is the weight vector and $\boldsymbol{f}(\mathbf{y}, \mathbf{x})$ is the extracted feature vector for the observed sequence $\mathbf{x}$.

SEARN is a powerful structured prediction algorithm, which formulates the sequence labeling problem as a search process. The objective is to find the label sequence $\boldsymbol{y} = (y_1, ..., y_M)$ that maximizes

$$p(\mathbf{y}|\mathbf{x}) \propto \sum_{m=1}^{M} \mathbb{I}_{[\boldsymbol{C}(\mathbf{x}, y_1, ..., y_{m-1}) = \hat{y}_m]},$$

where $\boldsymbol{C}(\bullet)$ is a cost sensitive multiclass classifier and $\hat{\mathbf{y}}$ are the ground-truth labels.

STRUCTPERCEPTRON is an extension of the standard perceptron. In our setting we model a segment-based search algorithm, where each unit is a segment of $\mathbf{x}$ (e.g., $\langle b_i, e_i \rangle$), rather than a single word (e.g., x_i). The objective is to find the label sequence $\boldsymbol{y} = (y_1, ..., y_M)$ that maximizes

$$p(\mathbf{y}|\mathbf{x}) \propto \mathbf{w}^{\top} \cdot \boldsymbol{f}(\mathbf{x}, \mathbf{y}),$$

where $\boldsymbol{f}(\mathbf{x}, \mathbf{y})$ represents the feature vector for instance $\mathbf{x}$ along with the configuration $\mathbf{y}$ and $\mathbf{w}$ is updated as $\mathbf{w} \leftarrow \mathbf{w} + \boldsymbol{f}(\mathbf{x}, \hat{\mathbf{y}}) - \boldsymbol{f}(\mathbf{x}, \mathbf{y})$.

LSTM-CRF The above algorithms heavily rely on manually-crafted features to perform sequence tagging. We decided to alleviate that by using long short-term memory networks with a CRF layer. Our model is similar to R-CRF (Mesnil et al., 2015), but for the hidden recurrent layer we use LSTM (Hochreiter and Schmidhuber, 1997; Graves, 2012). We denote with h_i the hidden vector produced by the LSTM cell at i-th token. Then the conditional probability of $\mathbf{y}$ given a query $\mathbf{x}$ becomes:

$$p(\mathbf{y}|\mathbf{x}) = \frac{1}{Z(h)} \exp\{\sum_i (W^h_{y_i} h_i + W^t_{y_i, y_{i-1}})\},$$

where $W^h_{y_i}$ is the weight vector corresponding to label y_i, and $W^t_{y_i, y_{i-1}}$ is the transition score corresponding to y_i and y_{i-1}. During training, the values of W^h, W^t, the LSTM layer and the input word embeddings are updated through the standard back-propagation with AdaGrad algorithm. We also concatenate pre-trained word embedding and randomly initialized embedding (50-d) for the knowledge-base types of each token and use this information in the input layer. In our experiments, we set the learning rate to 0.05 and take each query as a mini-batch and run 5 epochs to finish training.

[1] taku910.github.io/crfpp/

[2] github.com/JohnLangford/vowpal_wabbit

Features	CRF			SEARN			STRUCTPERCEPTRON		
	P (%)	R (%)	F$_1$	P (%)	R (%)	F$_1$	P (%)	R (%)	F$_1$
POS	39.86	35.51	37.56	34.97	33.55	34.25	33.03	24.70	28.27
KB	51.64	41.08	45.76	41.96	37.26	39.47	35.70	35.97	35.84
WE	65.31	61.02	63.11	67.58	67.00	67.29	71.29	68.12	69.67
LEX+ORTO+PSTNL + POS + KB	86.49	83.84	**85.15**	84.19	84.30	**84.24**	88.88	86.87	**87.87**
LEX+ORTO+PSTNL + POS + WE	88.30	85.74	**87.00**	84.32	84.15	**84.24**	87.85	85.69	**86.76**
LEX+ORTO+PSTNL + POS + KB + WE	88.86	86.29	87.55	84.30	84.50	84.40	89.18	87.10	**88.13**

Table 1: Results from feature study.

2.3 Features

Lexical (LEX): are widely used N-gram features. We use unigrams of the current w_0, previous w_{-1} and next w_{+1} words, and bigrams $w_{-1}w_0$ and w_0w_{+1}.

Orthographic (ORTO): are binary mutually non-exclusive features that check if w_0, w_{-1} and w_{+1} contain *all-digits*, *any-digit*, *start-with-digit-end-in-letter* and *start-with-letter-end-in-digit*. They are designed to capture model names like *hero3* and *m560*.

Positional (PSTNL): are discrete features modeling the position of the words in the query. They capture the way people tend to write products and brands in the query.

Part-of-Speech (POS): capture nouns and proper names to better recognize *products* and *brands*. We use Stanford tagger (Toutanova et al., 2003).

Knowledgebase (KB): are powerful semantic features (Tjong Kim Sang, 2002; Carreras et al., 2002; Passos et al., 2014). We automatically collected and manually validated $200K$ brands, products, models and product families items extracted from Macy's and Amazon websites.

WordEmbeddings (WE): While external knowledge bases are great resource, they are expensive to create and time-consuming to maintain. We use word embeddings (Mikolov et al., 2013) [3] as a cheap low-maintenance alternative for knowledge base construction. We train the embeddings over 2.5M unlabeled shopping queries. For each token in the query, we use as features the 200 dimensional embeddings of the top 5 most similar terms returned by cosine similarity.

3 Experiments and Results

Data Set To the best of our knowledge, there is no publicly available shopping query data annotated with *product*, *brand*, *model*, *product family* and *other* categories. To conduct our experiments, we collect 2.5M shopping queries through click logs (Hua et al., 2013). We randomly sampled $37,000$ unique queries from the head, torso and tail of a commercial web search engine and asked two independent annotators to tag the data. We measured the Kappa agreement of the editors and found .92 agreement, which is sufficient to warrant the goodness of the annotations.

We randomly split the data into 80% for training and 20% for testing. Table 2 shows the distribution of the entity categories in the data.

	Product	Brand	Model	Prod. Family	⊥
Train	21,688	10,417	4,394	6,697	47,517
Test	5,413	2,659	1,099	1,716	11,780

Table 2: Entity category distribution.

We tune all parameters on the training set using 5-fold cross validation and report performance on the test set. All results are calculated with the CoNLL evaluation script[4].

Performance w.r.t. Features Table 1 shows the performance of the different models and feature combinations. We use the individual features as a baseline. The obtained results show that these are insufficient to solve such a complex task. We compared the performance of the KB and WE features when combined with (LEX+ORTO+PSTNL) information. As we can see, both KB and WE reach comparable performance. This study shows that training embeddings on large in-domain data of shopping queries is a reliable and cheap source for knowledge base construction, when such information is not present. In our study the best performance is reached when all features are combined. Among all machine learning classifiers for which we manually designed features, structured perception reaches the best performance of 88.13 F$_1$ score.

In addition to the feature combination and model comparison, we also study in Figure 1 the training time of each model in log scale against its F$_1$ score. SEARN is the fastest algorithm to train,

[3] https://code.google.com/p/word2vec/

[4] `cnts.ua.ac.be/conll2000/chunking/conlleval.txt`

Category	CRF			SEARN			STRUCTPERCEPTRON			LSTM-CRF		
	P (%)	R (%)	F_1	P (%)	R (%)	F_1	P (%)	R (%)	F_1	P (%)	R (%)	F_1
brand	91.79	87.93	89.82	89.3	89.3	89.3	93.99	91.20	92.57	95.15	92.29	93.70
model	86.28	80.71	83.40	80.7	78.9	79.8	85.56	80.89	83.16	87.25	85.90	86.57
product	87.85	88.16	88.00	83.4	85.0	84.2	87.90	87.92	87.91	91.94	90.98	91.46
product family	89.27	81.41	85.16	81.4	79.0	80.2	88.12	82.17	85.04	87.98	87.47	87.73
Overall	88.86	86.29	87.55	84.3	84.5	84.4	89.18	87.10	88.13	**91.61**	**90.24**	**90.92**

Table 3: Per category performance.

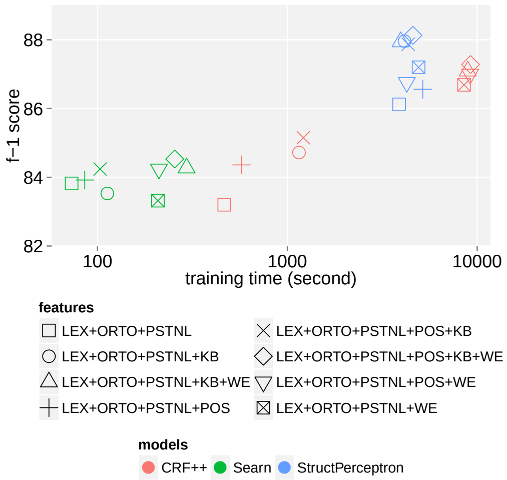

Figure 1: Training time vs F_1 performance.

while CRF takes the longest time to train. Among all STRUCTPERCEPTRON offers the best balance between efficiency and performance in a real time setting.

Performance w.r.t. Entity Category Table 3 shows the performance of the algorithms with the manually designed features against the automatically induced ones with LSTM-CRF. We show the performance of each individual product entity category. Compared to all models and settings, LSTM-CRF reaches the best performance of 90.92 F_1 score. The most challenging entity types are *product family* and *model*, due to their "wild" and irregular nature.

Performance w.r.t. Query Length Finally, we also study the performance of our approach with respect to the different query length. Figure 2 shows the F_1 score of the two best performing algorithms LSTM-CRF and STRUCTPERCEPTRON against the different query length in the test set. Around 83% of the queries have length between 2 to 5 words, the rest are either very short or very long ones. As it can be seen in Figure 2, independent of the query length, our models reach the same performance for short and long queries. This shows that the models are robust and agnostic to the query length.

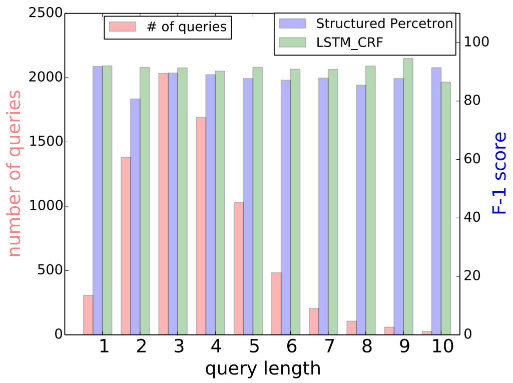

Figure 2: F_1 performance with varying query length.

4 Conclusions and Future Work

In this work, we have defined the task of product entity recognition in shopping queries. We have studied the performance of multiple structured prediction algorithms to automatically recognize *product*, *brand*, *model* and *product family* entities. Our comprehensive experimental study and analysis showed that combining lexical, positional, orthographic, POS, knowledge base and word embedding features leads to the best performance. We showed that word embeddings trained on large amount of unlabeled queries could substitute knowledge bases when they are missing for specialized domains. Among all manually designed feature classifiers STRUCTPERCEPTRON reached the best performance. While among all algorithms LSTM-CRF achieved the highest performance of 90.92 F1 score. Our analysis showed that our models reach robust performance independent of the query length. In the future we plan to tackle attribute identification to better understand queries like "*diamond shape emerald ring*", where *diamond shape* is a cut and *emerald* is a gemstone type. Such fine-grained information could further enrich online shopping experience.

References

Xavier Carreras, Lluís Màrques, and Lluís Padró. 2002. Named entity extraction using adaboost. In *Proceedings of CoNLL-2002*, pages 167–170.

Michael Collins. 2002. Discriminative training methods for hidden markov models: Theory and experiments with perceptron algorithms. In *Proc. EMNLP*, pages 1–8.

Hal Daumé III, John Langford, and Daniel Marcu. 2009. Search-based structured prediction. *Machine Learning*, 75(3):297–325.

Gian Fulgoni. 2014. State of the US retail economy in q1 2014. In *Comscore, Technical Report*.

Alex Graves. 2012. *Supervised Sequence Labelling with Recurrent Neural Networks*, volume 385 of *Studies in Computational Intelligence*. Springer.

Marti Hearst. 1992. Automatic acquisition of hyponyms from large text corpora. In *Proc. of the 14th conference on Computational linguistics*, pages 539–545.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Computation*, 9(8):1735–1780.

Xian-Sheng Hua, Linjun Yang, Jingdong Wang, Jing Wang, Ming Ye, Kuansan Wang, Yong Rui, and Jin Li. 2013. Clickage: Towards bridging semantic and intent gaps via mining click logs of search engines. In *Proceedings of the 21st ACM International Conference on Multimedia*, MM '13, pages 243–252.

Anitha Kannan, Inmar E. Givoni, Rakesh Agrawal, and Ariel Fuxman. 2011. Matching unstructured product offers to structured product specifications. In *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '11, pages 404–412.

Zornitsa Kozareva and Eduard Hovy. 2010. Learning arguments and supertypes of semantic relations using recursive patterns. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 1482–1491.

Zornitsa Kozareva, Ellen Riloff, and Eduard Hovy. 2008. Semantic class learning from the web with hyponym pattern linkage graphs. In *Proceedings of ACL-08: HLT*, pages 1048–1056.

Zornitsa Kozareva. 2015. Everyone likes shopping! multi-class product categorization for e-commerce. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1329–1333.

John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *icml*, pages 282–289.

Xiao Li. 2010. Understanding the semantic structure of noun phrase queries. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 1337–1345.

Mehdi Manshadi and Xiao Li. 2009. Semantic tagging of web search queries. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 2 - Volume 2*, pages 861–869.

Grégoire Mesnil, Yann Dauphin, Kaisheng Yao, Yoshua Bengio, Li Deng, Dilek Z. Hakkani-Tür, Xiaodong He, Larry P. Heck, Gökhan Tür, Dong Yu, and Geoffrey Zweig. 2015. Using recurrent neural networks for slot filling in spoken language understanding. *IEEE/ACM Transactions on Audio, Speech & Language Processing*, 23(3):530–539.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Distributed representations of words and phrases and their compositionality. volume abs/1310.4546, pages 3111–3119.

Marius Paşca and Benjamin Van Durme. 2008. Weakly-supervised acquisition of open-domain classes and class attributes from web documents and query logs. In *Proceedings of ACL-08: HLT*, pages 19–27.

Alexandre Passos, Vineet Kumar, and Andrew McCallum. 2014. Lexicon infused phrase embeddings for named entity resolution. *CoRR*, abs/1404.5367.

Duangmanee (Pew) Putthividhya and Junling Hu. 2011. Bootstrapped named entity recognition for product attribute extraction. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 1557–1567.

Erik F. Tjong Kim Sang. 2002. Introduction to the conll-2002 shared task: Language-independent named entity recognition. In *Proceedings of CoNLL-2002*, pages 155–158.

Kristina Toutanova, Dan Klein, Christopher D Manning, and Yoram Singer. 2003. Feature-rich part-of-speech tagging with a cyclic dependency network. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology - Volume 1*, pages 173–180.