Title-Based Product Search – Exemplified in a Chinese E-commerce Portal

Chien-Wen Chen and Pu-Jen Cheng

Department of Computer Science and Information Engineering, College of Electrical Engineering and Computer Science, National Taiwan University
{r97019,pjcheng}@csie.ntu.edu.tw

Abstract. There are too many products in an on-line shopping website. We need to help buyers to find products they want in an efficient way. A keyword-based IR system seems suitable for searching products. Unfortunately, we observe from real world query logs and find that queries for product search are usually very short. What is worse, a document described a product may have lots of words of related products. It is hard for an IR system to distinguish representative terms from other noisy terms. Hence, we propose a supervised learning method to realize semantic types of each term in product document titles. Then we modify Language Model to improve the relevance of search results. Our methods have significant improvement in search result precision in real world document collection and query collections.

Keywords: Information Retrieval, Language Specific IR, Domain-specific IR, product search, E-commerce.

1 Introduction

There are too many products in an on-line shopping website. Take eBay for example, their platform sells more than 75 million kinds of products at any time [8]. We need to help buyers to find products they want in an efficient way. The two common methods are searching and browsing by categories. A keyword-based IR system seems suitable for searching products. Figure 1 shows a snapshot example of shopping.com[1] issuing a query "iphone" on June 2010.

An efficient retrieval system that can provide the most relevant ranking list of products to meet buyer's query is very important. If the retrieval system cannot provide suitable ranking list corresponding to buyers' query in an efficient way, buyers may change to other shopping platforms which therefore declines shopping platform provider's profit. Unfortunately, we observe from real world query logs and find that queries for product search are usually very short. A query for product search consists of only 1.28 English words or 3.22 Chinese characters in average. So, it is hard to realize buyer's intention. What is worse, a document describing a product may contain lots of words associating with related products. It is hard for an IR system to distinguish representative terms from other noisy terms. The retrieved products often match the keywords from query issued by buyers, but have poor relevance.

[1] http://www.shopping.com/

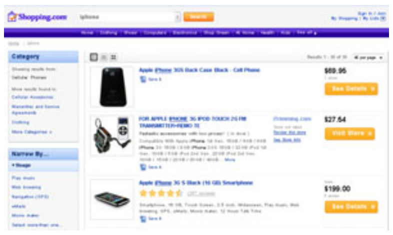

Shopping.com
iphone
Category
Cellular Phones
Narrow By...
Apple iPhone 3GS Back Case Black - Cell Phone
$89.95
See Details
FOR APPLE IPHONE 3G IPOD TOUCH 2G FM TRANSMITTER+REMO TE
$27.54
Visit Store
Apple iPhone 3G S Black (16 GB) Smartphone
$199.00
See Details

Fig. 1. A snapshot of an on-line shopping platform. (Shopping.com).

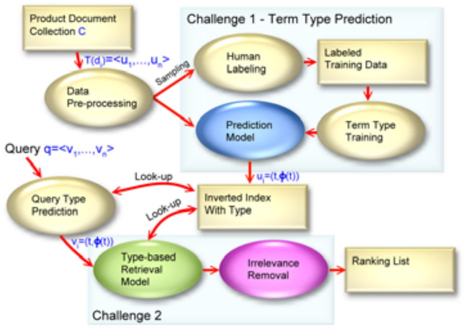


Fig. 2. System architecture

For example, when a buyer issued a query "iphone", we assume that the buyer was interested in the iphone cellphones, instead of iphone accessories such as iphone compatible batteries, leathers, or screen protectors. Unfortunately, from Fig. 1, we can see that the top two retrieved results when issuing query "iphone" are iphone accessories: black case and transmitter, instead of iphone cellphones. Accordingly, the IR system for product search needs more sophisticated methods in the hope of understanding buyers' and sellers' intentions.

We make an assumption that a buyer specifies keywords to describe products that he or she wants to buy. The buyer is interested in main products, instead of accessories. Ravi Chandra et al. also make similar assumption [3]. In this work, we observe that each term in a product document title has different semantic types in different circumstances. Hence, we propose an approach which considers not only the string of the term but also the semantic type of the term in each product document title. We apply a supervised learning method to labeling the semantic type for each term in product document title and modify the conventional Language Model to improve the relevance of search results.

The rest of this paper is organized as the following. Section 2 introduces many related works and existing approaches. Section 3 formulates this problem and defines notations in this paper. In section 4, our methods are described in detail. And in section 5, we introduce our dataset and report experiment results to verify our methods. Finally, section 6 is our conclusions and future works.

2 Related Work

Some existing works are based on logs, which are knowledge from previous buyers. On-line shopping platforms record users' browsing history, keywords for search, clicking behavior on products, or transaction records to purchase products. Methods using the history data are known as log-based solutions. Nish Parikh and Neel Sundaresan make a detailed study of eBay logs. They try to find semantic query relations and build graphs for searching or recommendation applications [7]. Also, they try to find burst events from queries [8]. Ravi Chandra et al. directly explore rules from logs for product search [3]. An association-rule-mining-liked method is proposed by them and has good performance.

Logs can also be used by collaborative filtering or recommendation systems. Using collaborative filtering techniques in e-commerce platforms has been discussed for years [9,2,10]. Raz Lin et al. integrate both information retrieval and collaborative filtering techniques for search in on-line shopping platforms. Their solution considers not only buyers' but also their neighbors' browsing history and preference profiles [5,6]. Another research group, Y.S. Kim et al., considers navigational and behavior patterns as well [4]. They also give more considerations to the products that are clicked but not purchased.

Log-based solutions may have good performance, but they also have many disadvantages. First of all, they cannot response to the market immediately. Time and adequate users are required to shape up rules from the logs. Second, log data is usually difficult to obtain due to privacy issues. Most of large scale websites have their own privacy policies. Last, using log data to change searching behavior will influence future log data. Log-based methods can only be applied once in some circumstance. For these reasons, we would like to solve this problem using only contents.

3 Problem Specification

To formalize this problem, we define a term in the first place. A **term** t can be a separate word in English, a single symbol such as punctuation marks, or characters in Chinese. Each term t has its own part of speech, denoted as $POS(t)$. We regard a product as a document with one or more terms. A **product document** d_i representing a product to be sold contains two disjoint parts: title $T(d_i)$ and description $D(d_i)$. A **title** consists of m tuples: $T(d_i) =< u_1, u_2, ..., u_m >$. A **description** consists of a set of terms: $D(d_i) = \{t_j\}$. A tuple consists of two parts: a term and a semantic type of the term. A **tuple** is defined as $u_j =< t_j, \phi(t_j) >$, where $\phi(t_j)$ is a notation representing a semantic type for term t_j. We will describe how to determine the semantic type for a term in our methodologies later. Also, a **product query** q consists of a set of tuples: $\{v_1, v_2, ..., v_n\}$.

The problem to be focused in this paper can be formulated. Given a product document collection $C = \{d_1, d_2, ..., d_k\}$ with k product documents and a query q, our objective is to rank all product documents $d_i \in C$ ordered by a scoring function $Score(q, T(d_i))$ in a decreasing order.

4 Methodology

Figure 2 is our system architecture. As defined above, given a data collection C, we have to pre-process the product documents, including storage, indexing, and Chinese word segmentation. Our methods have three main phases to solve the problem. The first phase of our method is named "term type prediction". In this phase, we aim to predict the semantic type of each term from product document titles. Using this type information, the second phase is a type-based retrieval model which retrieves relevant product documents from product document collection. And the final phase is a filter to remove irrelevant product documents in the final ranking list. The three main phases will describe in detail in the following subsections.

4.1 Term Type Prediction

Term Type Prediction in Product Document Title. In our observation for mismatched product documents, the crux of the problem we find is that the retrieval system does not understand the semantic type of each term in the product document. For instance, consider these two product document titles:

– USB Desktop Battery Charger Cradle For Nokia N97 Mini **Cellphone**
– Nokia N97 Mini Unlocked **Cellphone** Plus Free Battery

When a buyer issues a query "cellphone" and intents to buy a cellphone, these two product documents may be retrieved because of matched keywords. However, we humans can easily distinguish that the second product document is the real cellphone, but the first product document is only an accessory of cellphones. Even though both product document titles contain the same term "cellphone", the semantic type of this term are different. In the second product document, the term "cellphone" is a product, but in the first product document is not. The main product of the first product document is battery, whereas the main product of the second product document is cellphone.

Hence, our idea is to let the retrieval system to understand the semantic type of each term for each product document title. By observation from query collection in our experiments, high frequency query terms have three main kinds of types: product, brand, and model. To simplify, we aim to categorize each term into one of the four types:

– Product (P): the major sale goods.
– Brand (B): production companies or retailers.
– Model (M): the design model, usually appears in electric products. E.g. "N97", "nano", "shuffle", "X61", or "hero".
– None (N): none of above.

The objective categorizing results for the above examples are:

– USB^N Desktop^N Battery^P Charger^P Cradle^P For^N Nokia^B N98^M Mini^M Cellphone^N
– Nokia^B N97^M Mini^M Unlocked^N Cellphone^P Plus^N Free^N Battery^N

We apply a well-known supervised machine learning approach named support vector machine (SVM). We train three binary classifiers to determine each term in title: "Is a Product or not", "Is a Brand or not", and "Is a Model or not". We use about 50 features for each term in product document title. For a product document title $T(d_i) = \{u_1, u_2, ...u_m\}$, a term $t_j$in the tuple $u_j =< t_j, \phi(t_j) >$ has the following features:

– Part of speech features. (t_{j-1}, t_j, t_{j+1}) We observe that a term with Product type tends to be a noun term ($Pos(t_j) = Noun$), which follows an adjective term ($Pos(t_{j-1}) =$ *Adj.*), or follows a non-word term ($Pos(t_{j-1}) = NotAWord$). Brand or Model type terms are usually not written in Chinese.

- Is parentheses (t_{j-1}, t_j, t_{j+1}). Chinese sellers like to put Brand terms into parentheses. E.g. "()", "[]", "{}" or "⟨⟩".
- Is color (t_{j-1}, t_j, t_{j+1}). Color terms are usually neither Brand, Model, nor Product. E.g. "red", "gold", or "two color".
- Length of term (t_{j-1}, t_j, t_{j+1}). In multi-byte encoded languages like Chinese, this value is the number of characters, instead of number of bytes.
- Is alphabets and/or digits only (t_{j-1}, t_j, t_{j+1}). Brand or Model type terms usually consist of only alphabets or digits.
- Term frequency features (t_j). We find that Brand terms sometimes occur twice in a product document title.
- Document frequency features (t_j). A term with very high document frequency may be a useless stopword. A term with very low document frequency may be a Model term since Model terms are usually very specific.
- Location-based features (t_j). Term occurrence position is very useful. Brand terms are usually placed in the first term and Product terms are usually placed in the last term because of Chinese grammars. But there are still many exceptions.

Term Type Prediction in Query. After giving a type for each term in each product document title, we also have to give a type for each term in the query. For a query $q = \{v_1, v_2, ...v_n\}$ with n terms, we predict each type $\phi(t_j)$ for each term t_j in tuple v_j as the most likely semantic type according to the whole corpus. This method is good for some terms that have ambiguous semantic types. For example, the term "Apple" is not only a famous computer company but also a kind of fruit. But in the whole corpus, the term "Apple" is more likely to be a brand of Apple Computer Inc., instead of a fruit product. For this reason, we predict $\phi(t_j) = U$ if $P(\phi(t_j) = U|T(C)) > 50\%$ where $U \in \{Product, Brand, Model\}$. Otherwise, $\phi(t_j) = None$.

4.2 Retrieval Model

To find relevant product documents, we modify Language Model with Bayesian smoothing using Dirichlet Priors. Given a query q, we have to estimate the probability of generating q from each product document title $T(d)$. The probability $P(q|T(d))$ is the final scoring function $Score(q, T(d))$ for retrieval ranking. With Bayesian smoothing using Dirichlet Priors, we assume pseudo counts $\mu_1 P(t_i|T(C))$ for unseen terms in product document title $T(d)$.

$$P(q|T(d)) \approx \prod_{1}^{n} \left(\frac{|T(d)|}{|T(d)| + \mu_1} P(t_i|T(d)) + \frac{\mu_1}{|T(d)| + \mu_1} P(t_i|T(C)) \right) \quad (1)$$

For a query term t_q and another term t_d from product document title, we define two matching conditions: 1) Exactly matching: $t_q = t_d$ and $\phi(t_q) = \phi(t_d)$. 2) Partial matching: $t_q = t_d$. Note that the two conditions are not disjoint, a term satisfies exactly matching also satisfies partial matching. We give some pseudo count $\mu_2 P(t|T(d))$ for terms satisfying partial matching for smoothing propose, instead of restricting the exactly matching for all terms. Hence, we have:

$$P(t|T(e)) = \frac{|T(d)|}{|T(d)| + \mu_2} P_{ext}(t = u, \phi(t) = \phi(u)|e, u \in T(e)) \tag{2a}$$

$$+ \frac{\mu_2}{|T(d)| + \mu_2} P_{par}(t = u|e, u \in T(e)) \tag{2b}$$

where t is a term and e can be a product document d or the whole product document collection C. To observe the effect of this approach, take the logarithm of equation 1 and place equation 2 into it, we have:

$$logP(q|T(d)) \approx \tag{3a}$$

$$\sum_{i=1, t_i \in T(d)}^{n} log \left(\frac{|T(d)|^2 P_{ext}(t_i|T(d)) + \mu_2 |T(d)| P_{par}(t_i|T(d))}{\mu_1 |T(d)| P_{ext}(t_i|T(C)) + \mu_1 \mu_2 P_{par}(t_i|T(C))} + 1 \right) \tag{3b}$$

$$+ n \frac{\mu_1}{|T(d)| + \mu_1} \tag{3c}$$

Part (3b) is like the term frequency - inverse document frequency (TF-IDF) weighting and part (3c) can be regard as document length normalization. We can also find that a product document title with a term exactly matching another term in query will have more likelihood to be retrieved than only partial matching.

4.3 Irrelevant Removal

In the previous phases, many smoothing strategies are adopted to improve the search recall. But this also causes noises. To have a better ranking precision, we try a naive method to eliminate irrelevant product documents from the retrieved set. We make an assumption that most of top retrieved product documents are relevant. Hence, this phase aims to eliminate product documents that are not similar to the top retrieved product documents. The elimination method simply calculates an average vector from top ten retrieved product document titles, and removes product documents with cosine similarity to the average vector less than or equal to 0.5.

5 Experiments

In the experiments of term type prediction, we will show our prediction result. In the other phases, the type-based retrieval model and the irrelevant removal, we will evaluate our methods by common information retrieval evaluation criteria.

5.1 Data Collection

Our experimental datasets have two parts: the product document collection and the query collection.

Product Document Collection. The product document collection contained 984,096 product documents and was crawled from Yahoo! Taiwan Shopping platform on October, 2009. Yahoo! Taiwan Shopping is a famous shopping website in Taiwan. Each product document has about 10 columns including id, category, title, subtitle, thumbnail image, description and timestamps. In our observation, the average number of terms in all product document titles is 11.4703 and the average unique number of terms in all product document titles is 11.2823.

Table 1. Query collection

Data Source	Y! TW Shopping	Ruten Auction
Number of queries	29,537	542,824
Number of unique queries	11,314	74,765
Average query length (Chinese characters)	3.22	3.00
Average query length (English words)	1.28	1.12

Query Collection. The query collection was a set of query keywords issued to Yahoo! Taiwan Shopping and Ruten Auction Platform[2] on April 25, 26 and 27 in 2010. Ruten Auction Platform is another shopping website in Taiwan funded by PChome Online and eBay Inc. Different from Yahoo! Taiwan Shopping, products in Ruten Auction Platform were decided by various sellers like eBay, yet products in Yahoo! Taiwan Shopping were decided by inner salesmen. The query collection was collected from the logs of Web Reputation Service provided by Trend Micro Inc. A simple observation is shown in Table 1.

5.2 Term Type Prediction Evaluation

To produce training data for supervised learning, we have to sample some product to be labeled by human. We reference to the categories of the product document collection. The three main categories of products in amount are clothing accessories (17%), clothes(17%), and 3C (Computer, Communication, and Consumer Electronic) (13%). Therefore, we sample 400 product documents for each category from the three main categories. The total 1,200 product document titles are labeled by humans. We have three volunteers. A volunteer has to label one of the four types for each term in each product document title. We choose the labels with at least two agreements on type from the volunteers.

We use LibSVM [1] for SVM training. To find the best parameters γ and C for the C-Support Vector Classification and the Gaussian radial basis function, we try different parameters using a grid tool to reach the best accuracy. Using the parameters for the best accuracy, the performances including accuracy, recall, precision and F-score are shown in the left side in Table 2 (Micro View). In this classification problem, recall means the percentage of correct prediction from all true positive examples. Precision means the percentage of correct prediction from all examples which are predicted as positive. The F-score is a measurement considering both recall and precision. Note that F-score does not consider true negative examples, which are very common in this training method. This report is a micro view since we evaluate in term level. We find out that type Brand is the easiest one to predict, yet type Product and Model are good for precision but poor in terms of recall.

To avoid the problem of Chinese word segmentation, we also evaluate in a macro view. We concatenate terms by human and restrict the correct prediction to only exactly matching the concatenated section. For example, we can consider a title with four tuples

[2] http://www.ruten.com.tw/

Table 2. Term type prediction performance

	Micro View				Macro View		
Type	Accuracy	Recall	Precision	F-score	Recall	Precision	F-score
Product	92.80%	59.09%	79.23%	67.69%	42.83%	69.80%	53.08%
Brand	96.20%	89.67%	83.76%	86.61%	87.58%	82.78%	85.11%
Model	96.43%	62.11%	76.82%	68.69%	43.48%	67.77%	52.97%

$T(d_i) =< u_1, u_2, u_3, u_4 >$. If the ground truths of terms with Product type are u_2 and u_3, a prediction $\phi(u_1) = N, \phi(u_2) = P, \phi(u_3) = P, \phi(u_4) = N$ is correct and otherwise conditions like $\phi(u_1) = P, \phi(u_2) = P, \phi(u_3) = P, \phi(u_4) = N$ or $\phi(u_1) = N, \phi(u_2) = P, \phi(u_3) = N, \phi(u_4) = N$ are incorrect. The performances including recall, precision and F-score are available in the right side in Table 2 (Macro View). We do not report the accuracy because it is hard to define true negative conditions in macro view. In this report, the recall of Product type and Model type is very low, similar to the micro view performance.

Table 3. Feature contribution analysis

Type		NR	PAll	PN	PV	PAdv	PAdj	PNC	PNW	PTS	CLR	LEN	AD	TF	DF	LOC
Product	Accuracy	-00.00%	-01.20%	-00.10%	-00.22%	+00.03%	+00.10%	-00.00%	-00.47%	+00.01%	-00.20%	-00.95%	-00.04%	+00.04%	-00.20%	**-04.68%**
	Recall	-00.00%	+07.48%	+02.03%	+01.57%	+00.45%	+01.13%	-00.00%	**-01.13%**	+00.34%	+00.06%	+03.82%	+00.06%	+00.62%	+01.29%	+14.24%
	Precision	-00.00%	-11.96%	-02.32%	-02.79%	-00.11%	-00.06%	-00.00%	-02.97%	-00.14%	-01.72%	-09.11%	-00.39%	-00.13%	-02.58%	**-26.73%**
	F-score	-00.00%	-00.77%	+00.42%	-00.05%	+00.25%	+00.71%	-00.00%	-01.83%	+00.17%	-00.60%	-01.37%	-00.11%	+00.35%	-00.14%	**-06.50%**
Brand	Accuracy	-00.00%	-00.57%	-00.20%	-00.09%	-00.08%	-00.04%	-00.06%	-00.25%	-00.38%	-00.12%	-00.41%	-00.15%	-00.06%	-00.25%	**-01.64%**
	Recall	-00.00%	+02.73%	+00.32%	+00.37%	-00.00%	-00.10%	-00.16%	-00.73%	-00.31%	-00.42%	-00.16%	-00.10%	+00.11%	+00.26%	**-02.04%**
	Precision	-00.00%	-04.56%	-01.33%	-00.76%	-00.43%	-00.12%	-00.23%	-00.96%	-01.94%	-00.44%	-02.19%	-00.76%	-00.38%	-01.56%	**-07.51%**
	F-score	-00.00%	-01.32%	-00.57%	-00.24%	-00.23%	-00.11%	-00.20%	-00.85%	-01.19%	-00.43%	-01.26%	-00.46%	-00.16%	-00.72%	**-05.07%**
Model	Accuracy	-00.00%	-00.28%	-00.09%	-00.01%	+00.03%	+00.03%	-00.00%	-00.12%	-00.07%	-00.01%	**-00.51%**	-00.14%	+00.01%	-00.06%	-00.20%
	Recall	-00.00%	+05.69%	+00.57%	+00.23%	+00.68%	+00.57%	-00.00%	**-00.46%**	+00.79%	+00.91%	+02.16%	+02.05%	+00.68%	+02.50%	+12.97%
	Precision	-00.00%	-06.62%	-01.57%	-00.27%	-00.02%	+00.06%	-00.00%	-01.47%	-01.55%	-00.71%	-07.75%	-03.20%	-00.28%	-02.33%	**-08.54%**
	F-score	-00.00%	+00.29%	-00.30%	+00.03%	+00.41%	+00.37%	-00.00%	**-00.87%**	-00.15%	+00.26%	-02.10%	-00.12%	+00.30%	+00.51%	+02.83%

NR : Reserve all features. PAll : Remove all part of speech features. PN : Remove feature "Is noun". PV : Remove feature "Is verb".
PAdv : Remove feature "Is adverb". PAdj : Remove feature "Is adjective". PNC : Remove feature "Is not Chinese". PNW : Remove feature "Is not a word".
PTS : Remove all parentheses related features. CLR : Remove feature "Is color". LEN : Remove term length feature.
AD : Remove all alphabets or digits related features. TF : Remove all term frequency related features.
DF : Remove all document frequency related features. LOC : Remove all location-based features.

Feature Contribution Analysis. We are curious about which feature is useful during training. Accordingly, we decide to remove some features and check whether the performance is affected or not.

The percentage of change in accuracy, recall, precision, and F-score are shown in Table 3. Negative value indicates that the performance has decreased after the removal of feature or group of features and vice versa. Values that contribute the highest performance in a row are marked in bold. In this result, we can find that location-based features (LOC) have significant contributions when predicting Product or Brand. A possible reason is that sellers tend to put Product or Brand terms in the first place or the last place in a product document title. Another discovery is that term length (LEN) or "Is not a word" (PNW) features contribute a lot when predicting Model because sellers usually use one character symbols to decorate model strings.

Category of Products Prediction Performance Analysis. Finally, we observe the prediction performance in each category of products. Figure 3 shows the three main

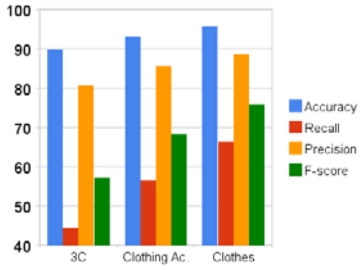


Fig. 3. Product type prediction performance in three main categories

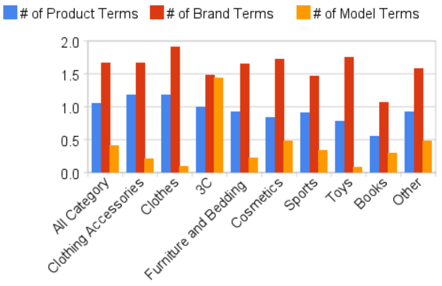


Fig. 4. Number of terms in prediction result

categories labeled by human in advance. Clothes products are easier to predict than 3C products in terms of accuracy. We speculate that 3C products have the poorest recall due to the diversity of 3C products. 3C products contain various products and their accessories. Like the cellphone examples in section 4.1, a cellphone can have lots of possible related accessories terms together in title. So, it is hard to distinguish Product terms in 3C product titles. After applying the prediction model to all categories of products, the average number of terms in the prediction result is reported in Fig. 4. Clothes products tend to have more Brand type terms because a Chinese brand is often split into many separated terms. 3C products have more Model terms, since models usually appear in 3C products. For example, “K800i”, “3310”, “X61”, or “U5F” are models for cellphones and laptops.

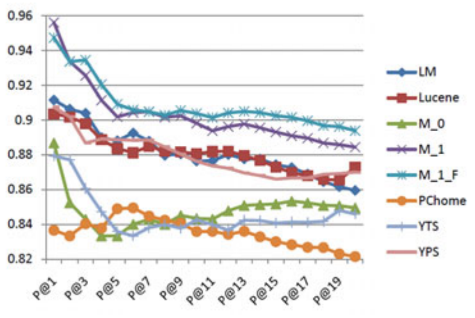
5.3 Retrieval Model Evaluation

We split the query collection into two parts: 1) High frequency queries (HFQ). The most frequent queries to be issued. 2) Random queries (RQ). Randomly sampled queries from the original list of queries. We sample 30 queries from each part for evaluation.

We compare our methods with different parameter μ_2 and some existing methods or implementations, including Vector Space Model (Apache Lucene[3]), conventional Language Model, and search results from on-line product search services for well-known Chinese shopping platforms in Taiwan. We compare with on-line product search services including Yahoo! Taiwan Shopping (YTS), Yahoo! Taiwan Portal Product Search (YPS)[4], and PChome Shopping[5] on May 2010. It is important to note that the on-line product search services have different product corpora from our methods currently used. Hence, the comparisons of on-line product search services are for reference only.

Figure 5 is an average evaluation result of HFQ and RQ dataset from query collection Ruten Auction and Yahoo Taiwan Shopping because each separated result is almost the same. LM means conventional Language Model. M_0, M_1 and M_1_F are our methods

[3] http://lucene.apache.org/
[4] http://tw.info.search.yahoo.com/products
[5] http://shopping.pchome.com.tw/



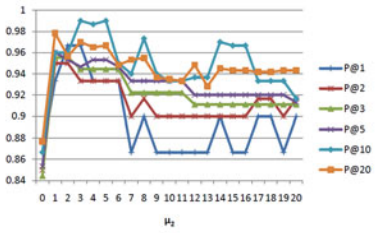


Fig. 5. Precision of Retrieval Results. (HFQ and RQ).

Fig. 6. Precision of different parameter values

with parameter $\mu_2 = 0$, $\mu_2 = 1$, and $\mu_2 = 1$ with irrelevant removal. Our methods M_1 and M_1_F are significantly different from baseline methods LM and Lucene from Student's t-test with $p-value < 0.01$ in precision at one. Compared only with baseline Lucene, M_1 and M_1_F also have significant different with $p - value < 0.05$ when measuring P@1, P@2, P@3, P@5, P@10 and P@20. In this Fig., we can see that our method with $\mu_2 = 1$ plus irrelevant removal has the best performance. Our method with $\mu_2 = 0$ causes very bad performance. Therefore, giving some pseudo counts for terms with type mismatch is necessary. Figure 5 also shows that conventional methods without considering semantic types of each term, such as Language Model and Vector Space Model (Lucene), may have worse performance.

We also try to figure out the effect of our parameter μ_2. In fig. 6, we can see that allowing only exactly matching ($\mu_2 = 0$) results in very bad performance. $\mu_2 = 1$ is much better. When μ_2 is closed to the average number of terms in product document titles $|T(d)| = 11.4703$, the performance costs down again.

5.4 An Example

Consider a query "laptop". Obviously, a buyer issuing this query tends to buy a laptop. The top ten retrieved product document titles (translated from Chinese to English by human) by Lucene are shown in the left side of Table 4. Without considering each semantic type of each term in title, even though all these ten titles have the term "Laptop", they are obviously not laptops.

On the other hand, the right side of Table 4 are product document titles retrieved by our methods. The top eight results are truly laptops and highly related to the buyer's query. In the term type prediction phase, we have predicted the term "MSI" as a Brand and "Laptop" as a Product in the first retrieved product document. The query term "laptop" is predicted as a Product type because 71.13 percent of product document titles having this term are predicted this term as Product type in the whole corpus. Therefore, the first product document gets high score because of not only term "Laptop" matching but also type Product matching.

Table 4. A search result example comparison

1.	Ideastye **Laptop** Accessories Package	1.	MSI U100Plus Small **Laptop** (Black)
2.	Fujitsu S6120/S6120D/S6130/S6210 **Laptop** Transformer	2.	MSI U100Plus Small **Laptop** (Black)
3.	Fujitsu S2010/S2020/S2110/S2120 **Laptop** Transformer	3.	MSI U100 Small **Laptop** (White)
4.	Fujitsu S6000/S6010/S6120/S6110 **Laptop** Transformer	4.	MSI U100Plus Small **Laptop** (Black)
5.	NU Multi-functional **Laptop** Dock	5.	MSI U100Plus Small **Laptop** (N280)
6.	KINYO **Laptop** Heat Sink (NCP-001)	6.	IdeaPad S10 Magic Butterfly Long Performance Mini **Laptop** (Black)
7.	OMEGA Lightweight **Laptop** Bag	7.	IdeaPad Magic Butterfly 10-inch S10 Mini **Laptop** (Black)
8.	**Laptop** USB Cooling Plate	8.	MSI U100Plus (N280) Small **Laptop** (White)
9.	New **Laptop** RS233 Parallel Expresscard	9.	ACER TravelMate 290 4050Series **Laptop** Battery
10.	Pads 10.2-inch **Laptop** Bag	10.	DELL Inspiron 300M Latitude X300 **Laptop** Battery

6 Conclusion

In this work, we firstly point out the problem of product search in current on-line shopping platforms. Next, we describe our key idea to find the semantic types of each term in product document titles and propose a supervised learning method to learn a prediction model. To find relevant products, we modify Language Model to improve the relevance of search results using the prediction model.

In the future, we can try to use the type prediction model in other applications like recommendations or user interfaces in browsing products. Also, our method may be applied to other vertical search domains such as movie search, news search, or blog search.

References

1. LIBSVM: a library for support vector machines, `http://www.csie.ntu.edu.tw/~cjlin/libsvm`
2. Gil, A., Garcíai, F.: E-Commerce Recommenders: Powerful Tools for E-business. Crossroads 10(2), 6–6 (2003)
3. Jammalamadaka, R.C., Chittar, N., Ghatare, S.: Mining Product Intention Rules from Transaction Logs of an Ecommerce Portal. In: Proceedings of the 2009 International Database Engineering and Applications Symposium, pp. 311–314. ACM, New York (2009)
4. Kim, Y.S., Yum, B.J., Song, J., Kim, S.M.: Development of a recommender system based on navigational and behavioral patterns of customers in e-commerce sites. Expert Syst. Appl. 28(2), 381–393 (2005)
5. Lin, R., Kraus, S., Tew, J.: Attaining Fast and Successful Searches in E-commerce Environments. In: Sebastiani, F. (ed.) ECIR 2003. LNCS, vol. 2633, pp. 120–134. Springer, Heidelberg (2003)
6. Lin, R., Kraus, S., Tew, J.: OSGS - A Personalized Online Store for E-commerce Environments. Inf. Retr. 7(3-4), 369–394 (2004)

7. Parikh, N., Sundaresan, N.: Inferring Semantic Query Relations from Collective User Behavior. In: Proceeding of the 17th ACM Conference on Information and Knowledge Management, pp. 349–358. ACM, New York (2008)
8. Parikh, N., Sundaresan, N.: Scalable and Near Real-Time Burst Detection from eCommerce Queries. In: Proceeding of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 972–980. ACM, New York (2008)
9. Sarwar, B., Karypis, G., Konstan, J., Riedl, J.: Analysis of Recommendation Algorithms for E-Commerce. In: Proceedings of the 2nd ACM Conference on Electronic Commerce, pp. 158–167. ACM, New York (2000)
10. Wang, H.F., Wu, C.T.: A strategy-oriented operation module for recommender systems in E-commerce. In: Proceedings of the 9th WSEAS International Conference on Applied Informatics and Communications, pp. 78–83. WSEAS Stevens Point, Wisconsin (2009)