Predicting Primary Categories of Business Listings for Local Search

Changsung Kang, Jeehaeng Lee, Yi Chang
Yahoo! Labs
Sunnyvale, CA
{ckang,jeehaeng,yichang}@yahoo-inc.com

ABSTRACT

We consider the problem of identifying primary categories of a business listing among the categories provided by the owner of the business. The category information submitted by business owners cannot be trusted with absolute certainty since they may purposefully add some secondary or irrelevant categories to increase recall in local search results, which makes category search very challenging for local search engines. Thus, identifying primary categories of a business is a crucial problem in local search. This problem can be cast as a multi-label classification problem with a large number of categories. However, the large scale of the problem makes it infeasible to use conventional supervised-learning-based text categorization approaches.

We propose a large-scale classification framework that leverages multiple types of classification labels to produce a highly accurate classifier with fast training time. We effectively combine the complementary label sources to refine prediction. The experimental results indicate that our framework achieves very high precision and recall and outperforms a Centroid-based method.

Categories and Subject Descriptors

H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval—*Retrieval models*

General Terms

Algorithms, Design, Experimentation

Keywords

Vertical search, Text categorization, Primary category

1. INTRODUCTION

Local search is growing faster than Web search with more people using mobile devices. Studies show that at least 20% of Web queries have some local intent [9]. In local search, category queries such as "Restaurants" are known to be harder than business name queries such as "Best Buy" in the sense that the ranking quality for these queries by search engines is inferior [6].

One of the challenges in category queries in local search is that the category descriptions submitted by business owners are often incorrect. They often add some secondary or irrelevant categories to increase recall. For example, the owner of a Japanese restaurant may add "Korean Restaurants" to the category description of the business, hoping that the business may appear in the search results for the query "Korean Restaurants" as well as for the query "Japanese restaurants". This motivates the problem of identifying *primary (or true) categories* of a business. This can be considered as an multi-label classification problem [7] in which we can assign multiple primary categories to a business. There has been a lot of research in text categorization [4, 3, 1]. See [5, 8] for comprehensive surveys of the topic. However, the large number of categories in local search (2K categories in total) makes it impossible to use conventional supervised-learning-based text categorization methods.

In this paper, we present a solution to the large-scale primary category prediction (multi-label classification) problem by combining three complementary label sources:

- Labels provided by human judges
- Labels provided by business owners
- Click signals provided by users in local search

In our proposed solution, a set of highly predictive features are derived from labels from business owners and click signals and then a classifier is trained from these features using labels by human judges as targets. The experimental results demonstrate that integrating these multiple sources of labels is highly beneficial for a large-scale classification problem.

2. PREDICTING PRIMARY CATEGORIES

In this section, we propose a machine learning approach to identify primary categories of a business listing page.

2.1 Problem

Let $D = \{d_1, d_2, \ldots\}$ be the set of all business listing pages stored in a local search index and $C = \{c_1, c_2, \ldots\}$ be the set of all categories for local businesses defined by human editors. Let $T = \{t_1, t_2, \ldots\}$ be the set of all terms appearing in D. We assume that each business page d is represented by a vector space model. The set of categories assigned to a business listing d is denoted by C_d. We use $D_c = \{d \mid c \in C_d, d \in D\}$ to denote the set of all listings that have c as

CIKM'12, October 29–November 2, 2012, Maui, HI, USA.

Figure 1: An example of a business listing page. Categories are highlighted in a red box.

one of the categories. A category c is a *primary category* of a business listing d if c represents one of the main categories of d. The primary category classification problem is posed as follows: Given a business listing d and a category $c \in C_d$, determine whether c is a primary category of d.

Figure 1 shows an example of a business listing page. Based on the above definitions, C_d of this listing is the set {"Steak Houses", "Restaurants", "Carry Out & Take Out", "American Restaurants", "Seafood Restaurants"}. These categories are either provided by the owner of the business or a third-party information provider. In this example, only "Seafood Restaurants" is a primary category of the business and the other categories are either secondary or irrelevant.

Note that our problem can be seen as a multi-label text categorization problem since it is reasonable to assume that a business may have multiple primary categories. Also, we assume that there is at least one primary category for one business listing. This assumption leads to a useful feature normalization, which is discussed in Section 2.3.3.

2.2 Proposed Solution

The main challenge for our problem is the large number of categories (2K categories in total), which makes it very difficult to apply conventional supervised-learning-based text categorization approaches. Obtaining enough labels to train a classifier for each category is not feasible. Thus, we need to leverage some other types of pseudo labels to train a classifier. There are two such pseudo labels available for our problem. One is the category description C_d assigned to listings. Although C_d contains some incorrect categories, a collection of listings with the same category carry important information about the category. The other is the user clicks gathered from local search click logs. Category names (e.g., "chinese restaurants") are common queries in local search. User clicks on business listings in the search results provide important signals about the relationship between categories and business listings.

Our proposed solution is as follows. We first derive a set of features $\mathbf{x}$ to be used in our classifier from the above two pseudo-label sources (discussed in detail in Section 2.3). Note that a feature vector $\mathbf{x}$ is defined for a business listing-category pair (d,c). For example, the click-through rate of d for c (from the search results when c is used as a query) is such a feature. Then, we train a classifier $f(d, c)$ using training data $\{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots\}$ where y_i is a label provided by human judges.

Table 1 compares the three different types of label sources leveraged in our solution. It is clear that the three sources complement one another in terms of accuracy and coverage. Since categories by owners and user clicks have large coverage, they are appropriate for being used as targets to generate features (discussed in Section 2.3 in detail). On the other hand, the editorial labels by human judges are very accurate although they are not enough to train a classifier for each class. Thus, we use it as the final learning targets to combine the features (in Section 2.4).

2.3 Features

In this section, we discuss how features are derived. Note that each feature is defined for a business listing-category pair (d,c) to be used as a signal for our classifier $f(d, c)$.

2.3.1 Centroid-based Similarity Features

For each category $c \in C$, we define

$$Centroid_c = \sum_{d \in D_c} d$$

where d is a business listing represented as a tf-idf weight vector. In other words, $Centroid_c$ is the cumulated weight vector for all the listings that share the category c in their assigned categories. Then, we can compute the cosine similarity measure between a business listing d and the centroid vector $Centroid_c$ for a category c:

$$cosine\_sim(d, c) = \frac{d \cdot Centroid_c}{||d||\ ||Centroid_c||} \quad (1)$$

We use $cosine\_sim(d, c)$ as a feature for our classifier. The motivation for this features is as follows. Each centroid vector $Centroid_c$ is a mixture of the true distribution of terms for the category c and the noise due to errors in C_d. However, the true distribution dominates the centroid since the error rate of categories assigned by owners is low (around 10%). Also, it should be noted that even when the owner of a business may assign secondary or irrelevant categories to the category description part of the content (C_d) to increase recall for local search, it is less likely for the owner to corrupt the overall content of the business data (d). For example, "Korean Restaurants" can be easily added to the category description of a Japanese restaurant by the owner hoping that the business may appear in the search results for the query "Korean Restaurants" as well as for the query "Japanese restaurants". However, the owner would not add many Korean menu items to the content. Hence, we expect that $cosine\_sim(d, c)$ will be high when c is a primary category of d and low otherwise. Indeed, this hypothesis is verified in the experimental results which show that the centroid-based features are very effective for our classification problem.

To reduce the dimension of the document vectors and minimize the noise in the centroids, we propose another similarity feature based on new centroids generated by the χ^2 method [10]. The χ^2 statistic score $\chi^2(t, c)$ for a term t and a category c is

$$\frac{N(N_{r+}N_{n-} - N_{r-}N_{n+})^2}{(N_{r+} + N_{r-})(N_{n+} + N_{n-})(N_{r+} + N_{n+})(N_{r-} + N_{n-})}$$

where N_{r+} is the number of times t and c co-occur, N_{n+} is the number of times t occurs without c, N_{r-} is the number of times c occurs without t, N_{n-} is the number of times neither

Label sources	Accuracy	Coverage (how many listings are covered)	Usage
Categories assigned by owners (C_d)	medium	high	features in Section 2.3.1
User clicks	high	medium	features in Section 2.3.2
Editorial labels	high	low	labels y in training & testing

Table 1: Comparing different types of label sources.

c nor t occurs, and N is the total number of documents. If the χ^2 statistic score for t is high, it belongs to characteristic vocabulary of c.

Using χ^2 statistic scores, we generate a reduced set of terms $T'_c = \{t \in T \mid \chi^2(t,c) > \alpha\}$ for each category c. Then, we generate a new centroid filtered by the reduced terms:

$$Centroid'_c = (M^c)(Centroid_c)$$

where M^c is a diagonal matrix with $M^c_{ii} = 1$ if $t_i \in T'_c$ and 0 otherwise. Finally, a new similarity feature is

$$cosine\_sim\chi^2(d,c) = \frac{d \cdot Centroid'_c}{||d||\ ||Centroid'_c||} \quad (2)$$

2.3.2 Click-based Features

In local search, category queries such as "Restaurants" are very common. User clicks on business listings in the search results page provide crucial information about the relationship between the query and the clicked listings: The more clicks on a listing, the more likely the listing is to be about the query. The key observation is: When a query q matches a category name c, clicks on a listing d in the search results page for q can be translated into a positive relationship between d and c. In this section, the features for a category c are obtained from click statistics for c as a query in click logs.

The simplest form of a click-based feature is the click-through rate

$$CTR(d,c) = clicks/views \quad (3)$$

where clicks and views are for d in the query sessions for c. It is well known that CTR suffers from the position bias: The results at higher positions get more clicks regardless of their relevance. To address the position bias problem, we use the following two click measures in addition to CTR.

$$COEC(d,c) = \sum_{i=1}^{N} clicks_i \Big/ \sum_{i=1}^{N} aCTR_{p_i} \quad (4)$$

where $clicks_i \in \{0,1\}$ denotes if d was clicked in the i-th session out of N sessions in which d appeared for c, p_i is the position of d in the i-th session and $aCTR_p$ is the aggregated CTR (over all queries and sessions) for position p.

$$SKIP\_CTR(d,c) = clicks/(clicks + skips) \quad (5)$$

where *skips* is the number of sessions in which d was not clicked but some other results below d were clicked. Note that SKIP_CTR is a good approximation of so-called *attractiveness*, defined to be the probability of a click on a document given that the document is examined by the user.

2.3.3 Adding Normalized Features

The hypothesis that there is at least one primary category for each d suggests that we need to consider the relationships among the categories in C_d. To this end, we propose to add a normalized feature $normalized(feature(d,c))$ for each feature $feature(d,c)$:

$$normalized(feature(d,c)) = \frac{feature(d,c)}{\max_{c' \in C_d} feature(d,c')}$$

For example, let $C_d = \{c_1, c_2\}$ be the category set for d and $CTR(d,c_1) = 0.4$, $CTR(d,c_2) = 0.2$. Then, $normalized(CTR(d,c_1)) = 1$, $normalized(CTR(d,c_2)) = 0.5$. The intuition behind this normalization is: The category $c_{max} = \arg\max_{c' \in C_d} feature(d,c')$ is likely to be a primary category regardless of its feature value according to the above hypothesis. The relative information provided by the normalized features combined with the original features increases the predictive power of our classifier.

2.4 Classifier Training

Given training data $\{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots\}$, we use the gradient boosting method (GBDT) [2] to train a classifier $f(d,c)$. Each feature vector $\mathbf{x}$ consists of the features defined in the previous section: $\mathbf{x} = \{cosine\_sim(d,c),\ normalized(cosine\_sim(d,c)),\ CTR(d,c),\ normalized(CTR(d,c)), \ldots\}$.

The major difference between our framework and typical text categorization frameworks is that there are much fewer features in our framework but each feature is much stronger. Most classifiers for text categorization use a set of terms as features. On the other hand, we use a small number of very strong features for the classifier. Each feature used in our classifier can be even considered as a stand-alone *model*. In Section 3, we show that each feature performs reasonably well as a classifier. In this sense, the training step in our framework can be viewed as combining multiple *models* to improve prediction.

3. EXPERIMENTS

In this section, we present experimental results to validate our approach.

3.1 Data

We use data sets from a commercial local search engine. There are 20M business listings, 2K categories and 53M terms in total. That is, $||D|| = 20$M, $||C|| = 2$K and $||T|| = 53$M. We obtain labels from human judges for 42K (listing, category) pairs. We generate 12 features for the 42K (listing, category) pairs in the editorial judgment data. To generate click-based features in Section 2.3.2, we use 6-month click logs in the local search engine. In addition to the features defined in the previous section, one simple category term frequency feature and its normalized version are included. We use 50% of the data as training data and the rest as test data.

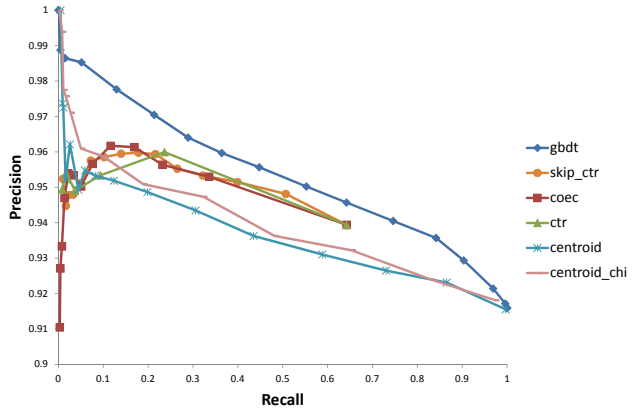

Figure 2: Precision vs. recall of different models. Each point corresponds to a threshold value for the output of a model.

3.2 Results

We evaluate the classifier trained as described in Section 2.4 using precision-recall as the evaluation metric. To see the effectiveness of our proposed method, we compare it with each of the features as a baseline:

- **gbdt**: our proposed method described in Section 2.4
- **centroid**: centroid-based similarity defined in Eq. (1)
- **centroid_chi**: centroid-based similarity with χ^2 filtering defined in Eq. (2)
- **ctr**: CTR defined in Eq. (3)
- **coec**: COEC defined in Eq. (4)
- **skip_ctr**: SKIP_CTR defined in Eq. (5)

Figure 2 shows the comparion of different models based on precision-recall. Our proposed classifier **gbdt** significantly outperforms all baselines. The results show some interesting characteristics of each feature. In general, click-based features (**ctr**, **coec** and **skip_ctr**) show higher precision than centroid-based features given the same recall. However, click-based features suffer from a sudden drop in precision as recall decreases. This happens since a very high CTR, for example, is likely to be due to a very small number of views (with a similar number of clicks). Also, we can see that click-based features have limited recall compared to centroid-based features: They can never achieve recall higher than 70%. On the other hand, centroid-based features can achieve much better recall (over 70%). Also, they do not suffer from a sudden drop in precision as recall decreases. We also observe that the χ^2-based term filtering improves prediction. It is clear that our classifier combines the benefits of different features to predict primary categories.

Table 2 shows the ordered categories by **gbdt** for "Red Lobster" in Figure 1. The primary category of the business "Seafood Restaurants" is on top while a irrelevant category "Carry Out & Take Out" is on bottom.

Category
Seafood Restaurants
Restaurants
Steak Houses
American Restaurants
Carry Out & Take Out

Table 2: Categories sorted by the gbdt outputs for "Red Lobster" in Figure 1

4. CONCLUSIONS

We presented a solution to a large-scale multi-label classification problem in the context of finding primary categories of local businesses. We showed that we can combine multiple label sources effectively to train a highly accurate classifier. Also, we demonstrated that our classifier outperforms a Centroid-based method.

One promising future direction is to investigate a method that updates centroids iteratively using a classifier. We can filter out bad categories in each listing based on the first classifier, which leads to improved centroids. We can then train a new classifier based on the improved centroids. Also, we will investigate the usefulness of adding variants of similarity features to the feature set.

5. REFERENCES

[1] W. W. Cohen and Y. Singer. Context-sensitive learning methods for text categorization. *ACM Trans. Inf. Syst.*, 17(2):141–173, Apr. 1999.

[2] J. Friedman. Greedy function approximation: a gradient boosting machine. *Ann. Statist.*, 29:1189–1232, 2001.

[3] T. Joachims. A probabilistic analysis of the rocchio algorithm with tfidf for text categorization. In *Proceedings of the Fourteenth International Conference on Machine Learning*, ICML '97, pages 143–151, San Francisco, CA, USA, 1997. Morgan Kaufmann Publishers Inc.

[4] T. Joachims. Text categorization with suport vector machines: Learning with many relevant features. In *Proceedings of the 10th European Conference on Machine Learning*, ECML '98, pages 137–142, London, UK, UK, 1998. Springer-Verlag.

[5] T. Joachims, Y. Yang. Text categorization. *Scholarpedia*, 3(5):4242, 2008.

[6] C. Kang, X. Wang, Y. Chang, and B. Tseng. Learning to rank with multi-aspect relevance for vertical search. In *Proceedings of the fifth ACM international conference on Web search and data mining*, WSDM '12, pages 453–462, New York, NY, USA, 2012. ACM.

[7] F. Sebastiani. Machine learning in automated text categorization. *ACM Comput. Surv.*, 34(1):1–47, Mar. 2002.

[8] F. Sebastiani. Text categorization. In *Text Mining and its Applications to Intelligence, CRM and Knowledge Management*, pages 109–129. WIT Press, 2005.

[9] P. Venetis, H. Gonzalez, C. S. Jensen, and A. Y. Halevy. Hyper-local, directions-based ranking of places. *PVLDB*, 4(5):290–301, 2011.

[10] Y. Yang and J. O. Pedersen. A comparative study on feature selection in text categorization. pages 412–420. Morgan Kaufmann Publishers, 1997.