A POI Categorization by Composition of Onomastic and Contextual Information

Su Jeong Choi Hyun-Je Song Seong-Bae Park Sang-Jo Lee
School of Computer Science and Engineering
Kyungpook National University
Daegu 702-701, South Korea
{sjchoi,hjsong,sbpark,sjlee}@sejong.knu.ac.kr

***Abstract*—Point of interest (POI) categorization is the task of finding of categories of POIs within a document. Because the documents that possess POIs have clue words for identifying POI categories, the task can be solved as document classification. However, this approach misses two crucial factors for identifying the category of a POI. First, the approach pays no attention to onomastic information, even though POI names reveal much categorical information in many cases. Second, the approach ignores the fact that most clue words for identifying a POI category are located near the POI name. This paper proposes a novel method that incorporates both onomastic and local contextual information in POI categorization. The proposed method uses support vector machines (SVMs) to categorize POIs. In order to utilize the onomastic information of POIs, The proposed method adopts the string kernel that manages variations of the POI names efficiently at the character level. The method also proposes a Gaussian weighting to content words in a document. By setting the mean of a Gaussian weighting at the position of a POI name, the method imposes higher weights to the words near the POI name and lower weights to the words far from the name. Then, these two types of information are combined by a composite kernel of the string kernel and a linear kernel with the Gaussian weighting. A series of experiments prove that SVMs with the combined information outperforms those with single information.**

***Keywords*—POI categorization, Onomastic information, Contextual information, Kernel composition**

I. Introduction

A *point of interest* (POI) is a specific geographical entity such as a library or a restaurant. It has a number of attributes including name, category, address, and offering services, and these attributes are important for providing POI-related services. However, to gather manually the attributes of a number of POIs requires great effort and cost, because most POIs are non-permanent and are distributed within a large area. However, a number of documents, such as reviews, articles, or advertising descriptions are now available online. That is, it is possible to gather easily documents that include information on specific POIs. Therefore, the attributes can be also obtained automatically by analyzing such documents.

The category of a POI is one of the most widely used POI attributes. The service of many location-based applications is strongly related with the category of POIs. For instance, Foursquare[1] users often query category-based questions such as "What is the most famous *restaurant* nearby?" In order to answer such questions, the categories of all POIs should be known in advance. As a result, a process for finding the category of a POI automatically is required. If a number of documents that include POIs are available, categories of POIs can be determined automatically from the documents. We call this process as *POI categorization*.

One simple solution for POI categorization is to formulate it as a document classification task. That is, when a document that contains a POI is given, the category of the document is first determined, and then the category of POI is decided as that of the document. This approach is simple and shows relatively good performance. However, it has two problems to be used for POI categorization. The first problem is that it loses information within POI names. Even if many POIs expose their attributes through their names, document classification treats the names as normal words and ignores specific information within the names. The second is that the importance of all words appearing in a document is uniform in document classification, whereas the words around the mention of a POI are more important than those far from the POI mention.

This paper proposes a novel method that overcomes the problems. Basically the proposed method classifies POIs using SVMs. In this classification, the SVMs uses two types of information about a POI that can be obtained from a document to solve the problems. One type is *onomastic information* that is derived from the name of a POI. Many POIs reveal their categories in their names. For instance, one can recognize easily that 'Lou Mitchell's Restaurant' in Chicago is a restaurant. Therefore, the use of onomastic information solves the first problem. In order to use the onomastic information in SVMs, we adopt the string kernel [1], a well-known kernel to handle subsequences of strings. It is adopted because it effectively manage not only continuous subsequences, but also discontinuous subsequences. Thus, the SVMs with a string kernel copes with spelling variants of POI names.

The other type is *contextual information* that is derived from words surrounding a POI. Note that the words appearing near a POI name have a higher possibility to be relevant to the POI than those far from the POI name. Therefore, we solve the second problem by modeling the local importance of the words appearing in a document. Using a Gaussian function whose mean is set to the position of the POI name. As a result, the closer a word is to the POI name, the higher is its importance. In order to combine these two types of information, a composite kernel is adopted [2]. That is, the final POI classifier is the SVMs of which kernel is a composite

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

978-1-4799-4143-8/14 $31.00 © 2014 IEEE
DOI 10.1109/WI-IAT.2014.78

kernel of a string kernel and a linear kernel with Gaussian weighting.

The proposed method is evaluated with a Yelp data set collected using the APIs provided by Yelp. In the experiments on string kernel, the SVMs with a string kernel outperforms other word-based or character-based kernels. Thus, it can be stated that string kernel is adequate to express onomastic information. In addition, in the experiments for the contextual information, the proposed Gaussian weighting enhances the performance of POI categorization. We also show that the SVMs with a composite kernel achieves the higher performance than any SVMs with a single information. These results prove that the local importance is an important factor and the combination of onomastic and contextual information results in good performance for POI categorization.

The rest of this paper is organized as follows. We present related studies on POI categorization in Section 2. Section 3 introduces the problem of POI categorization, and Section 4 explains the proposed method to consider onomastic and contextual information. Then, Section 5 presents the experimental results. Finally, Section 6 draws conclusions.

II. Related work

Document classification aims to assign a document automatically to a predefined class. Since there exist various kinds of documents, it has a number of applications. For instance, spam mail filtering removes spam mails from incoming mails of e-mail users [3], and sentiment analysis from blog identifies the sentimental mood of the blog writer through her blogs [4]. Not only document classification but also its applications show good performance without other outer resources [5], [6].

With the recent popularity of location-based services, POIs have become a core component for the services. As a result, there have been many studies on POIs, but most previous studies on POIs have focused on *POI localization*, a task of finding the locations of POIs. Adam et al. tried to localize POIs using a location model inferred from Flickr data [7]. Based on the study of Serdyukov et al. [8], they mapped POIs into a quantized coordinate system. Amitay et al. assigned a geographic location to each geographic mention within a document with several heuristics [9]. In addition, there have been a few studies to predict POI locations from geo-tagged multimedia data. In particular, a set of geo-tagged images associated with POIs from Flickr is widely used [8], [10]. However, because of GPS errors in the range of nearly a hundred meters, the location model from Flickr data is insufficient to localize POIs accurately. Moreover, as the coordinate system is quantized in more than one kilometer, the predicted locations of POIs are imprecise.

To the best of our knowledge, there have been no study on POI categorization. We believe that this is because POI categorization is somewhat different from document classification. In order to categorize POIs, not only the document itself but also POI names should be considered simultaneously. Therefore, the problem can be regarded also as a named entity recognition (NER) problem [11]. That is, it can be a sequential classification of tokens in a document into a predefined class such as ‘Restaurants’ or ‘Education’. However, NER does not correspond to POI categorization perfectly, because it focuses on local characteristics of POIs. That is, a sequential tagger designed to categorize POIs from their local context fails when considering long-range context. In addition, as the number of classes increases, it requires a greater quantity of labeled sequence data that are not easily obtained. Therefore, both local and long-range context should be considered in POI categorization.

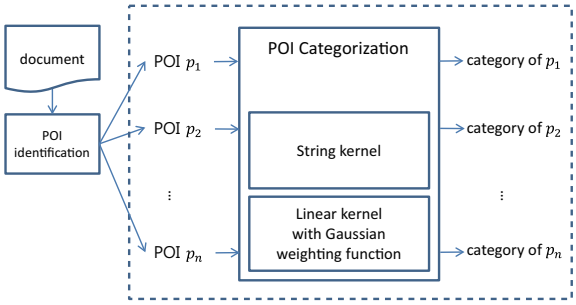

Fig. 1. Overall process of POI categorization.

There have been a few studies on local contextual information [12], [13]. The previous studies tried to solve disambiguation tasks when the targets of the tasks share spelling but have multiple meanings like location names. Li et al. tried to solve *toponym resolution*, a task to identify the correct *sense* of ambiguous location names [12]. For example, when the city name ‘Buffalo’ appears in a document, they identified its proper location from 23 different cities named as ‘Buffalo’. Under the assumption that the context around a location name is a reliable source for deciding the sense of a location, they modeled the local context using lexical grammar patterns. One example of such patterns is ‘city of + location1 + , + location2’ to find ‘city of Albany, New York’. Lieberman and Samet proposed a feature-based approach to solve the same problem [13]. They observed that when a toponym is presented, the geographic names appearing close to the toponym usually aid to resolve the toponym. In order to reflect this observation, they defined *proximity* of a toponym as an average of geographic distances from the toponym to other geographic names appearing that its local context. Then, they resolved toponyms using proximity and other geographical features such as population. Although these studies proved that toponym resolution can be solved by reflecting the local context, they paid no attention to long-range context. This paper adopts the Gaussian distribution that has a bell curve to order to consider local context and long-range context simultaneously.

III. POI Categorization

Figure 1 depicts the overview process of POI categorization. When a document is given, the phrases that correspond to POI names in the document are recognized first. This process of finding POI names within a documents is called as POI identification or POI detection. A few previous studies addressed POI identification [7], [14] as part of NER [11]. For instance, conditional random fields (CRFs) [15] are applied to POI identification. They trained CRFs with various types of features including lexical features (word shape, prefix,

Fig. 2. An example of POI categorization from Yelp.

and suffix), geographical features from Yahoo! Placemaker[2], grammatical features, and statistical features. Because most previous studies show high performance in POI identification [7], we do not pay attention to this problem. Rather, we assume that POIs in a document are all identified in advance.

The main focus of this paper is POI categorization, which is marked with a dashed box in Figure 1. The result of POI identification is P, a set of POIs that appear in the document. Thus, POI categorization takes each POI, $p_i \in P$ and the document as input, and determines the category of p_i using the information within the document.

Figure 2 shows an example of POI categorization. This document is collected from Yelp reviews. It contains one POI, 'the Codmother Fish and Chips' which is marked with an underline. When categorizing the POI, two types of information are available from the document. First, the category of the POI can be found out from the POI name. Because the POI name contains the words 'fish' and 'chips', its category can be easily recognized as a restaurant. That is, the onomastic information is one of the sources used to categorize POIs. Contextual information is another source. All content words of the document affect POI categorization to a certain degree. For instance, the words 'food', 'eat', 'fries', and 'money' are related to the category of 'the Codmother Fish and Chips'. However, note that the effects of 'food', 'eat', and 'fries' are stronger than that of 'money'. This is because 'food', 'eat', and 'fries' are located closer to 'the Codmother Fish and Chips' than 'money'. Therefore, the distance of each word to a POI name is important in using contextual information.

POI categorization can be understood as a classification task. When a POI $\mathbf{x}_i \in \mathcal{R}^n$ is given that is represented as an n-dimensional vector for p_i, its category, y^*, is determined by

$$y^* = f(\mathbf{x}_i),$$

where $f(\cdot)$ is a classification function. In this paper, SVMs is adopted as the classification function because it is one of the widely-used and high-performing models for classification tasks. SVM is trained as follows. Assume that a data set $D = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_i, y_i), \ldots, (\mathbf{x}_N, y_N)\}$ is given, where $y_i \in \mathcal{Y}$ is the category of POI p_i. Temporarily assume that Y has only two members. That is, $Y = \{+1, -1\}$. Then, SVM is trained by finding an optimal hyperplane $\mathbf{x} \cdot \mathbf{w} + b$ satisfying

$$\begin{aligned} \mathbf{x}_i \cdot \mathbf{w} + b \geq +1 \quad &\text{for} \quad y_i = +1, \\ \mathbf{x}_i \cdot \mathbf{w} + b \leq -1 \quad &\text{for} \quad y_i = -1, \end{aligned}$$

for all $(\mathbf{x}_i, y_i)$'s. The parameters $\mathbf{w}$ and b are optimized using training data D. This optimization for a soft-margin SVM is defined as

$$\min_{\mathbf{w},\xi} \frac{1}{2}||\mathbf{w}||^2 + C\sum_{i=1}^{N} \xi_i, \tag{1}$$

subject to

$$y_i(\mathbf{x}_i \cdot \mathbf{w} + b) \geq 1 - \xi_i, \text{ and } \xi_i \geq 0 \quad \forall i,$$

where C is a user parameter to control the trade-off between allowing training errors and forcing rigid margin, and ξ_i is a slack variable. The optimization problem in Equation (1) is equivalent to the following dual quadratic problem,

$$\min_{\boldsymbol{\alpha}} -\sum_{i=1}^{N} \alpha_i + \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} y_i y_j \alpha_i \alpha_j \mathbf{x}_i \cdot \mathbf{x}_j \tag{2}$$

subject to

$$\begin{aligned} &\sum_{i=1}^{N} y_i \alpha_i = 0, \\ &\forall i, 0 \leq \alpha_i \leq C, \end{aligned}$$

where $\boldsymbol{\alpha}$ is a vector of the Lagrange multiplier α_i's. POI categorization normally has more than two classes. That is, it is a multi-class classification task rather than a binary classification task. In order to solve a multi-class classification task with SVM, we adopt the one-against-one approach [16].

A key feature of SVMs is that it depends only on the inner products of training data, which allows us to use a kernel technique, so-called *kernel trick*. Kernel trick is a technique to deal with the situation in which data points are not linearly separable. If data points are linearly non-separable, they are transformed into a higher order space Φ in which they get linearly separable. That is, $\mathbf{x}_i \cdot \mathbf{x}_j$ in Equation (2) becomes $\Phi(\mathbf{x}_i) \cdot \Phi(\mathbf{x}_j)$. A kernel $K(\mathbf{x}_i, \mathbf{x}_j)$ computes $\Phi(\mathbf{x}_i) \cdot \Phi(\mathbf{x}_j)$ without mapping data points into the higher order space explicitly. Therefore, we can compute the kernel values directly and efficiently without explicitly representing the feature vectors. Moreover, it can be also used when data points have a complex data structure. When $K(\mathbf{x}_i, \mathbf{x}_j)$ is used instead of $\Phi(\mathbf{x}_i) \cdot \Phi(\mathbf{x}_j)$, Equation (2) becomes

$$\min_{\boldsymbol{\alpha}} -\sum_{i=1}^{N} \alpha_i + \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} y_i y_j \alpha_i \alpha_j K(\mathbf{x}_i, \mathbf{x}_j). \tag{3}$$

IV. POI Categorization by Combining Onomastic and Contextual Information

In this paper, we make use of two types of information that can be obtained from a document for POI categorization. According to the type of information regarding a POI p_i, $\mathbf{x}_i$ is expressed differently. It represents the name of p_i for onomastic information. Because POI names are usually a short

[2] http://developer.yahoo.com/boss/geo/

phrase, a string kernel is used for the information. On the other hand, in the contextual information, $\mathbf{x}_i$ consists of the words appearing in the document. Thus, a linear kernel is used for contextual information, because it shows relatively high performance in document classification [17].

A. String Kernel for Onomastic Information

For many POIs, their names provide clear hints about their category. A simple representation of POI names is a vector of the words used in the names. This vector representation is useful, because some words used in POI names provide much information about the category like the word 'chips' in Figure 2. However, it suffers from two problems in POI categorization. The first problem is that it loses character-level information. For instance, let us consider two restaurants named 'The Codmother Fish and Chips' and 'Cape Cod Shellfish & Seafood' respectively. These two restaurants share no common words. However, they are similar because they share the character sequences 'cod' and 'fish'. If POI names are expressed at the word level, such common sequences are lost. The second problem is that the word-level representation of POI names causes the data sparseness problem. Since many POI names use proper nouns such as a person's name or place's name, many words used in a POI name do not appear again in the names of other POIs.

The character n-gram of POI names is another candidate to represent a POI name as a vector. Even if it can express the information at the character level, it fails in expressing non-continuous character sequences. Because a POI name can have diverse variants, non-continuous character sequences are important information. For instance, let us consider 'Roma Restaurant' in New York and 'Roma Ristorante' in New Jersey. They both are restaurants. The words 'Restaurant' and 'Ristorante' are completely different at the word level, but look similar each other. Their similarity is derived from the fact that they share continuous and noncontinuous character sequences. That is, they both have a continuous sequence 'rant' and a non-continuous sequence 'R?st'. Therefore, both continuous and non-continuous character sequences are important in manipulating onomastic information.

String subsequence kernel, also known as *string kernel* computes the inner product of two strings efficiently using both continuous and non-continuous character sequences [1]. First, this kernel generates all possible subsequences of length k. A subsequence is any ordered sequence of k characters occurring in a string though not necessarily contiguously. In order to emphasize contiguous ones, subsequences are weighted by an exponentially decaying factor that penalizes the length of gaps.

Let Σ be a finite set of alphabets and Σ^k be a set of all finite strings of length k. A string s is a finite sequence of characters from Σ and $|s|$ is its length. Then, u is a subsequence of s if there exists an index vector $\mathbf{i} = (i_1, i_2, ..., i_{|u|})$ such that $1 \leq i_1 < ... < i_{|u|} \leq |s|$ and $u_j = s_{i_j}$ for $j = 1, \ldots, |u|$, or $u = s[\mathbf{i}]$ for short. A string s is expressed as a $|\Sigma^k|$-dimensional vector in which each element corresponds to a specific subsequence. That is, each coordinate of s is defined as

$$\phi_u(s) = \sum_{\mathbf{i}:u=s[\mathbf{i}]} \lambda^{l(\mathbf{i})}.$$

Here, $l(\mathbf{i})$ is the length of an occurrence of u with an index vector $\mathbf{i}$, and $0 \leq \lambda \leq 1$ is a decay factor to penalize longer subsequences. Therefore, $\phi_u(s)$ measures the number of occurrences of u in s weighted by their lengths.

Assume that $\mathbf{x}_i$ and $\mathbf{x}_j$ are two POI names. Then, the string kernel $K_k(\mathbf{x}_i, \mathbf{x}_j)$ used in Equation (3) is defined as

$$\begin{aligned} K_k(\mathbf{x}_i, \mathbf{x}_j) &= \sum_{u \in \Sigma^k} \langle \phi_u(\mathbf{x}_i) \cdot \phi_u(\mathbf{x}_j) \rangle \\ &= \sum_{u \in \Sigma^k} \sum_{\mathbf{i}:u=\mathbf{x}_i[\mathbf{i}]} \sum_{\mathbf{j}:u=\mathbf{x}_j[\mathbf{j}]} \lambda^{l(\mathbf{i})+l(\mathbf{j})}. \end{aligned} \tag{4}$$

Since $K_k(\mathbf{x}_i, \mathbf{x}_j)$ is computed based on all possible u's, it utilizes both continuous and non-continuous character sequences. However, the direct computation of $K_k(\mathbf{x}_i, \mathbf{x}_j)$ is computationally too expensive. Its complexity is $O(|\Sigma|^k)$, where $|\Sigma|^k$ is extremely large. However, Lodhi et al. proposed a dynamic programming technique for computing $K_k(\mathbf{x}_i, \mathbf{x}_j)$ [1]. Thus, it can be computed in a linear time to the string length, rather than $|\Sigma|^k$.

B. Linear Kernel with Gaussian Weighting for Contextual Information

Note that onomastic information is not enough to identify a POI category. This is because there are a number of POIs whose name consists of common nouns, or there exist a number of POIs that share a name but belong to different categories. Figure 3 shows such an example. Both Figure 3(a) and 3(b) are reviews about POIs whose name is 'Urban Animal'. However, the POI in Figure 3(a) has a Yelp category 'Pets', while that in Figure 3(b) belongs to a category 'Active'.

Contextual information is another clue to identify the category of a POI. It provides information about the category that is not available in onomastic information. That is, some words in a document play a role of *hint* to classify POIs. For instance, the words 'food' and 'eat' in Figure 2 indicate that 'The Codmother Fish and Chips' belongs to 'Restaurants' category. 'Urban Animal' in Figure 3 has different contextual information even if it has an identical name in both Figure 3(a) and 3(b). The words 'dog', 'dachshund', and 'pet' are contextual information in Figure 3(a), while the words 'fitness' and 'muscle' become contextual information in Figure 3(b). Thus, 'Urban Animal' can be categorized differently according to the different representations of contextual information.

The simplest way to use contextual information for POI categorization is to cast it as a document classification. Let $d_{\mathbf{x}}$ be a document in which a POI $\mathbf{x}$ appear. When $\mathcal{W}$ is a vocabulary set, the contextual information vector for $\mathbf{x}$ is

$$\left\langle \Phi(w_1), \ldots, \Phi(w_i), \ldots, \Phi(w_{|\mathcal{W}|}) \right\rangle,$$

where $\Phi(w_i)$ is the weight of $w_i \in \mathcal{W}$. One of the most popular representations for $\Phi(w_i)$ is the term frequency (TF) that represents the weight of w_i as its frequency in a document. However, it is not adequate for POI categorization, because the weights do not reflect the distance of the words positions from the position of the target POI name. As a result, if two or more different POIs appear in a document, they will have an identical contextual information vector. Thus, even though the POIs have different categories, they can not be distinguished though

I started going here with two elderly dachshunds with attendant health problems. The staff at Urban Animal are amazing. They helped us make our girl-dog comfortable in her final months and helped us say goodbye to her. They are compassionate and conscientious about doing the right thing for you and for your pet. Our boy-dog will continue to be a loyal consumer of their excellent service!

(a) A Yelp review about a POI 'Urban Animal' whose category is 'Pet'

The great thing about Urban Animal is that Laura will push you, no matter your skill level, and you end every week feeling more fit than the last. I started Urban Animal to get back in shape after a broken collarbone and couldn't be happier with how it has improved my overall strength and fitness. I've done both Urban Animal and CrossFit, and although the Urban Animal classes ..

(b) A Yelp review about a POI 'Urban Animal' whose category is 'Active'

Fig. 3. An example of POIs which have the same name but different category.

TF-based contextual information. In POI categorization, all words in a document are not equally important to identify the category of a POI. In general, the words around a POI name are more important than those far from the name.

In order to solve this problem, we adopt the Gaussian distribution that allows the reflection of locality into word weighting. Gaussian distribution has a bell-curve shape with its peak at the mean of input data. To emphasize POI names in the contextual information, we set the mean at the position of a POI name. As a result, the words around the POI name receive large weights, while those far from the name get low weights.

Let $w_{\mathbf{x}}$ be the name of a POI $\mathbf{x}$ and $position(w_i)$ be the position of w_i in a document $d_{\mathbf{x}}$. Then, $lw(w_i; w_{\mathbf{x}}, \sigma)$, the local weight of w_i with respect to $w_{\mathbf{x}}$, is defined as

$$lw(w_i; w_{\mathbf{x}}, \sigma) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(position(w_{\mathbf{x}}) - position(w_i))^2}{2\sigma^2}}, \quad (5)$$

where σ is the variance of the Gaussian distribution. The degree of this local weighting depends on σ. In this paper, we set σ as the length of the document $d_{\mathbf{x}}$ to cover long-range contexts. Therefore, this weighting is affected mainly by the distance between w_i and $\mathbf{x}$. Since local weight and term frequency are both important for contextual information, the final weight $\Phi(w_i)$ of a word w_i is defined as their multiplication. That is,

$$\Phi(w_i) = tf(w_i) \cdot lw(w_i; w_{\mathbf{x}}, \sigma), \quad (6)$$

where $tf(w_i)$ is the frequency of w_i in $d_{\mathbf{x}}$.

TABLE I. STATISTICS ON EXPERIMENTAL DATA.

Information	Value
No. of unique POIs	3,865
Total No. of reviews	22,019
Average No. of reviews per POI	5.70
Total No. of mentioned POIs	34,666
Average No. of POI mentioned per review	1.57
No. of categories	20
Average No. of POIs per category	193.25

Since document classification shows high performance with a linear kernel, we adopt a linear kernel for the contextual information in POI categorization. That is, the kernel for the contextual information, $K_c(\mathbf{x}_i, \mathbf{x}_j)$ is defined as

$$K_c(\mathbf{x}_i, \mathbf{x}_j) = \sum_{w \in \mathcal{W}} \Phi_i(w) \cdot \Phi_j(w), \quad (7)$$

where $\Phi_i(w)$ is the weight of w for POI $\mathbf{x}_i$ and $\Phi_j(w)$ is that for POI $\mathbf{x}_j$.

C. Composite Kernel for Combining Onomastic and Contextual Information

The string kernel in Equation (4) and the local linear kernel in Equation (7) manage the onomastic and contextual information of a POI respectively. Thus, they handle their own information while ignoring other types of information. In order to make use of both types of information in POI categorization, the composition of the two kernels is required. Cristianini and Shawe-Taylor proved that a new kernel can be obtained by combining previous several kernels with some closure properties such as weighted sum and scalar multiplication [18]. Among various closure properties, the weighted sum property is used in this paper.

The base kernels are normalized before being combined. This is because they are not bound, and thus one kernel could dominate others in their composition. When a kernel $K(\mathbf{x}_i, \mathbf{x}_j)$ is given, its normalized version, $K'(\mathbf{x}_i, \mathbf{x}_j)$ is actually used which is defined as

$$K'(\mathbf{x}_i, \mathbf{x}_j) = \frac{K(\mathbf{x}_i, \mathbf{x}_j)}{\sqrt{K(\mathbf{x}_i, \mathbf{x}_i) \cdot K(\mathbf{x}_j, \mathbf{x}_j)}}.$$

Our composite kernel K_{co} is composed of the normalized string kernel for onomastic information and the normalized linear kernel for contextual information. That is, when a normalized string kernel K'_k and a normalized linear kernel K'_c are given, the final kernel, K_{co}, is defined as

$$K_{co}(\mathbf{x}_i, \mathbf{x}_j) = \gamma K'_k(\mathbf{x}_i, \mathbf{x}_j) + (1 - \gamma) K'_c(\mathbf{x}_i, \mathbf{x}_j), \quad (8)$$

where γ is the mixing weight between the normalized kernels. The larger γ is, the more significant K'_k is. On the other hand, when the value of γ is small, K'_c is more important than K'_k in determining POI category.

V. EXPERIMENTS

A. Experimental Setting

For the experiments, we crawled POIs and their reviews from Yelp using Yelp API[3]. First, we chose 20 famous US and

[3] http://api.yelp.com

TABLE II. STATISTICS OF EXPERIMENTAL DATA ACCORDING TO CATEGORY.

Category	# of unique POIs	# of reviews
Active	262 (6.78%)	1,798 (8.17%)
Arts	247 (6.39%)	1,661 (7.54%)
Auto	232 (6.00%)	1,133 (5.15%)
Beauty & Spa	230 (5.95%)	1,371 (6.23%)
Education	195 (5.05%)	983 (4.46%)
Event Services	217 (5.61%)	1,482 (6.73%)
Financial Services	131 (3.39%)	419 (1.90%)
Health	167 (4.32%)	796 (3.62%)
Home Services	233 (6.03%)	1,371 (6.23%)
Hotels & Travel	171 (4.42%)	826 (3.75%)
Local Services	225 (5.82%)	1,347 (6.12%)
Mass Media	125 (3.23%)	384 (1.74%)
Nightlife	222 (5.74%)	1,733 (7.87%)
Pets	254 (6.57%)	1,722 (7.82%)
Professional	138 (3.57%)	496 (2.25%)
Public Services	185 (4.79%)	828 (3.76%)
Real Estate	126 (3.26%)	447 (2.03%)
Religious	106 (2.74%)	326 (1.48%)
Restaurants	228 (5.90%)	1,742 (7.91%)
Shopping	171 (4.42%)	1,154 (5.24%)
Total	3,865 (100%)	22,019 (100%)

TABLE III. SIMPLE STATISTICS OF EXPERIMENTAL DATA ACCORDING TO DATA TYPES

	Training	Validation	Test
No. of unique POIs	2,331	767	767
Total No. of reviews	13,504	4,223	4,292
Total No. of mentioned POIs	21,262	6,687	6,717

Canadian cities, and then crawled the reviews of POIs in the cities. Yelp provides a total of 22 first-level categories[4], but we removed two, 'Food' and 'Local Flavor'. 'Food' is excluded because it is quite similar to the category, 'Restaurants' in terms of their inherent attributes about food. 'Local Flavor' refers to beloved POIs by local people. Thus, all POIs in 'Local Flavor' also belong to at least one other category. That is, 'Local Flavor' is a redundant class, and this is why it is removed from the category set. As a result, our category set has 20 members. Table I shows the simple statistics of the crawled data. The total number of unique POIs is 3,865, and the number of reviews regarding the POIs is 22,019. Thus, the average number of reviews per POI is 5.70. The total number of POIs mentioned is 34,666, and thus 1.57 POIs appear on average in a review. The average number of POIs per category is 193.25.

Table II shows the statistics of the experimental data according to category. 'Active' category has the greatest number of POIs. It has 262 unique POIs and 1,798 reviews. On the other hand, 'Religious' contains the least number of POIs. It has only 106 unique POIs, and 326 reviews.

In order to determine the parameters in Equation (4) and Equation (8), an independent validation set is required. Thus, we divided the crawled data into three parts; training, validation and test sets. All the crawled data are randomly distributed into the sets, but their size ratio is fixed at 3:1:1. We also controlled the category ratio in Table II in order to avoid generating a skewed data set. Table III shows some statistics of the data sets. The training set contains 2,331 POIs with 13,504 reviews, while the validation set has 767 POIs with 4,223 reviews. On the other hand, there are 767 POIs and 4,292 reviews in the test set. The total number of POIs mentions is 21,262 in the training set, 6,687 in the validation set, and 6,717 in the test data respectively.

[4] http://www.yelp.com/developers/documentation/category_list

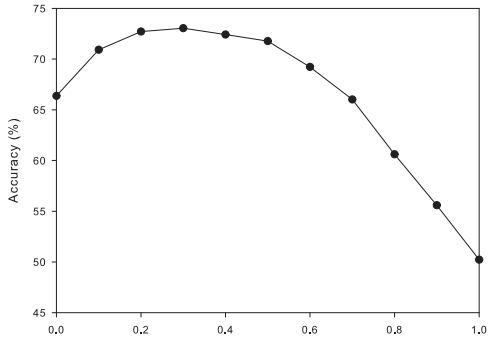

Fig. 4. Performances with various values of γ in the validation data set

The proposed method is compared with several baselines. For onomastic information, word-level representation (BOW) of POI names and character-level n-gram (Character n-gram) are adopted as baselines. For contextual information, TF and term-frequency with a window size (TF-Window-10) are used as baselines. TF-Window-10 considers only 10 words on either side of a POI name as its context. On the other hand, TF uses all words in a document. By comparing TF and TF-Window-10, we can find out the importance of context size. In addition, the comparison of the proposed method and TF demonstrates the effectiveness of local importance.

For all experiments, we removed stopwords and digits in contextual information. On the other hand, all characters, words, and digits are used in onomastic information. The performance of the methods is measured with the accuracy of POI category classification, and LIBSVM package [19] is used for the implementation of SVMs.

B. Experimental Results

The parameters of the string kernel in Equation (4) are determined using the validation set. The best performance of the string kernel is achieved when k = 4 and λ = 0.1. The string kernel shows the accuracy of 47.204% with these parameters. k = 4 implies that at most 4-character subsequences in POI names are reflected, and λ = 0.1 means that non-contiguous subsequences receive more penalty. Thus, these parameters are used in all the following experiments on the string kernel.

Figure 4 depicts the performances of the proposed composite kernel with various values of γ in Equation (8) on the validation set. $\gamma = 0$ means that only onomastic information is used, while $\gamma = 1$ implies that only contextual information is used. The string kernel with k = 4 and λ = 0.1 is used for onomastic information, while the linear kernel with local Gaussian weighting (TF-Gaussian) is used for contextual information. The best accuracy is observed at $\gamma = 0.3$ with the accuracy of 73.037%. From the result that the best accuracy is observed at $\gamma = 0.3$, it can be inferred that contextual information is slightly more significant than onomastic information. We use $\gamma = 0.3$ for all the following experiments on the test set.

TABLE IV. ACCURACY OF VARIOUS METHODS WHEN EACH POI IN A DOCUMENT IS PROCESSED INDEPENDENTLY.

	Method	Accuracy (%)
Onomastic	BOW	50.856
	Character 3-gram	47.685
	Character 4-gram	50.661
	Character 5-gram	45.928
	String Kernel (4, 0.1)	**51.959**
Contextual	TF-Window-10	52.107
	TF	66.756
	TF-Gaussian	**68.349**
Composition	Proposed method ($\gamma = 0.3$)	**73.053**

TABLE V. ACCURACY OF VARIOUS METHODS WHEN POIS WITH THE SAME NAME ARE TREATED AS AN IDENTICAL POI.

	Method	Accuracy (%)
Onomastic	BOW	52.289
	Character 3-gram	49.113
	Character 4-gram	53.406
	Character 5-gram	51.106
	String Kernel (4, 0.1)	**53.954**
Contextual	TF-Window-10	53.428
	TF	65.937
	TF-Gaussian	**66.528**
Composition	Proposed Method ($\gamma = 0.3$)	**71.610**

Table IV gives the experimental results of the proposed method and the baselines on the test set. As indicated in the table, character 4-gram shows the highest accuracy among n-gram representations. It achieves the accuracy of 50.661%. BOW whose accuracy is 50.856% results in slightly higher accuracy than the character 4-gram. However, note that the string kernel outperforms all of them in the experiments with onomastic information. Its accuracy is 51.959%, which is 1.103% higher than that of BOW and 1.298% higher than that of character 4-gram. These results imply that the string kernel is the best method for onomastic information and it manages both continuous and discontinuous subsequences well.

In the experiments with contextual information, the baseline models, TF and TF-Window-10 achieve the accuracies of 66.756% and 52.107%,respectively. The fact that TF reports higher performance than TF-Window-10 indicates that overall context in a document helps improve POI categorization. The accuracy of the proposed TF-Gaussian is 68.349% which is higher than those of TF and TF-Window-10. The proposed method also is based on term frequency. Therefore, the fact that TF-Gaussian outperforms both TF and TF-Window-10 proves that local weighting of context words is actually effective.

On the other hand, the proposed composite kernel that is composed of a string kernel (4, 0.1) and TF-Gaussian achieves the best accuracy among all compared methods. Its accuracy is 73.053%. This is an improvement of 21.094% over the string kernel and 4.704% over TF-Gaussian. Therefore, it is important to consider onomastic and contextual information simultaneously in POI categorization.

C. Treatment of Homonymic POIs

In many documents like advertisements and POI reviews, the same POI appear several times. Therefore, one possible variant of POI categorization is to treat homonymic POIs within a document as an identical POI. In order to see the effect of the variant, the variant is compared with independent processing of the POIs. For the implementation of the variant, the representation of contextual information is slightly changed, while that of onomastic information remains. When P_u is a set of POI mentions with the same name u, the word weight in Equation (6) is changed into

$$\Phi(w_i) = tf(w_i) \cdot lw^{new}(w_i; \sigma),$$

where $lw^{new}(w_i; \sigma)$ is defined as

$$lw^{new}(w_i; \sigma) = \arg\max_{pu_i \in P_u} lw(w_i; pu_i, \sigma). \quad (9)$$

Here, $lw(w_i; pu_i, \sigma)$ is defined in Equation (5). Then, the word weight is affected only by the mention with the shortest distance to the POI.

Table V shows the accuracies of various methods when the POIs with the same name are regarded as an identical one. The accuracy of the methods shows a similar tendency to those in Table IV. That is, the string kernel achieves the highest accuracy for onomastic information, and TF-Gaussian outperforms both TF and TF-Window-10 in contextual information. When compared with Table IV, the performances of the methods is improved for onomastic information and it decreases for contextual information. The reason why the performance improves for onomastic information is that the number of POIs tested is reduced because POIs with the same name are regarded as a single POI, while all methods are superior to random-guess. In addition, the reason why the methods show worse accuracy for contextual information is that some irrelevant words have high weight. According to Equation (9), the weight of a word is determined by its nearest POI mention. Thus, when a word is relevant to one POI mention but irrelevant to all other mentions, its effect is governed by the relevant case and it affects all POI mentions with the same name negatively. On the other hand, it affects only one POI in the independent processing of POIs. Because the composite kernel is more affected by contextual information, its accuracy in Table V is lower than that in Table IV. Therefore, we can conclude that it is better to process POIs independently even if they share a name.

TABLE VI. PERFORMANCE OF THE PROPOSED METHOD FOR EACH CATEGORY.

Category	Accuracy (%)
Active	80.826
Arts	68.582
Auto	86.331
Beauty & Spa	**93.985**
Education	60.248
Event Services*	61.937
Financial Services*	69.841
Health	82.379
Home Services*	68.182
Hotels & Travel	54.194
Local Services*	62.454
Mass Media	47.423
Nightlife	71.245
Pets	**97.872**
Professional	42.857
Public Services*	72.046
Real Estate	20.359
Religious	77.586
Restaurants	86.252
Shopping	62.136

D. Performance per Category

Even if the overall accuracy of the proposed method is high, its accuracy is not always high for all categories in Yelp.

Table VI shows the accuracies of the proposed method for 20 Yelp categories. The highest accuracy is achieved at 'Beauty & Spa' and 'Pets' in which the accuracy of the proposed method is 93.985% and 97.872%, respectively. In these categories, the words used in POI names reflect the category of POIs well. For instance, many POIs in 'Beauty & Spa' have the word 'spa' or 'salon' in their names, and those in 'Pets' also contain 'dog' or 'pet' in their names. As a result, the POIs in these categories are easily classified.

The accuracy for service-related categories is relatively low. The categories marked with an asterisk in Table VI are service-related ones. Their accuracy is approximately 60 $\sim$ to 70 %. The reason for the low accuracy in these categories is that most reviews in these categories describe the service quality of the POIs rather than their specific information. This phenomenon makes it difficult to classify POIs using contextual information, because contextual information in these categories becomes similar. This is also true for 'Real Estate' and 'Hotels & Travel'. A significant portion of the reviews in these categories is also written for the service of POIs. Thus, most POIs in these categories are misclassified as 'Home Services' and 'Local Services'. As a result, the accuracy of 'Real Estate' is merely 20.359% and that of 'Hotels & Travel' is 54.194%.

VI. Conclusion

We have proposed a novel method for POI categorization, a task of identifying the category of a POI within a document. Since it is a classification task, it can be solved through SVMs. In implementing SVMs for POI categorization, both onomastic information and contextual information are used. Onomastic information provides clues for classifying POIs through their names. In order to utilize the names at the character-level, we adopted a string kernel for onomastic information. In addition, words appearing with a POI can be used as a context of the POI for POI categorization. From the fact that the words near a POI are more important than those far from the POI, we adopted Gaussian weighting for the local importance of words. Then, a linear kernel with the Gaussian weighting is used for contextual information. Because both kinds of information are necessary for POI categorization, a composition kernel of the string kernel and the linear kernel is used as the final kernel for POI categorization.

Through the experiments with the Yelp data set, we have showed that the proposed kernels achieve higher accuracy than simple baselines. That is, the string kernel outperforms BOW and character n-grams, and the linear kernel with the Gaussian weighting achieves higher accuracy than the models without local weighting. This implies that the proposed base kernels reflect well the information that they manage. In addition, we showed that the composite kernel outperforms the string kernel and the linear kernel. This result proves that the simultaneous use of both types of information is important in POI categorization.

Acknowledgements

This work was supported by the Industrial Strategic Technology Development Program (10035348, Development of a Cognitive Planning and Learning Model for Mobile Platforms) funded by the Ministry of Knowledge Economy(MKE, Korea)

References

[1] H. Lodhi, C. Saunders, J. Shawe-Taylor, N. Cristianini, and C. Watkins, "Text classification using string kernels," *Journal of Machine Learning Research*, vol. 2, pp. 419–444, 2002.

[2] T. Joachims, T. J. De, N. Cristianini, and N. R. A. Uk, "Composite kernels for hypertext categorisation," in *In Proceedings of the International Conference on Machine Learning*, pp. 250–257, 2001.

[3] H. Drucker, S. Wu, and V. N. Vapnik, "Support vector machines for spam categorization," *IEEE Transactions on Neural Networks*, vol. 10, no. 5, pp. 1048–1054, 1999.

[4] P. Melville, W. Gryc, and R. D. Lawrence, "Sentiment analysis of blogs by combining lexical knowledge with text classification," in *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 1275–1284, 2009.

[5] S. Dumais, J. Platt, D. Heckerman, and M. Sahami, "Inductive learning algorithms and representations for text categorization," in *Proceedings of the Seventh International Conference on Information and Knowledge Management*, pp. 148–155, 1998.

[6] R. J. Mooney and L. Roy, "Content-based book recommending using learning for text categorization," in *Proceedings of the 5th ACM Conference on Digital Libraries*, pp. 195–204, 2000.

[7] A. Rae, V. Murdock, A. Popescu, and H. Bouchard, "Mining the web for points of interest," in *Proceedings of the 35th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 711–720, 2012.

[8] P. Serdyukov, V. Murdock, and R. van Zwol, "Placing flickr photos on a map," in *Proceedings of the 32th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 484–491, 2009.

[9] E. Amtay, N. Har'El, R. Sivan, and A. Soffer, "Web-a-where: Geotagging web content," in *Proceedings of the 27th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 273–280, 2004.

[10] D. J. Crandall, L. Backstrom, D. Huttenlocher, and J. Kleinberg, "Mapping the world's photos," in *Proceedings of the 18th International Conference on World Wide Web*, pp. 761–770, 2009.

[11] A. McCallum and W. Li, "Early results for named entity recognition with conditional random fields, feature induction and web-enhanced lexicons," in *Proceedings of the 7th Conference on Natural Language Learning*, pp. 188–191, 2003.

[12] H. Li, R. K. Srihari, C. Niu, and W. Li, "Location normalization for information extraction," in *Proceedings of the 19th International Conference on Computation Linguistics*, pp. 1–7, 2002.

[13] M. D. Lieberman and H. Samet, "Multifaceted toponym recognition for streaming news," in *Proceedings of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 843–852, 2011.

[14] B. Martins, I. Anastácio, and P. Calado, "A machine learning approach for resolving place references in text," in *Geospatial Thinking*, pp. 221–236, 2010.

[15] J. D. Lafferty, A. McCallum, and F. C. N. Pereira, "Conditional random fields: Probabilistic models for segmenting and labeling sequence data," in *Proceedings of the 18th International Conference on Machine Learning*, pp. 282–289, 2001.

[16] C.-W. Hsu and C.-J. Lin, "A comparison of methods for multiclass support vector machines," *IEEE Transactions on Neural Networks*, vol. 13, no. 2, pp. 415–425, 2002.

[17] E. Leopold and J. Kindermann, "Text categorization with support vector machines. how to represent texts in input space?" *Machine Learning*, vol. 46, no. 1-3, pp. 423–444, 2002.

[18] N. Cristianini and J. Shawe-Taylor, *An introduction to support vector machines*. Cambridge University Press Cambridge, 2000.

[19] C.-C. Chang and C.-J. Lin, "LIBSVM: A library for support vector machines," *ACM Transactions on Intelligent Systems and Technology*, vol. 2, no. 3, pp. 27:1–27:27, 2011.