Special issue: Leading ICT technologies in the Information Explosion

Technical Note

Statistical string similarity model for information linkage

Atsuhiro TAKASU
National Institute of Informatics

ABSTRACT

This paper proposes a statistical string similarity model for approximate matching in information linkage. The proposed similarity model is an extension of hidden Markov model and its learnable ability realizes string matching function adaptable to various information sources. The main contribution of this paper is to develop an efficient learning algorithm for estimating parameters of the statistical similarity model. The proposed algorithm is based on the Expectation-Maximization (EM) technique where dynamic programing technique is used to update parameters in EM process.

KEYWORDS

String similarity, statistical model, EM algorithm

1 Introduction

String similarity is a basic information for linking records and documents. When linking records, we usually measure similarity between corresponding fields, then merge them to obtain a similarity between records. String similarity is often used for field similarity (e.g., [1]). It has been studied in the literature of approximate string matching [7].

The edit distance is the most frequently used string similarity that is defined as the minimum number of edit operations required to convert one string to another. Since the cost of each edit operation is fixed, it cannot represent similarity pattern specific to objective data set.

To solve this problem several string similarity models have been proposed. They are categorized into two groups: one is a discrete similarity model, such as a confusion table and automaton; the other is a quantitative model, such as a statistical confusion matrix. Quantitative models have the advantage of constructing a more precise model when a large quantity of training data is available. Probabilistic models are especially suitable for integrating a language model with a similarity model such as in speech recognition [9].

The statistical confusion matrix [3] is a typical similarity model for OCR errors. Myka *et al.* [6] reported that it had good performance in their comparative study of text search. However, string mapping handled in the confusion matrix are limited to the (1,1) substitution operation. Li and Lopresti proposed a model where string mapping patterns were categorized from the viewpoint of string lengths [5]. In this model, a pair (i, j) of lengths of compared strings are used as string mapping pattern, and weight or cost is assigned to each pattern in order to calculate the similarity of two strings by dynamic programming matching. This model can represent insertion, deletion, and framing operations, as well as substitution operation. However, in order to utilize the model, we must determine the weight for each pattern. Furthermore, the weight is the same if the mapping pattern is the same. Ohta *et al.* [8] proposed a probabilistic automaton that handles insertion and deletion operations as well as substitution operation and showed that the model is effective for fuzzy full text searches. Ristad proposed a statistical model for the cost of edit operations [10]. In this model, the edit operations converting strings is regarded as a state transition sequence of a hidden Markov model. The cost of edit operations is defined via parameters of the model. They are estimated from training pairs of similar strings.

This paper proposes a similarity model that handles any pair of lengths of original and recognition strings including insertion, deletion, and substitution operations, and derives an efficient parameter estimation algorithm. The proposed model is similar to Ristad's model, but it is designed to handle record structure [11]

Received September 16, 2008; Revised December 24, 2008; Accepted December 25, 2008.
takasu@nii.ac.jp
DOI: 10.2201/NiiPi.2009.6.7

as well as unstructured string. The rest of this paper is organized as follows. Section 2 defines the proposed model. Section 3 proposes a parameter estimation algorithm for the model based on the expectation maximization (EM) technique. Appendix derives the update formulas in EM step.

2 Statistical String Similarity Model

In order to handle various types of string similarity, we introduce an extended hidden Markov model called the dual variable length output hidden Markov model (DVHMM) that produces pairs of similar strings [12]. The states of the DVHMM correspond to the string mapping patterns from the viewpoint of output string length, and they are characterized by a pair of lengths of similar strings. For example, a pair (2,1) of output string lengths means that two consecutive characters are similar to one character. As an output symbol, the state corresponding to a pair (i, j) produces a pair of similar strings whose lengths are i and j, respectively. For example, suppose a set {a,b} is output characters. Then, the output symbols of the state (2, 1) are $\{(aa, a), (aa, b), (ab, a), (ab, b), (ba, a), (ba, b), (bb, a), (bb, b)\}$. A state (1,0) means that the length of the original (resp. converted) output string is 1 (resp. 0), i.e., an insertion operation, whereas a state (0,1) corresponds to a deletion operation. A state (1,1) corresponds to a substitution operation. The states of the DVHMM are categorized into two groups:

- *non-null* states that produce a pair of similar strings and
- *null* states that produce a null output and work to prohibit undesirable output production.

Fig. 1 shows an example of a DVHMM in which the output symbols are omitted due to space restrictions. Three non-null states q, r and s corresponds to delete, insert, and substitute operations, respectively. On the other hand, there is one null state f that stands for the termination of state transition. With this statistical model, we start a transition from any non-null state according to the initial probability distribution, moves to any non-null state according to the transition probabilities, then stop at the null state f. At each non-null state, the model produces a pair of strings of corresponding lengths.

Let us define the DVHMM formally. First, we define the notation used in the following discussion. Generally, an upper case letter denotes a set, a bold face letter denotes a sequence and a string, calligraphic letter denotes a function symbol, and a Greek letter denotes a parameter. For a set S, S^l denotes the set of sequences of length l consisting of elements in S. $|S|$ denotes the cardinality of a set S. $|\boldsymbol{x}|$ denotes the length of $\boldsymbol{x}$, $\boldsymbol{x}_i$ denotes the ith component of $\boldsymbol{x}$, and $\boldsymbol{x}_{i:j}$ denotes a partial sequence of $\boldsymbol{x}$ starting at the ith character and ending at the jth character of $\boldsymbol{x}$. A partial sequence $\boldsymbol{x}_{:i}$ and $\boldsymbol{x}_{i:}$ denote a prefix ending at the ith component and a suffix starting at the ith component of $\boldsymbol{x}$, respectively.

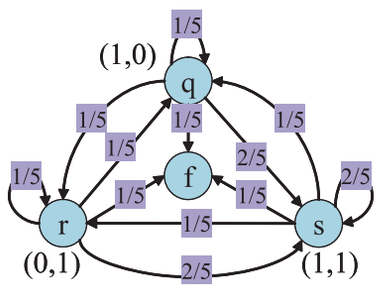

Fig. 1 An example of DVHMM

The DVHMM is denoted by a tuple $M \equiv (A, S, \boldsymbol{\theta})$.

- The set A is an alphabet.
- The set S is states consisting of non-null states S_x and null states S_0, i.e., $S = S_x \cup S_0$. Each non-null state q produces pairs of strings $\{(\boldsymbol{a}, \boldsymbol{b}) \mid \boldsymbol{a} \in A^a, \boldsymbol{b} \in A^b\}$ where a and b are the lengths of the original and converted strings of s. O_s denotes the set of pairs of strings produced by the state s.
- The parameters $\boldsymbol{\theta}$ consists of the following initial, transition, and output probabilities:
  - For a state $s \in S$, the probability $\pi(s)$ denotes the initial probability.
  - For a state s and r in S, the probability $\tau(s, r)$ denotes the transition probability from a state s to a state r.
  - For a state $s \in S$ and a pair $(\boldsymbol{a}, \boldsymbol{b}) \in O_s$ the probability $o(s, \boldsymbol{a}, \boldsymbol{b})$ denotes the output probability that a state s produces $\boldsymbol{a}$ and $\boldsymbol{b}$ as an original and a converted string, respectively.

From the definition of the DVHMM, a sequence $\boldsymbol{s} \equiv s_1 s_2 \cdots s_n$ of state transitions uniquely decomposes the pair $(\boldsymbol{u}, \boldsymbol{v})$ of strings into the sequence

$$(\boldsymbol{u}_1, \boldsymbol{v}_1), (\boldsymbol{u}_2, \boldsymbol{v}_2), \cdots, (\boldsymbol{u}_n, \boldsymbol{v}_n) .$$

This means that a state transition sequence represents an alignment of a pair of strings. The probability that a DVHMM produces the pair $(\boldsymbol{u}, \boldsymbol{v})$ with the state sequence $\boldsymbol{s}$ is given by

$$\Pr(\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{s} \mid \boldsymbol{\theta}) = \pi(s_1) \prod_{i=1}^{n-1} \{\tau(s_i, s_{i+1}) \cdot o(s_i, \boldsymbol{u}_i, \boldsymbol{v}_i)\} \tag{1}$$

[**Example 1**] For a pair $(abaa, aba)$ of original and converted strings, let us consider a sequence $\boldsymbol{s} \equiv sqss$ of state transitions shown in Fig. 1. Then, non-null states s, q, s and s produce the pairs (a, a), (b, ϕ), (a, a) and (a, a), respectively. These pairs of strings represent the alignment of $(abaa, aba)$ with $\boldsymbol{s}$.

For a pair $(\boldsymbol{u}, \boldsymbol{v})$ of strings, $Q(\boldsymbol{u}, \boldsymbol{v})$ denotes a set of sequences of state transitions that produces $(\boldsymbol{u}, \boldsymbol{v})$. Then, we can solve the alignment of $(\boldsymbol{u}, \boldsymbol{v})$ by finding the following optimal state transition sequence

$$\boldsymbol{s}^* \equiv \operatorname*{argmax}_{\boldsymbol{s} \in Q(\boldsymbol{u}, \boldsymbol{v})} \Pr(\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{s} \mid \boldsymbol{\theta}).$$

We refer to $\boldsymbol{s}^*$ as the *most likely state transition*. We use the joint probability $\Pr(\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{s}^*)$ for the most likely state transition as the similarity of the pair $(\boldsymbol{u}, \boldsymbol{v})$ of strings.

For a pair $(\boldsymbol{u}, \boldsymbol{v})$, the joint probability of the pair is obtained by

$$\Pr(\boldsymbol{u}, \boldsymbol{v} \mid \boldsymbol{\theta}) = \sum_{\boldsymbol{s} \in Q(\boldsymbol{u}, \boldsymbol{v})} \Pr(\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{s} \mid \boldsymbol{\theta}) \ .$$

DVHMM can be used to measure similarity in approximate string matching.

There are several studies on HMM handling variable length sequences such as the polygram model [4] and the multigram model [2]. The DVHMM was originally introduced to handle variable length recognition errors. However, it is similar to the multigram model in that it decomposes a sequence into partial sequences of variable length, although the DVHMM differs from the multigram model in its handling of a pair of sequences that require a more complex parameter estimation algorithm.

3 Parameter Estimation of DVHMM

This section derives a maximum likelihood (ML) parameter estimation algorithm for the model described in Section 2. The proposed algorithm is based on the EM technique.

3.1 EM Algorithm

Let $\boldsymbol{T} = \{(\boldsymbol{u}_1, \boldsymbol{v}_1), (\boldsymbol{u}_2, \boldsymbol{v}_2), \cdots, (\boldsymbol{u}_{|\boldsymbol{T}|}, \boldsymbol{v}_{|\boldsymbol{T}|})\}$ be a set of training pairs of similar strings. We assume that each string pair $(\boldsymbol{u}_i, \boldsymbol{v}_i)$ in $\boldsymbol{T}$ is generated by a DVHMM independently. Then,

$$\Pr(\boldsymbol{T} \mid \boldsymbol{\theta}) = \prod_{i=1}^{|\boldsymbol{T}|} \sum_{\boldsymbol{s}_i \in Q(\boldsymbol{u}_i, \boldsymbol{v}_i)} \Pr(\boldsymbol{u}_i, \boldsymbol{v}_i, \boldsymbol{s}_i \mid \boldsymbol{\theta}) \ . \tag{2}$$

In ML estimation, we obtain the parameter $\boldsymbol{\theta}$ that maximizes the likelihood Eq. (2). By Jensen's inequality, we obtain

$$\begin{aligned} &\ln \Pr(\boldsymbol{T} \mid \boldsymbol{\theta}) \\ &= \sum_{i=1}^{|\boldsymbol{T}|} \ln \sum_{\boldsymbol{s}_i \in Q(\boldsymbol{u}_i, \boldsymbol{v}_i)} q_i(\boldsymbol{s}_i) \frac{\Pr(\boldsymbol{u}_i, \boldsymbol{v}_i, \boldsymbol{s}_i \mid \boldsymbol{\theta})}{q_i(\boldsymbol{s}_i)} \\ &\geq \sum_{i=1}^{|\boldsymbol{T}|} \sum_{\boldsymbol{s}_i \in Q(\boldsymbol{u}_i, \boldsymbol{v}_i)} q_i(\boldsymbol{s}_i) \ln \frac{\Pr(\boldsymbol{u}_i, \boldsymbol{v}_i, \boldsymbol{s}_i \mid \boldsymbol{\theta})}{q_i(\boldsymbol{s}_i)} \\ &\equiv \mathcal{F}(q_1(\boldsymbol{s}_i), \cdots, q_{|\boldsymbol{T}|}(\boldsymbol{s}_{|\boldsymbol{T}|}), \boldsymbol{\theta}) \end{aligned} \tag{3}$$

where $q_i(\boldsymbol{s}_i)$ $(1 \leq i \leq |\boldsymbol{T}|)$ is any probability distribution over the state sequences that generate the training pair $(\boldsymbol{u}_i, \boldsymbol{v}_i)$. In EM-algorithm, we modify the distributions $q_i(\boldsymbol{s}_i)$ and $\boldsymbol{\theta}$ alternately. Hereafter we denote $q_i(\boldsymbol{s}_i)$ simply as $q(\boldsymbol{s}_i)$.

In E-step, we calculate the distributions $q(\boldsymbol{s}_i)$ that maximize the lower bound of Eq. (3) using tth parameters $\boldsymbol{\theta}_t$. Using a Lagrange multiplier λ, we obtain

$$\begin{aligned} &\frac{\partial}{\partial q(\boldsymbol{s}_i)} \left\{ \mathcal{F} - \lambda \left(1 - \sum_{\boldsymbol{s}_i \in Q(\boldsymbol{u}_i, \boldsymbol{v}_i)} q(\boldsymbol{s}_i)\right) \right\} \\ &\quad = \ln \Pr(\boldsymbol{u}_i, \boldsymbol{v}_i, \boldsymbol{s}_i \mid \boldsymbol{\theta}_t) - \ln q(\boldsymbol{s}_i) - c = 0 \end{aligned} \tag{4}$$

where $\mathcal{F}$ is abbreviation of $\mathcal{F}(q(\boldsymbol{s}_i), \cdots, q(\boldsymbol{s}_{|\boldsymbol{T}|}), \boldsymbol{\theta}_t)$. From this equation, we obtain

$$q(\boldsymbol{s}_i) = \frac{1}{c(\boldsymbol{u}_i, \boldsymbol{v}_i)} \Pr(\boldsymbol{u}_i, \boldsymbol{v}_i, \boldsymbol{s}_i \mid \boldsymbol{\theta}_t) \tag{5}$$

where $c(\boldsymbol{u}_i, \boldsymbol{v}_i)$ is normalizing constant given by

$$\begin{aligned} c(\boldsymbol{u}_i, \boldsymbol{v}_i) &= \sum_{\boldsymbol{s}_i \in Q(\boldsymbol{u}_i, \boldsymbol{v}_i)} \Pr(\boldsymbol{u}_i, \boldsymbol{v}_i, \boldsymbol{s}_i \mid \boldsymbol{\theta}_t) \\ &= \Pr(\boldsymbol{u}_i, \boldsymbol{v}_i \mid \boldsymbol{\theta}_t) \ . \end{aligned} \tag{6}$$

In M-step, we calculate the parameters $\boldsymbol{\theta}_{t+1}$ that maximize the lower bound of Eq. (3) by fixing $q(\boldsymbol{s}_i)$. Suppose a pair $(\boldsymbol{u}, \boldsymbol{v})$ of strings are segmented into $\{(\boldsymbol{u}_1, \boldsymbol{v}_1), \cdots, (\boldsymbol{u}_{|\boldsymbol{s}|}, \boldsymbol{v}_{|\boldsymbol{s}|})\}$ by the state transition sequence $\boldsymbol{s}$. Then, let us first define the following functions:

- For a state s and a state sequence $\boldsymbol{s}$, $C_\pi(\boldsymbol{s}, s) = 1$ if the initial state of $\boldsymbol{s}$ is s. Otherwise $C_\pi(\boldsymbol{s}, s) = 0$.
- For states s, r, and a state sequence $\boldsymbol{s}$, $C_\tau(\boldsymbol{s}, s, r)$ denotes the number of state transitions from the state s to r in the sequence $\boldsymbol{s}$.
- Suppose a state sequence $\boldsymbol{s}$ produces a pair $(\boldsymbol{u}, \boldsymbol{v})$ of strings. For a state s and an output pair $(\boldsymbol{a}, \boldsymbol{b})$ of strings at s, $C_o(\boldsymbol{s}, s, \boldsymbol{a}, \boldsymbol{b})$ denotes the number of emissions of the pair $(\boldsymbol{a}, \boldsymbol{b})$ at the state s when the state sequence $\boldsymbol{s}$ produces the pair $(\boldsymbol{u}, \boldsymbol{v})$.

Using these functions, Eq. (1) is written by

$$\begin{aligned}&\ln \Pr(\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{s} \mid \boldsymbol{\theta}_{t+1}) \\ &= \sum_{s \in S} C_\pi(\boldsymbol{s}, s) \ln \pi_{t+1}(s) \cdot \sum_{s,r \in S} C_\tau(\boldsymbol{s}, s) \ln \tau_{t+1}(s, r) \cdot \\ &\quad \sum_{s \in S} \sum_{(\boldsymbol{a}, \boldsymbol{b}) \in O_s} C_o(\boldsymbol{s}, s, \boldsymbol{a}, \boldsymbol{b}) \ln o_{t+1}(s, \boldsymbol{a}, \boldsymbol{b}) \end{aligned} \tag{7}$$

where π_{t+1}, τ_{t+1}, and o_{t+1} denote the $(t+1)$th initial, transition and output probabilities.

For each state s, the following equation gives the optimal parameter

$$\begin{aligned}&\frac{\partial}{\partial \pi_{t+1}(s)} \left\{ \mathcal{F} - \lambda (1 - \sum_{r \in S} \pi_{t+1}(r)) \right\} \\ &= \frac{1}{\pi_{t+1}(s)} \sum_{i=1}^{|\boldsymbol{T}|} \sum_{\boldsymbol{s}_i \in Q(\boldsymbol{u}_i, \boldsymbol{v}_i)} q(\boldsymbol{s}_i) C_\pi(\boldsymbol{s}_i, s) - \lambda = 0 \end{aligned} \tag{8}$$

where λ is Lagrange multiplier. From this equation, we obtain

$$\begin{aligned}\pi_{t+1}(s) &= \frac{\sum_{i=1}^{|\boldsymbol{T}|} \sum_{\boldsymbol{s}_i \in Q(\boldsymbol{u}_i, \boldsymbol{v}_i)} q(\boldsymbol{s}_i) C_\pi(\boldsymbol{s}_i, s)}{\sum_{i=1}^{|\boldsymbol{T}|} \sum_{r \in S} \sum_{\boldsymbol{r}_i \in Q(\boldsymbol{u}_i, \boldsymbol{v}_i)} q(\boldsymbol{r}_i) C_\pi(\boldsymbol{r}_i, r)} \\ &= \frac{\sum_{i=1}^{|\boldsymbol{T}|} \langle C_\pi(\boldsymbol{s}_i, s) \rangle_{q(\boldsymbol{s}_i)}}{\sum_{i=1}^{|\boldsymbol{T}|} \sum_{r \in S} \langle C_\pi(\boldsymbol{s}_i, r) \rangle_{q(\boldsymbol{s}_i)}} \end{aligned} \tag{9}$$

where $\langle \cdot \rangle_P$ denotes the expected value with respect to the probability distribution P, and the probability distribution $q(\boldsymbol{s}_i)$ is given by Eq. (5).

Similarly we obtain the following equations for transition and output probabilities

$$\tau_{t+1}(s, r) = \frac{\sum_{i=1}^{|\boldsymbol{T}|} \langle C_\tau(\boldsymbol{s}_i, s, r) \rangle_{q(\boldsymbol{s}_i)}}{\sum_{i=1}^{|\boldsymbol{T}|} \sum_{t \in S} \langle C_\tau(\boldsymbol{s}_i, s, t) \rangle_{q(\boldsymbol{s}_i)}} \tag{10}$$

$$o_{t+1}(s, \boldsymbol{a}, \boldsymbol{b}) = \frac{\sum_{i=1}^{|\boldsymbol{T}|} \langle C_o(\boldsymbol{s}_i, \boldsymbol{a}, \boldsymbol{b}) \rangle_{q(\boldsymbol{s}_i)}}{\sum_{i=1}^{|\boldsymbol{T}|} \sum_{(\boldsymbol{c}, \boldsymbol{d}) \in O_s} \langle C_o(\boldsymbol{s}_i, \boldsymbol{c}, \boldsymbol{d}) \rangle_{q(\boldsymbol{s}_i)}} \tag{11}$$

Using Eqs. (9), (11), and (11) the following EM algorithm is derived

1. set initial parameters $\boldsymbol{\theta}_0$.
2. repeat until convergence :
   (a) calculate the $(t+1)$th parameters $\boldsymbol{\theta}_{t+1}$ by Eqs. (9), (11), and (11) using tth parameters $\boldsymbol{\theta}_t$.

3.2 Reestimation Algorithm

To update parameters in the EM-algorithm derived in the previous section, we need to calculate the expected values $\langle C_\pi(\boldsymbol{s}_i, s) \rangle_{q(\boldsymbol{s}_i)}$, $\langle C_\tau(\boldsymbol{s}_i, s, r) \rangle_{q(\boldsymbol{s}_i)}$, and $\langle C_o(\boldsymbol{s}_i, \boldsymbol{a}, \boldsymbol{b}) \rangle_{q(\boldsymbol{s}_i)}$ for initial, transition and output probabilities, respectively. Because calculating $q(\boldsymbol{s}_i)$ for all state transitions is computationally prohibited, we introduce forward and backward probabilities just like the parameter estimation of HMM. For a state s, let lo_s and lc_s denote the lengths of the original and converted output symbol of a state s, respectively. For a prefix pair $(\boldsymbol{u}_{:i}, \boldsymbol{v}_{:j})$ of training data and a state s, a forward probability of a DVHMM M, denoted as $\mathcal{F}_{i,j}(\boldsymbol{u}, \boldsymbol{v}, s)$, is defined as the probability that M reaches the state s after producing $(\boldsymbol{u}_{:i}, \boldsymbol{v}_{:j})$. The forward probability is calculated inductively in the following manner:

$$\begin{aligned}&\mathcal{F}_{0,0}(\boldsymbol{u}, \boldsymbol{v}, s) = \pi(s) \\ &\mathcal{F}_{i,j}(\boldsymbol{u}, \boldsymbol{v}, s) = \sum_{r \in S} \mathcal{F}_{\tilde{i},\tilde{j}}(\boldsymbol{u}, \boldsymbol{v}, r)\ \tau(r, s)\ o(s, \boldsymbol{a}, \boldsymbol{b}) \end{aligned} \tag{12}$$

where

$$\begin{aligned}\tilde{i} &= i - lo_r \\ \tilde{j} &= j - lc_r \\ \boldsymbol{a} &= \boldsymbol{u}_{i-\tilde{i}+1:i} \\ \boldsymbol{b} &= \boldsymbol{v}_{j-\tilde{j}+1:j} \end{aligned} \tag{13}$$

hold.

Form the definition of the forward probabilities, we derive the following equation

$$\Pr(\boldsymbol{u}, \boldsymbol{v} \mid \boldsymbol{\theta}) = \sum_{s \in S} \mathcal{F}_{|\boldsymbol{u}|,|\boldsymbol{v}|}(\boldsymbol{u}, \boldsymbol{v}, s) \tag{14}$$

Similarly, for a suffix pair $(\boldsymbol{u}_{i:}, \boldsymbol{v}_{j:})$ of training data and a state s, the backward probability of M, denoted by $\mathcal{B}_{i,j}(\boldsymbol{u}, \boldsymbol{v}, s)$, is defined as the probability that M produces $(\boldsymbol{u}_{i:}, \boldsymbol{v}_{j:})$ from a state q. The backward probability is calculated inductively in the following manner:

$$\begin{aligned}&\mathcal{B}_{|\boldsymbol{u}|+1,|\boldsymbol{v}|+1}(\boldsymbol{u}, \boldsymbol{v}, s) = 1.0 \\ &\mathcal{B}_{i,j}(\boldsymbol{u}, \boldsymbol{v}, s) = \sum_{r \in S} o(s, \boldsymbol{a}, \boldsymbol{b})\ \tau(s, r)\ \mathcal{B}_{\hat{i},\hat{j}}(\boldsymbol{u}, \boldsymbol{v}, r) \end{aligned} \tag{15}$$

where

$$\begin{aligned}\hat{i} &= i + lo_s \\ \hat{j} &= j + lc_s \\ \boldsymbol{a} &= \boldsymbol{u}_{i:i+lo_s-1} \qquad (16) \\ \boldsymbol{b} &= \boldsymbol{v}_{j:j+lc_s-1} \qquad (17) \end{aligned}$$

hold.

For a pair (i,j) of positions of strings and a transition from a s to a state r let us consider the probability $\xi(\boldsymbol{u}, \boldsymbol{v}, i, j, s, r)$ that

- the model reaches the state s after producing the prefices $(\boldsymbol{u}_{:i}, \boldsymbol{v}_{:j})$,

- produces prefices $\boldsymbol{u}_{i:i+lo_s}$ and $\boldsymbol{v}_{j:j+lc_s}$ j
- moves to the state r, and
- produces the remaining suffices by the state transitions starting with r.

This probability is represented with the forward and backward probabilities

$$\begin{aligned}&\xi(\boldsymbol{u}, \boldsymbol{v}, i, j, s, r)\\ &\quad = \mathcal{F}_{i,j}(\boldsymbol{u}, \boldsymbol{v}, s)\ o(s, \boldsymbol{u}_{i:\tilde{i}-1}, \boldsymbol{v}_{j:\tilde{j}-1})\\ &\qquad \tau(s, r)\ \mathcal{B}_{\tilde{i},\tilde{j}}(\boldsymbol{u}, \boldsymbol{v}, r)\end{aligned} \tag{18}$$

where $\tilde{i} = i + lo_s$ and $\tilde{j} = j + lc_s$.

From the definition, the expected value for the initial probability is given by

$$\langle C_\pi(\boldsymbol{s}, s)\rangle_{q(s)} = \pi(s)\ \mathcal{B}_{1,1}(\boldsymbol{u}, \boldsymbol{v}, s)\ . \tag{19}$$

Similarly the expected value for the transition probability is given by

$$\langle C_\tau(\boldsymbol{s}_i, s, r)\rangle_{q(\boldsymbol{s}_i)} = \sum_{i,j} \xi(\boldsymbol{u}, \boldsymbol{v}, i, j, s, r) \tag{20}$$

and, the expected value for the output probability is given by

$$\begin{aligned}&\langle C_o(\boldsymbol{s}_i, \boldsymbol{a}, \boldsymbol{b})\rangle_{q(\boldsymbol{s}_i)}\\ &\quad = \sum_{i,j}\sum_{r\in S} \xi(\boldsymbol{u}, \boldsymbol{v}, i, j, s, r)\delta(\boldsymbol{u}, \boldsymbol{v}, i, j, \boldsymbol{a}, \boldsymbol{b}, s)\end{aligned} \tag{21}$$

where $\delta(\boldsymbol{u}, \boldsymbol{v}, i, j, \boldsymbol{a}, \boldsymbol{b}, s) = 1$ if

$$\begin{aligned}\boldsymbol{a} &= \boldsymbol{u}_{i:lo_s}\\ \boldsymbol{b} &= \boldsymbol{v}_{i:lc_s},\end{aligned}$$

otherwise it is 0.

In calculating the forward and backward probabilities, we must take care of the null state. If a null state s has a null transition to a state r, for every i and j $\mathcal{F}_{i,j}(\boldsymbol{u}, \boldsymbol{v}, s)$ must be calculated before $\mathcal{F}_{i,j}(\boldsymbol{u}, \boldsymbol{v}, r)$ and $\mathcal{B}_{i,j}(\boldsymbol{u}, \boldsymbol{v}, s)$ must be calculated after $\mathcal{B}_{i,j}(\boldsymbol{u}, \boldsymbol{v}, r)$. By this constraint, the DVHMM is not allowed to have a loop consisting of only null states. In order to handle null states, let $\boldsymbol{N}$ be an ordered set of states in S that satisfies the following conditions:

- any null state is located before any non-null state, and
- for any pair of null states s and r, if $s <_N r$ holds, there is no direct transition from a state r to a state s

estimate($\boldsymbol{T}$, M) // EM-algorithm for DVHMM

input: a set $\boldsymbol{T}$ of training pairs of similar strings, DVHMM M
output: a set $\boldsymbol{\theta}$ of parameters
begin
set initial values to $\boldsymbol{\theta}$
until the lower bound (3) converges
  foreach training pair $(\boldsymbol{u}_i, \boldsymbol{v}_i)$
    foreach positions i, j, and a state s
      $F[i][j] \leftarrow \mathcal{F}_{i,j}(\boldsymbol{u}, \boldsymbol{v}, s)$ by Eq. (12)
    foreach positions i, j, and a state s
      $B[i][j] \leftarrow \mathcal{B}_{i,j}(\boldsymbol{u}, \boldsymbol{v}, s)$ by Eq. (15)
    foreach states s, r and strings $\boldsymbol{a}$, $\boldsymbol{b}$
      add $\langle C_\pi(\boldsymbol{s}_i, s)\rangle_{q(\boldsymbol{s}_i)}$ to $E_\pi[s]$ by Eq. (19)
      add $\langle C_\tau(\boldsymbol{s}_i, s, r)\rangle_{q(\boldsymbol{s}_i)}$ to $E_\tau[s][r]$ by Eq. (20)
      add $\langle C_o(\boldsymbol{s}_i, s, \boldsymbol{a}, \boldsymbol{b})\rangle_{q(\boldsymbol{s}_i)}$ to $E_o[s][\boldsymbol{a}][\boldsymbol{b}]$ by Eq. (21)
    end
    update $\boldsymbol{\theta}$ by Eqs. (9), (11), and (11)
  end
end

Fig. 2 EM-algorithm for DVHMM.

where $s <_N r$ means that s is located before r in $\boldsymbol{N}$.

Fig. 2 shows an outline of the parameter estimation procedure. For each training pair $(\boldsymbol{u}, \boldsymbol{v})$ of training data $\boldsymbol{T}$, arrays F and B are used to keep forward and backward probabilities in the dynamic programming algorithm according to Eqs. (12) and (15). On the other hand, arrays E_π, E_τ and E_o are used to keep the sum the numerators and denominators of Eqs. (9), (11), and (11). The outline of the procedure is the same as for the ordinary HMM parameter estimation algorithm. The procedure differs from the ordinary HMM algorithm in its handling of the forward and backward probabilities that are defined for each pair i, j of positions in a training pair $\boldsymbol{u}$ and $\boldsymbol{v}$ as defined in Eqs. (12) and (15), whereas they are defined for each position i in a training string in the ordinary HMM.

4 Conclusion

This paper proposes a string similarity model called a DVHMM and gives a parameter estimation algorithm based on the EM algorithm. Although existing probabilistic models are limited to substitution (1,1), insertion (1,0), and deletion (0,1) errors, the DVHMM can handle error patterns of any pair (i, j) of lengths including substitution, insertion, and deletion.

References

[1] M. Bilenko and R. J. Mooney, "Adaptive Duplicate Detection Using Learnable String Similarity Measures". In *Proc. of 9th Intl. Conf. on Knowledge Discovery and Data Mining (KDD03)*, pp.39–48, 2003.

[2] S. Deligne and F. Bimbot, "Language Modeling by Vari-

able Length Sequences: Theoretical Formulation and Evaluation of Multigrams". In *Proc. of Intl. Conf. on Acoustic, Speech, and Signal Processing*, pp.169–172, 1995.

[3] S. Kahan, T. Pavlidis, and H. S. Baird, "On the recognition of printed characters of any font and size". *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol.9, no.2, pp.274–288, March 1987.

[4] T. Kuhn, H. Niemann, and E. G. Schukat-Talamazzini, "Ergodic Hidden Markov Models and Polygrams for Language Modeling". In *Proc. of Intl. Conf. on Acoustic, Speech, and Signal Processing*, pp.357–360, 1994.

[5] Y. Li, D. Lopresti, and A. Tomkins, "Validation of Document Image Defect Models for Optical Character Recognition". In *Proc. of 3rd Annual Symposium on Document Analysis and Information Retrieval*, pp.137–150, 1994.

[6] A. Myka and U. Guntzer, "Fuzzy Full-Text Searches in OCR Database". In *Proc. of Forum on Research & Technology Advances in Digital Libraries*, pp.87–100, 1995.

[7] G. Navarro, "A guided tour to approximate string matching". *ACM Computing Surveys*, vol.33, no.1, pp.31–88, 2001.

[8] M. Ohta, A. Takasu, and J. Adachi, "Probabilistic Automaton Model for Fuzzy English-text Retrieval". In *Lecture Notes in Computer Science 1923*, pp.35–44, 2000.

[9] L. R. Rabiner, "A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition". *Proceedings of the IEEE*, vol.77, no.2, pp.257–286, 1989.

[10] E. S. Ristad and P. N. Yianilos. Learning string-edit distance, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol.20, no.5, pp.522–532, 1998.

[11] A. Takasu, "Bibliographic Attribute Extraction from Erroneous References Based on a Statistical Model". In *Proc. of 3rd ACM & IEEE Joint Conf. on Digital Libraries*, pp.49–60, 2003.

[12] A. Takasu and K. Aihara, "DVHMM: Variable Length Text Recognition Error Model". In *Proc. of 15th Intl. Conf. on Pattern Recognition*, pp.110–114, 2002.

Atsuhiro TAKASU

Atsuhiro TAKASU received B.E., M.E. and Dr. Eng. from the University of Tokyo in 1984, 1986 and 1989, respectively. He is a professor of National Institute of Informatics, Japan. His research interests are database systems and machine learning. He is a member of ACM, IEEE, IEICE, IPSJ and JSAI.