An HMM-Based Algorithm for Content Ranking and Coherence-Feature Extraction

Chien-Liang Liu, Wen-Hoar Hsaio, Chia-Hoang Lee, and Hsiao-Cheng Chi

***Abstract*—In this paper, we propose an algorithm called coherence hidden Markov model (HMM) to extract coherence features and rank content. Coherence HMM is a variant of HMM and is used to model the stochastic process of essay writing and identify topics as hidden states, given sequenced clauses as observations. This study uses probabilistic latent semantic analysis for parameter estimation of coherence HMM. In coherence-feature extraction, support vector regression (SVR) with surface features and coherence features is used for essay grading. The experimental results indicate that SVR can benefit from coherence features. The adjacent agreement rate and the exact agreement rate are 95.24% and 59.80%, respectively. Moreover, this study submits high-scoring essays to the same experiment and finds that the adjacent agreement rate and exact agreement rate are 98.33% and 64.50%, respectively. In content ranking, we design and implement an intelligent assisted blog writing system based on the coherence-HMM ranking model. Several corpora are employed to help users efficiently compose blog articles. When users finish composing a clause or sentence, the system provides candidate texts for their reference based on current clause or sentence content. The experimental results demonstrate that all participants can benefit from the system and save considerable time on writing articles.**

***Index Terms*—Coherence-feature extraction, hidden Markov model (HMM), input devices and strategies, natural language processing (NLP), predictive content.**

I. Introduction

RECENTLY, essays have become central to a formal education, and exams require good writing. However, while writing is important for a literary education, it is costly for human raters to grade essays. Automated essay scoring (AES) is the ability of computer technology to evaluate and score written prose. AES was first proposed in 1966, and its capability has been proven through application to large-scale essay exams. Companies such as Vantage Learning and Educational Testing Service (ETS) Technologies have published research results demonstrating strong correlations and insignificant differences between AES and human scoring [1]. AES systems are designed to simulate grading by a human rater and are usable only if they can grade as accurately as human raters. Thus, if more features that are able to identify grading criteria are available, AES can increase the accuracy and reliability of essay grading. In this paper, we propose a novel algorithm, a variant of hidden Markov model (HMM) called coherence HMM, to extract coherence features.

Essay writing can be viewed as a temporal process, in which each clause is completed over time. This study employs coherence HMM to model the stochastic process of essay writing and identify topics as the hidden states while providing sequenced clauses as observations. The parameter estimation of HMM relies on expectation and maximization (EM) algorithm [2] to obtain the maximum-likelihood estimate of the parameters. The coherence HMM employs probabilistic latent semantic analysis (LSA) (PLSA) [3], a statistical technique for analyzing co-occurrence data, to estimate parameters. Essentially, PLSA is based on a mixture decomposition derived from a latent class model. Similarly, the maximum-likelihood estimation (MLE) in PLSA employs the EM algorithm for parameter estimation. When the parameters of PLSA are obtained, coherence HMM employs the topic and term-topic distribution information from PLSA for parameter estimation. Each hidden state of coherence HMM corresponds to a PLSA topic. The observed clauses are generated or emitted by these hidden states, where the clause emission probability can be calculated from the topic and term-topic distributions of PLSA. Each article is comprised of numerous clauses, each of which can be transformed into a corresponding topic using *maximum a posteriori* (MAP). Thus, each training article can be transformed into a sequence of topics. The initial state probability and state transition probabilities can be estimated from the collection of topic sequences.

Essay scoring and writing are related to coherence, explaining why we develop a coherence-HMM algorithm to extract coherence features from essays and rank online content. In the ranking model, coherence HMM can rank next clauses or sentences based on the results inferred from observed clauses. Moreover, blog has become a topic of significant recent interest due to the emergence and growth of social networks [4], [5]. Based on the ranking model, we design and implement an intelligent assisted blog writing system to help users compose blog articles. The system employs the Web as a corpus and issues a query to Google to obtain candidate texts. These candidate texts are then ranked according to coherence HMM with user-finished sentences or clauses. Besides text prediction, stylus is used for input. Terms are selected from predictive texts by crossing [6], [7]; that is, the user draws a stroke over one of the prediction results, tracing the desired words in the list.

Manuscript received November 17, 2010; revised June 7, 2011, November 16, 2011, and March 16, 2012; accepted May 15, 2012. Date of publication January 9, 2013; date of current version February 12, 2013. This work was supported in part by the National Science Council under the Grant NSC-100-2221-E-009-129. This paper was recommended by Associate Editor L. C. Jain.

C.-L. Liu, W.-H. Hsaio, and C.-H. Lee are with the Department of Computer Science, National Chiao Tung University, Hsinchu 300, Taiwan (e-mail: clliu@mail.nctu.edu.tw; bass28.cs96g@g2.nctu.edu.tw; chl@cs.nctu.edu.tw).

H.-C. Chi is with Foxconn International Holdings, Ltd., New Taipei City 236, Taiwan (e-mail: fgg1051@gmail.com).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TSMCA.2012.2207104

In coherence-feature extraction, the topic transitions between clauses are viewed as coherence features. Support vector regression (SVR) [8], [9] with surface features and coherence features is used for essay grading. The experimental results demonstrate that coherence features can effectively improve AES accuracy. In the ranking model, the intelligent assisted blog writing system ranks online content to help the user compose a blog article. The experimental results indicate that all participants can benefit from the system and save time on article composition. The feedback shows that the predictive texts provided by the system can also inspire the participants to devise new ideas. The main contributions include the following.

1) Proposing a coherence-HMM model to extract coherence features and rank text content.
2) Employing SVR with surface and coherence features for essay grading. The experimental results indicate that coherence features can improve grading accuracy.
3) Design and implementation of an intelligent assisted blog writing system, which uses stylus input and ranks online content for user reference.

The remainder of this paper is organized as follows. Section II surveys the research on human–computer interaction (HCI) design, assisted writing systems, and AES systems. Section III describes the coherence-HMM model, and Section IV introduces the system design. Section V then presents conducted experiments to evaluate the system design. Finally, Section VI presents conclusions.

II. Related Research

An increasing volume of research is focused on employing computer technology to improve user writing skills. Recently, various online writing-assistance tools have been developed to help users compose articles. Liu *et al.* [10] devised a computer-aided system that helps Chinese users check spelling and grammar errors when writing English. Additionally, Leacock *et al.* [11] designed and implemented a prototype Web-based writing-assistance tool, the Microsoft Research ESL[1] Assistant, for English language learners.

Besides online writing-assistance tools, an AES system, which can reduce the cost of manual grading, provides an environment for users to practice essay writing independently. More systems were developed during the late 1990s, with the the most prominent among them being Intelligent Essay Assessor (IEA) [12], e-rater, and IntelliMetric. IntelliMetric has successfully graded more than 370 000 essays in 2006 for the Analytical Writing Assessment portion of the Graduate Management Admission Test. Essentially, given more features that can identify grading criteria, AES can grade essays more accurately and reliably.

Practically, writing is closely related to coherence, since coherence is the feature of semantic facet of an article characterized as the connectivity and consistency of the whole discourse in semantics. Various coherence theories have been developed [13], [14], and their principles have been applied to many linguistic domains. For instance, numerous questions related to question answering systems are not isolated but, rather, are evolving and related to specific information goals. One method of simplification involves employing discourse information to process context questions and facilitate answer retrieval. To process coherence of discourse, Sun and Chai [15] employed centering theory [13], which describes the use of different linguistic devices to maintain local discourse coherence.

Burstein *et al.* [16] demonstrated that an essay-grading system combining coherence features with novel features related to grammar errors and word usage can significantly improve automated coherence prediction for student essays. Williams [17] noted that readers judge the coherence of a passage by rapidly and easily identifying two things: 1) the topics of individual sentences and clauses and 2) how the topics of the passage constitute a related set of concepts. Inspired by the research of Williams, this study employs topic transitions of clauses to represent coherence features.

Various computer-aided writing tools exist for different topics and fields. Although these tools have diverse forms, they share common purposes, providing users with immediate writing hints for improving article depth and breadth. Predictive text entry is yet another solution, where users can simply select predictions rather than type every word in full [18]–[23]. Moreover, Komatsu *et al.* [20] performed a study to examine the preferences of users of software for writing Japanese and found that people are generally in favor of selecting. Selection is especially preferred for languages comprising numerous strokes like Japanese and Chinese. The aforementioned user study also provides a direction for user interface design.

Text corpus is important in computational linguistics, since it provides a large and structured set of texts for analysis and hypothesis testing, checking occurrences, or validating linguistic rules. The Web can be viewed as a big database, and many corpus-based applications have employed the Web to design corpus-based systems. Liu *et al.* [24] designed and implemented a computer-assisted writing system to help users create love letters from scratch. Huang *et al.* [25] developed a system, which gathered data via Google Blog Search[2] based on the essay content previously read and written by the author. The system then shows the related documents to users as references. People must review all the paragraphs in their entirety, since the system presents these primal paragraphs directly.

III. Coherence-HMM Model

HMM has been widely used in temporal pattern recognition such as speech [26]–[29], handwriting [30], anomaly detection [31], [32], and part-of-speech (POS) tagging [33]. Similarly, essay writing can be viewed as a temporal process, where each clause is completed over time. This work proposes a coherence-HMM model, a variant of HMM, for modeling the stochastic process of essay writing and recognizing topics as hidden states and regards sequenced clauses as observations. In hidden-state modeling, we propose to apply statistical modeling to clause generation. The clause generation process can be formalized as a stochastic process that starts from hidden topics, generates a sequence of clause terms, then proceeds to the next topic, and generates the corresponding clause, until it reaches the end of a document. Intuitively, a generative model can be used

[1]http://research.microsoft.com/en-us/projects/msreslassistant/

[2]Google Blog Search: http://blogsearch.google.com/blogsearch

to model the process. Additionally, it is common for users to have a main topic in their mind when composing an article. This main topic can be further reduced to several subtopics, each associated with a collection of terms. Thus, this study proposes to employ PLSA to model the generation process, where each term in a document is generated using a mixture model and each document comprises a set of latent topics. The parameter estimations of coherence HMM and PLSA are described hereinafter.

A. Notation

The notations that will be used in the following sections are described in this section. Given a set of training documents $\mathcal{D} = \{d_1, \ldots, d_N\}$, where each document d_i is considered to be an ordered list of term events $\langle w_{i,1}, \ldots, w_{i,M}\rangle$, we use $w_{i,j}$ for the term w_j in the document d_i, where w_j is a term in the vocabulary $\mathcal{W} = \langle w_1, \ldots, w_M\rangle$. $\mathcal{W}$ is also the distinct observation symbol set of coherence HMM. The entry value for $w_{i,j}$ is represented as $n(d_i, w_j)$, meaning the number of times that w_j is occurring in d_i. The number of topics is K, so there are K latent variables $z_1, \ldots, z_K$ in the model, and they are also the hidden states of coherence HMM. $P(z_k)$ denotes the prior probability for the latent variable z_k, and a vector P_z is used to stand for all $P(z_k)$. $P(d_i)$ is used to denote the probability that a term occurrence will be observed in a particular document d_i. $P(z_k|d_i)$ represents a document-specific probability distribution over the latent variable space. $P(w_j|z_k)$ is the class-conditional probability of a specific term conditioned on the unobserved class variable z_k, and a matrix Θ is used to stand for all $P(w_j|z_k)$. Each row Θ_k of Θ and its entry Θ_{kj} represent a topic vector z_k and the probability of term w_j generated by topic z_k, respectively.

The coherence-HMM algorithm is a variant of HMM, and the variables used in the model are the same as those used in HMM. HMM models should include hidden states, observation symbols, and a set of parameters $\lambda = (A, B, \pi)$, where π represents initial state distribution vector, A denotes transition probability matrix, and B is emission probability matrix. The coherence HMM employs PLSA for parameter estimation. The hidden state is S, where $S = \{z_1, \ldots, z_K\}$ corresponds to the K topics of PLSA. The observation symbols are clauses, each of which is a subset of $\mathcal{W} = \{w_1, \ldots, w_M\}$, and these terms are the same as the observed terms of PLSA. Furthermore, without loss of generality, a variable k is used to denote state z_k, and a_{ij} is the element of matrix A, which stands for the transition probability from state i to j.

B. PLSA Parameter Estimation

PLSA is a statistical model, and it is also called the aspect model [34]. The aspect model is a latent variable model for co-occurrence data which associates an unobserved class variable $z_k \in \{z_1, \ldots, z_K\}$ with each observation, an observation being the occurrence of a term in a particular document [35]. The latent variables in a document collection can be viewed as unobserved topics of the documents. Topic modeling, discovering hidden "topic" from a collection of documents, has recently been studied by many researchers [35]–[38]. It has been demonstrated to be a reliable method in document retrieval and classification. Essentially, PLSA is based on a mixture decomposition derived from a latent class model. The standard procedure for MLE in latent variable models is the EM algorithm, which includes E-step and M-step. In E-step, the posterior probabilities are computed for the latent variable z based on the current estimates of the parameters. The E-step is

$$P(z_k|d_i, w_j) = \frac{P(w_j|z_k)P(z_k|d_i)}{\sum_{l=1}^{K} P(w_j|z_l)P(z_l|d_i)}. \tag{1}$$

In M-step, the parameters are updated based on the posterior probabilities of the latent variables. The estimate $P(d_i) \propto n(d_i)$ can be carried out independently. By standard calculation, one arrives at the following M-step reestimation equations:

$$P(w_j|z_k) = \frac{\sum_{I=1}^{N} n(d_i, w_j)P(z_k|d_i, w_j)}{\sum_{m=1}^{M}\sum_{I=1}^{N} n(d_i, w_m)P(z_k|d_i, w_m)} \tag{2}$$

$$P(z_k|d_i) = \frac{\sum_{j=1}^{M} n(d_i, w_j)P(z_k|d_i, w_j)}{n(d_i)}. \tag{3}$$

When the iteration process is completed, the system can obtain a term-topic distribution $P(w_j|z_k)$, which is represented as a matrix Θ. Furthermore, we can obtain the prior probability $P(z_k)$ for each latent variable z_k by

$$P(z_k) = \frac{\sum_{I=1}^{N} P(z_k|d_i)}{\sum_{l=1}^{K}\sum_{I=1}^{N} P(z_l|d_i)}. \tag{4}$$

C. Coherence-HMM Parameter Estimation

Although coherence HMM is a variant of HMM, the parameter estimation process in coherence HMM differs from that in HMM. The parameter estimation task in HMM usually involves deriving the maximum-likelihood estimate of the parameters given the set of output sequences using an iterative procedure such as the Baum–Welch algorithm. The coherence HMM employs the results of PLSA for parameter estimation. As described earlier, the observation symbols are clauses, each comprising a sequence of terms. The number of term combinations is generally enormous, making it infeasible to calculate emission probabilities for all possible clauses in advance. This study employs the topic distribution P_z and term-topic distribution Θ of PLSA to calculate clause emission probabilities. Each hidden state of coherence HMM corresponds to a topic of PLSA. These hidden states generate or emit the observed clauses. For each clause, this study employs the assumption used in naive Bayes. The presence of a specific term of a topic is unrelated to the presence of other terms. Restated, the terms in a clause are independent given the topic of the clause. Obviously, the more terms the clause contains, the less chance it will be generated. Hence, this study employs geometric mean to normalize clause emission probability. Given a state k, the emission probability $b_k(c)$ of a clause c containing $|c|$ terms $(x_1, \ldots, x_{|c|}$, where $x_i \in \mathcal{W})$ is as follows:

$$b_k(c) = P(z_k)\left(\prod_{i=1}^{|c|} P(x_i|z_k)\right)^{\frac{1}{|c|}}. \tag{5}$$

Using term-topic open matrix Θ and P_z, it is possible to determine the most likely topic generating each clause. To model state transition, it is assumed that each clause is generated by a topic. The hidden topic/state of the clause can be obtained from the topic distribution P_z, term-topic distribution Θ, and MAP as shown in (6), where $T(c)$ represents the topic index of clause c and x_i is a term of clause c

$$T(c) = \arg\max_k P(z_k) \left(\prod_{I=1}^{|c|} P(x_i|z_k) \right). \tag{6}$$

Each article in the training data can then be represented as a sequence of clauses which can be transformed into corresponding topics. Restated, the training documents can be transformed into a series of topics. Then, we introduce two counting variables τ and M to keep track of state frequency information. The variable τ_k is used to represent the frequency with state k as the initial state; while M_{nm} denotes the frequency of state transition from state n to state m. Finally, the initial state probability vector π and state transition probability matrix A can then be calculated using MLE. Algorithm 1 illustrates the coherence-HMM parameter estimation algorithm.

Algorithm 1: Coherence-HMM Parameter Estimation Algorithm

Input: The number of topics K, the corpus E, and the PLSA parameters Θ and P_z
Output: The coherence-HMM parameters A and π.
begin
  Reset E topic-transition matrix $M_{k\times k} \longleftarrow 0$, where $1 \le k \le K$
  Reset the count of initial state $\tau_k \longleftarrow 0$, where $1 \le k \le K$
  foreach *document* $E_i \in E$ **do**
    $L \longleftarrow$ the number of clauses in document E_i
    for $l = 1$ **to** L **do**
      $m \longleftarrow$ Estimate the latent topic index $T(C_l)$ for the clause C_l based on (6) with the estimated PLSA parameters Θ and P_z
      if $l \neq 1$ **then**
        $n \longleftarrow$ Obtain the latent topic index for the clause C_{l-1} in essay E_i
        $M_{nm} \longleftarrow M_{nm} + 1$
      else
        $\tau_m \longleftarrow \tau_m + 1$
      end
    end
  end
  Use MLE to estimate the state transition probability matrix A and the initial state probability vector π based on M and τ, respectively
end

D. Ranking Model

On completion of the aforementioned process, the parameters for coherence HMM are obtained. In the ranking model, the observation scope is a sentence, while a clause is the basic element. When the user finishes the clause at time t, the system attempts to predict the most likely clause at time $t+1$. The ranking mechanism proposed in this study ranks the candidate texts obtained from the Web or the corpus. The user then selects the best candidate text from the list. Each candidate text is considered as the clause at time $t+1$, and an observation sequence is constructed using the candidate text along with previous clauses. For each observation sequence, the aim is to calculate the posterior marginals of all hidden-state variables given the observation sequence. Generally, this problem can be resolved by using dynamic programming to efficiently calculate the values. By convention [27], given model λ, the forward variable $\alpha_t(k)$ is defined as follows:

$$\alpha_t(k) = P(c_1, \ldots, c_t, s_t = k|\lambda) \tag{7}$$

namely, the probability of the partial observation sequence $c_1, \ldots, c_t$ and state k at time t. Based on the emission probability as shown in (5), $\alpha_t(k)$ can be solved inductively:

1) initialization

$$\alpha_1(k) = \pi_k b_k(c_1), \qquad 1 \le k \le K \tag{8}$$

2) induction

$$\alpha_{t+1}(k) = \left[\sum_{j=1}^{K} \alpha_t(j) a_{jk} \right] b_k(c_{t+1}), \qquad 1 \le j \le K. \tag{9}$$

Algorithm 2 shows the ranking algorithm. By evaluating the marginal probability g_l for each candidate text/clause Y_l, the candidate texts/clauses obtained from the Web can be ranked.

Algorithm 2: Recommended Text Ranking Algorithm

Input: The number of topics K, the PLSA parameters Θ and P_z, the coherence-HMM parameters A and π, a sequence of observations $c_1, \ldots, c_t$ (including previous clauses $c_1, \ldots, c_{t-1}$ before time t and current clause c_t at time t), and a list of candidate texts Y with size L at time $t+1$
Output: An index list $\hat{l}$, which represents indices of the most possible texts/clauses at time $t+1$.
begin
  for $k = 1$**to**K**do**
    $\alpha_t(k) \longleftarrow$ Employ the coherence-HMM parameters A, π, (5) and (7) to compute the production probability.
  for
  for$l = 1$**to**L**do**
    $m \longleftarrow$ Estimate the latent topic index $T(Y_l)$ for the clause Y_l based on (6) with the PLSA parameters Θ and P_z.
    $g_l \longleftarrow \max_{1\le k\le K} \{\alpha_k(t) \times a_{km} \times b_m(Y_l)\}$
  end
  $\hat{l} \longleftarrow$ Sort g_l in descending order and get their corresponding indices.
end

E. Coherence-Feature Extraction

Besides content ranking, this section describes how to employ coherence HMM to extract coherence features. As mentioned earlier, coherence should consider the connectivity and consistency from the whole discourse in semantics, explaining why this study employs a topic to represent a clause and models the connectivity using topic transitions. The coherence features can then be extracted from the transition frequencies of topics. For instance, given K topics, a transition table R with dimensions $K \times K$ can be constructed. Each entry R_{nm} denotes the transition frequency from topic n to topic m. This study employs the essay data set to evaluate coherence-feature extraction. Algorithm 3 presents the coherence-feature-extraction algorithm. In Algorithm 3, the transition table for each article is transformed into a vector by row wise, which is called topic-transition vector. The transition table can be considered as the coherence features of the article. Moreover, the topic-transition matrix F can be obtained by using all extracted topic-transition vectors, where F_{ij} represents the times of the jth topic-transitive feature occurring in the ith article.

Algorithm 3: Coherence-Feature-Extraction Algorithm

Input: A set of essays E, the number of topics K, and the PLSA parameters Θ and P_z
Output: Topic-transitive feature matrix F for the set E, where F_{ij} represents the times that the jth topic-transitive feature occurs in essay E_i

```
1  begin
2    for each essay E_i ∈ E do
3      Reset E_i topic-transition matrix
4      R_{k×k} ⟵ 0, where 1 ≤ k ≤ K
5      L ⟵ the number of clauses in essay E_i
6      for l = 1 to L do
7        m ⟵ Estimate the latent topic index T(C_l)
         for the clause C_l in essay E_i based on
         (6) with the estimated PLSA
         parameters Θ and P_z
8        if l ≠ 1 then
9          n ⟵ Obtain the latent topic index for the clause
           C_{l−1} in essay E_i
10           R_{nm} ⟵ R_{nm} + 1
11         end
12       end
13       j ⟵ 1
14       for k = 1 to K do
15         for k' = 1 to K do
16           F_{ij} ⟵ R_{kk'}
17           j ⟵ j + 1
18         end
19       end
20   end
21 end
```

IV. System Design

This section describes an intelligent assisted blog writing system, which employs coherence HMM to rank content obtained from Google. Fig. 1 shows the system flow, which includes handwriting recognition, phrase prediction, query string generation, candidate text retrieval, and recommended text ranking. The assisted blog writing system focuses on assisting writers. The study uses stylus input and thus can be used by those unfamiliar with computers. The system can be deployed on a tablet PC or a graphics tablet device. When the users begin to write, the handwriting recognizer receives and interprets intelligible handwritten input from their strokes. The candidate terms are listed based on the recognition results. After the user selects a candidate term, phrase recommendation functionality lists possible candidate phrases starting from the candidate term for their reference. When the users finish a clause or sentence, the system automatically issues a query to Google search engine to retrieve candidate texts that may be of interest. These candidate texts are ranked and presented using the proposed model. The users can either choose the best clause from the list or ignore the list. The whole process is described hereinafter.

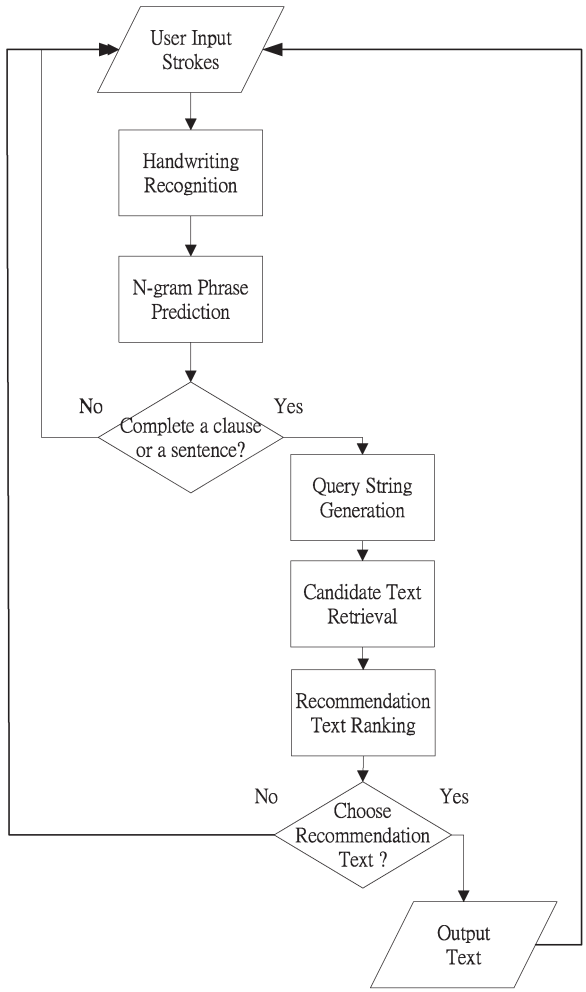

Fig. 1. System flow.

A. Input Interface Design

Fig. 2 shows the basic system interface, which includes areas for handwriting input, handwriting recognition results, clause segmentation, and history. The user uses a stylus to write words in the input area, and the handwriting recognition area then lists recognition results. The word selected by the user appears in

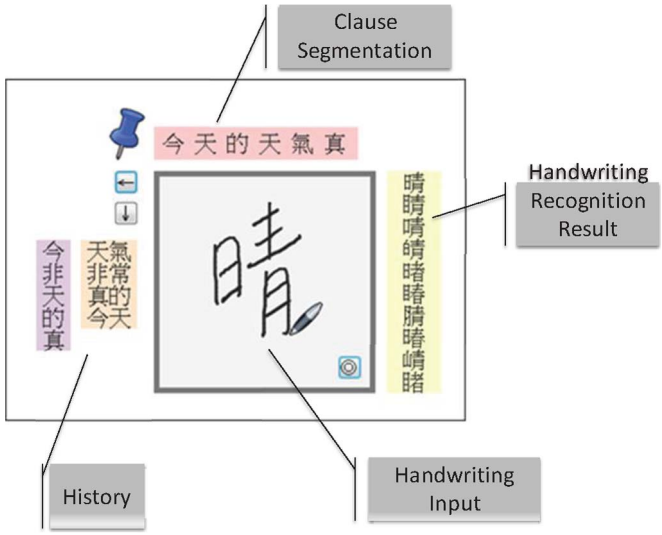

Fig. 2. System interface.

the clause segmentation area. Moreover, the system lists all the candidate phrases subsequent to the word selected by the user and allows the user to directly choose a candidate phrase. The history area lists the words the user has previously used. If the area contains their desired word, users can choose from these words directly, eliminating the need for further handwritten input.

B. Query String Generation

Besides phrase prediction, the predictive texts are obtained from Google, so the query string is important in candidate clause quality. In query string composition, keyword extraction is essential to candidate text quality. To provide content suitable for user requirements, the system refers to the content finished by users and then extracts the keywords to compose a query. The punctuation indicates the end of a clause or sentence. This study employs the position of a term and its POS tag to determine its suitability as a keyword.

According to our analysis, nouns and verbs generally provide more information than other types of words. Chandra *et al.* [39] also used nouns and verbs to highlight sentence semantics, since extractive summary generally requires informative sentences. Thus, only terms with verb and noun POS tags are taken as candidate keywords. Besides POS tags, term location is also considered. Terms close to the end of a clause tend to be more information rich in Chinese language. Predictive text should be connected with the current clause or sentence, and thus, the system assigns higher weightings to terms close to the end of a clause. The location score of each term is determined by its position index divided by the number of terms in the clause.

After completing the aforementioned POS tag filtering and location score computation processes, the system can determine keywords based on these two criteria. A maximum of three keywords are extracted from each clause to create a query string. If fewer than three terms are left, all terms are used. Otherwise, the three terms with the highest scores are used. In

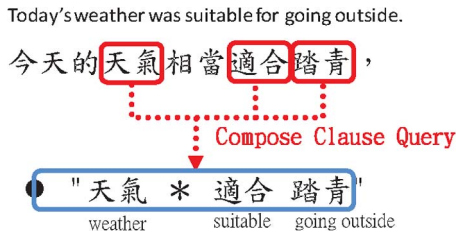

Fig. 3. Clause query compose example.

query string composition, different approaches are employed in clause-end and sentence-end conditions.

1) Clause-end query

   Clause ending with a comma indicates that the user has not yet finished a sentence. The user may continue to describe the content, which is related to the current clause. The system thus extracts the keywords from the current clause using the aforementioned mechanism. These keywords are then presented in the order of their positions. A wildcard is used to represent other terms existing between pairs of keywords. Finally, the query string is surrounded by quotation marks to restrict Google to searching the documents. Fig. 3 shows a simple example where "weather," "suitable," and "going outside" are keywords, and a term exists between "weather" and "suitable." The query string for this example is "weather $*$ suitable going outside."

2) Sentence-end query

   When a clause ends with a semicolon, period, question mark, or exclamation mark, the user has finished their sentence. Unlike a clause-end query, a sentence-end query considers all the keywords in the current sentence. Each sentence can be segmented into several clauses, each of which can be searched for further keywords using the aforementioned mechanism. The query string is not surrounded by quotation marks, since excessive constraints result in few Google matches. Fig. 4 shows a sentence query example, where the keywords in the current sentence are all used and are not surrounded by quotation marks.

C. Candidate Text Retrieval

Currently, given a query, most search engines provide brief excerpts of text under individual search results to help users identify useful links. Google does this through a mechanism called snippet. This study considers the content of each snippet as a paragraph, which can be further segmented into several sentences.

Basically, the search results are ranked by Google, and the system retrieves the top N search results (N is 50 in the system) as the data source for candidate texts. Each snippet obtained from Google can be further segmented into clauses or sentences based on punctuation marks. Clearly, the number of clauses or sentences is enormous, and not all the texts can be applied to the writing contexts of users.

Consequently, these candidate texts are further processed using a filtering mechanism. The filtering mechanism is based

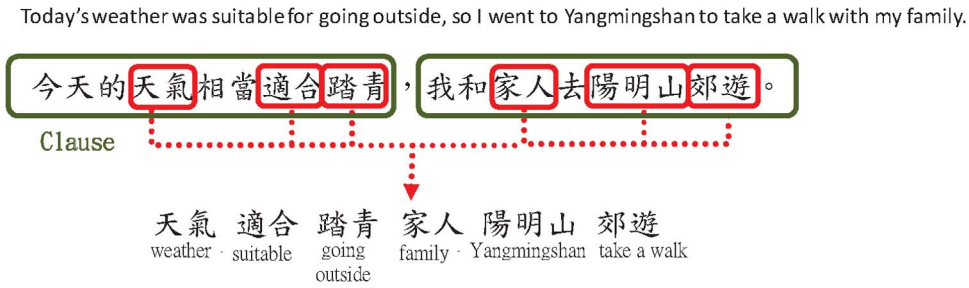

Fig. 4. Sentence query compose example.

on two factors: query term occurrence and sentence continuity. Only clauses containing query terms become candidate texts. However, if all the query terms appear in a specific clause of a snippet, only the next clause is retrieved as the candidate text. This arrangement is used because the appearance of all the query terms in the candidate clause indicates that the clause closely resembles the current user clause. If the query terms only partially appear in the clause, the clause selection is based on the number of matching terms. After completing the aforementioned scoring process, the system can obtain several candidate texts.

D. Ranking of Recommended Texts

After completing the aforementioned candidate text retrieval process, the system can obtain several candidate texts. These texts are ranked using the aforementioned ranking model. Each candidate text is considered as an observation at time $t+1$. Since the candidate text is obtained using the keywords of current clauses or sentences, the observation sequence of coherence HMM only considers current or previous sentences. If the user finishes a clause, the observation sequence only considers the clauses within the same sentence. On the other hand, if the user finishes a sentence, the observation sequence includes clauses in the previous sentence. Fig. 5 shows the system screen shot. These recommended texts are ranked using the model described earlier. The user interface employs a crossing interface design [7], [40], [41]. The user can use the stylus to select terms from different recommended texts and combine them to create a new clause as shown in Fig. 5.

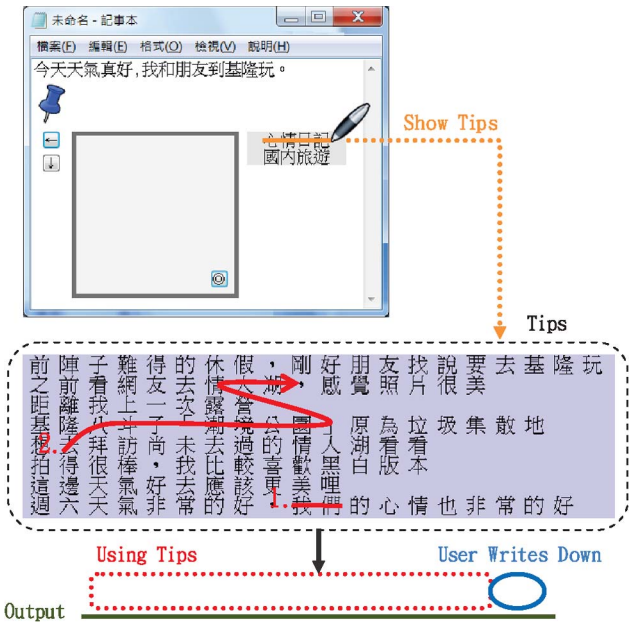

Fig. 5. Recommended text screen shot.

V. Experiments

A. Data Corpus

The predictive text includes two parts, namely, phrase prediction and clause prediction. Both these forms of prediction involve corpus-based approaches. In phrase prediction, the system uses phrases and their frequency information obtained from the libtabe project,[3] which offers a large and free-access database of Chinese words. The database contains approximately 130 000 entries, each including phrase and frequency. The frequency field indicates the statistical usage of the phrase, giving base for ranking of the candidates. When the user finishes typing a word, the system lists possible candidate phrases starting with that word. Furthermore, the system updates the frequency information based on user usage, so the candidate phrase ranking is personalized. In clause prediction, coherence HMM is used to rank the content obtained from Google. In practice, the search engine is not limited to Google, and other search engines such as Yahoo! Search and Microsoft Bing can be used for content retrieval. In system training, we gathered 2000 blog articles from Sina[4] to train the coherence-HMM model.

In the feature-extraction experiment, we employed essays for performance evaluation. The data set comprises essays written by junior high school students from different schools on the subject "If I were a teacher." The essay grades are divided into six performance levels, ranging from one to six, and the various levels contained 200, 200, 199, 200, 200, and 200 essays, respectively. Each essay is graded by two raters, and the final score is obtained by rounding off the average of the two scores.

[3]libtabe open source project: http://sourceforge.net/projects/libtabe/

[4]Sina blog service provider: http://blog.sina.com.tw/

B. Intelligent Assisted Blog Writing System

1) Experiment Environment: The system was implemented on Microsoft C#.NET platform, with a Microsoft Tablet PC Platform Software Development Kit (SDK) used for handwriting recognition. Additionally, the system was also deployed on a PC with a Wacom Intuos2 graphics tablet[5] and a Fujitsu LifeBook T4220 tablet PC.[6] Both systems worked well. The experiments presented hereinafter were conducted on a PC with a graphics tablet. The experiments were intended to evaluate the system and obtain participant feedback for further improvements. This kind of evaluation is frequently used to evaluate HCI systems. HCI systems such as speech pen [21] and audio notebook [42] employed similar methods of evaluation.

We invited 30 people, ranging in age from 12 to 43 years old, to experience and evaluate the system. The participants included an elementary school student, a junior high school student, and a university student majoring in social science, with the remainder being computer science graduate students. The system and its usage were explained to all participants before they started to use it. Since the input device is a stylus, the participants can manipulate the system effortlessly following the introduction.

Currently, the system only covers five categories, namely travel, sport, mood, food, and movie. Each user is randomly assigned two categories and asked to write blog articles on related topics. The time that users spend writing is then recorded. For each assigned category, we asked each participant to compose two articles, one with assistance from the system and one without. Moreover, each article should contain over 200 words. To increase the objectivity of the system experiment, the time period between two article compositions within the same category should be extended to a week or more. Additionally, the two articles should have different subjects to prevent users from writing similar content.

2) Writing Time Evaluation: The first experiment focused on the average time users spent writing. Naturally, the time required for people to finish an article varies with the number of the words written and the topic. Consequently, the average time users spent writing a word on different topics is used for performance evaluation. One of the articles was completed using the proposed system, while the other one was completed unassisted. Fig. 6 shows the results, with the black bar charts representing the results without system assistance and the gray ones representing those with system assistance. In Fig. 6, horizontal axis represents topic category, and vertical axis represents the average time for writing a word. The experimental results show that all participants required less time to write a word when using the proposed system.

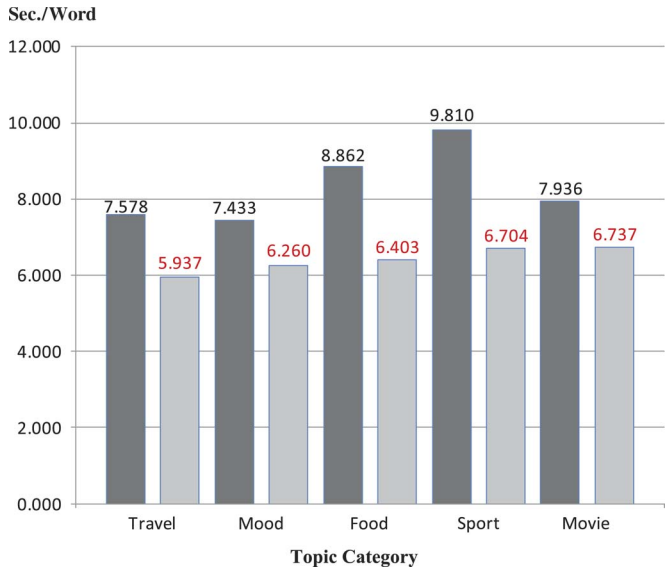

Fig. 6. Experiment result for different categories.

C. AES Evaluation

Traditionally, techniques for detecting similarity between long texts (documents) have focused on analyzing shared words [43]. In natural language processing (NLP) and information retrieval (IR), the bag-of-words model uses an unordered collection of words to represent a text, disregarding grammar and even word order. Restated, each term in the text contributes to a feature of the document. Each distinct term w_i in the document represents a feature, and a feature vector can represent each document. The aforementioned features are called surface features, and a total of 15 452 surface features exist in the essay data set, based only on considering term frequency information. Algorithms 1 and 3 are used for coherence-feature extraction. The number of coherence features is a parameter. We have used cross-validation technique to conduct experiments with parameters varying from 25, 100, 400, 900, and 1600 to 2500. The experimental results indicated that the system with 25 coherence features (i.e., five topics in an essay) outperforms the other ones. Generally speaking, a good essay should not contain too many topics because the main theme will get blurred.

To verify the capability of the extracted coherence features for grading essays, we combine the corresponding surface features and coherence features with different weight ratios to represent each document. The proposed system employs fivefold cross-validation to conduct experiments and presents the results as an average. SVR with surface features is viewed as a baseline experiment. Furthermore, for each experiment, this work evaluates the various performances of the essays and high-scoring essays. The high-scoring essays are those graded four, five, and six. Table I lists the experimental results. The exact agreement is when two or more raters grade an essay identically. On the other hand, adjacent agreement requires two or more raters to assign a score within one scale point of each other. This study employed LIBSVM [44] to conduct experiments, and the kernel function of SVR is the radial basis function.

D. Discussion

The results of the first experiment show that participants write articles faster using the system. One purpose of the system is to help users compose blog articles more efficiently.

[5]Wacom Web site: http://www.wacom.com
[6]Fujitsu Web site: http://www.fujitsu.com

TABLE I
GRADING RESULT USING SVR WITH FEATURE COMBINATION

	All Essays		High-Scoring Essays	
Weight Ratio(Surface : Coherence)	Adjacent Rate	Exact Rate	Adjacent Rate	Exact Rate
0.0 : 1.0	93.74%	53.47%	97.17%	55.83%
0.1 : 0.9	94.74%	56.72%	97.17%	57.50%
0.2 : 0.8	95.57%	58.80%	98.50%	61.33%
0.3 : 0.7	95.82%	59.21%	98.83%	61.33%
0.4 : 0.6	95.57%	58.81%	98.17%	61.00%
0.5 : 0.5	95.00%	59.48%	98.67%	62.67%
0.6 : 0.4	95.24%	59.80%	98.33%	64.50%
0.7 : 0.3	95.66%	58.47%	98.50%	62.17%
0.8 : 0.2	95.49%	59.81%	98.17%	64.33%
0.9 : 0.1	95.41%	57.97%	97.83%	60.83%
1.0 : 0.0	94.83%	56.30%	97.33%	59.00%

The assisted blog writing system employs several corpora to accelerate writing. The analysis and processing of various types of corpora are also the subject of much work in computational linguistics, speech recognition, and machine translation. This paper demonstrates that the corpus-based approach can enhance the assisted blog writing system. Besides the corpora, the incorporation of crossing techniques in the design of user interface also helps users compose desired content. Practically, intellectual property should be considered, since users may use extensive content from the same Web site. To resolve this problem, a Web site weighting mechanism can be enforced in the system design to avoid taking most content from specific Web sites. The source URL information for snippets is available when issuing queries to Google. If users use content from specific Web sites, the system can reduce the weightings of those sites to avoid the system repeatedly providing content coming from the same Web site.

Fig. 6 shows that participants can achieve roughly 30% time savings when writing article on the food and sport categories. This is due to the fact that most blog articles in the food and sport categories describe appetizing food, sports activities, and the health benefits of sport. Many Internet blogs in the two categories appear to share similar content, and thus, the system can provide more accurate predictions.

In contrast, people have different life experiences. Thus, the content of the mood category varies significantly, but the system still helps participants achieve time savings of approximately 20%. As for the movie and travel categories, most articles deal with reviews and traveling experiences, respectively, which involve highly subjective personal feelings and opinions. Consequently, participants cannot benefit significantly from the system but can still save time in composing blog articles.

The participants were asked to provide feedback on their experiences of the system. The feedback shows that the elementary school student and junior high school student were interested in the recommended texts provided by the system and enjoyed using these texts to compose new sentences. Additionally, most participants thought that the recommended texts provided by the system can sometimes inspire them to develop new ideas. We also found that most participants could write Chinese words using the input method but forgot how to write them by hand. One reason for this phenomenon is that people are increasingly accustomed to using computers for writing. However, despite the popularity of computers, it remains important for people to be able to write an article manually.

The second experiment assesses the effect of coherence features on the essay-grading application. Table I shows the adjacent agreement rate and exact agreement rate under different weight ratios between surface and coherence features. The system performs best when the weight ratio between surface and coherence features is "0.6:0.4" and achieves adjacent rate of 95.24% and agreement rate of 59.80%. The experimental results indicate that combining surface features and coherence features can generally improve system performance. Basically, the surface features only consider words, but the word-level features cannot capture the latent semantic information of essays. IEA [12] adopted LSA [45] to analyze essay semantics. The main advantage of this approach is that LSA captures transitivity relations and collocation effects among vocabulary terms and thus can accurately judge the semantic relatedness of two documents regardless of their vocabulary overlap [46]. This study employs PLSA, which stems from a statistical view of LSA, to estimate the topics behind the essay clauses and employs the transitivity relation among clauses to represent coherence features. The experimental results show that the coherence features can capture latent semantic information of essays.

High-scoring essays may involve carefully selected creative expressions to convey the thoughts of the writer, making them more difficult for AES systems to grade high-scoring essays [47]. If AES systems can extract semantic features of essays and apply the features to classification models, it is likely for the systems to grade high-scoring essays more accurately. Thus, we conduct further experiments on high-scoring essays to determine whether their scoring can benefit from coherence features. Table I shows that the exact agreement rate is 64.50% and the adjacent agreement rate is 98.33%, when the weight ratio between surface and coherence features is "0.6:0.4." Consequently, the combination of surface and coherence features can help an AES system grade high-scoring essays more accurately.

VI. CONCLUSION

We have proposed an algorithm called coherence HMM to extract coherence features and rank content. Central to coherence HMM is the topic-based representation of clause, which, we argue, captures important patterns of clause transitions. We view the coherence extraction process as a learning task and

show that the proposed algorithm is well suited in essay-scoring tasks. The experimental results indicate that the extracted coherence features can help an AES system grade essays more accurately. Basically, the coherence features can be considered as semantic features, which may discover latent semantic information of an article. Many computational linguistic problems such as text summarization, readability assessment, and machine translation may use the proposed approach to obtain coherence features. Additionally, we design and implement an intelligent assisted blog writing system based on the coherence-HMM ranking model, which ranks the content obtained from search engines and provides candidate texts for user reference. This paper demonstrates that the corpus-based approach can enhance the assisted blog writing system. Besides the corpora, the incorporation of crossing techniques in the design of interface also helps users compose desired content. The feedback shows that the predictive texts provided by the system can sometimes inspire the participants to devise new ideas. The experimental results indicate that all participants can benefit from the system and can save significant time on writing articles.

REFERENCES

[1] J. Wang and M. S. Brown, "Automated essay scoring versus human scoring: A comparative study," *J. Technol. Learn. Assess.*, vol. 6, no. 2, p. 29, Oct. 2007.

[2] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum likelihood from incomplete data via the EM algorithm," *J. Roy. Stat. Soc., Ser. B*, vol. 39, no. 1, pp. 1–38, 1977.

[3] T. Hofmann, "Probabilistic latent semantic analysis," in *Proc. UAI*, 1999, pp. 289–296.

[4] S.-H. Lim, S.-W. Kim, S. Park, and J. H. Lee, "Determining content power users in a blog network: An approach and its applications," *IEEE Trans. Syst., Man, Cybern. A, Syst., Humans*, vol. 41, no. 5, pp. 853–862, Sep. 2011.

[5] A. Sun and M. Hu, "Query-guided event detection from news and blog streams," *IEEE Trans. Syst., Man, Cybern. A, Syst., Humans*, vol. 41, no. 5, pp. 834–839, Sep. 2011.

[6] J. Accot and S. Zhai, "More than dotting the i's—Foundations for crossing-based interfaces," in *Proc. SIGCHI CHI*, 2002, pp. 73–80.

[7] G. Apitz and F. Guimbretière, "CrossY: A crossing-based drawing application," in *Proc. 17th Annu. ACM Symp. UIST*, 2004, pp. 3–12.

[8] V. N. Vapnik, *The Nature of Statistical Learning Theory*. New York: Springer-Verlag, 1995.

[9] A. J. Smola and B. Schölkopf, "A tutorial on support vector regression," *Stat. Comput.*, vol. 14, no. 3, pp. 199–222, Aug. 2004.

[10] T. Liu, M. Zhou, J. Gao, E. Xun, and C. Huang, "PENS: A machine-aided English writing system for Chinese users," in *Proc. 38th Annu. Meeting ACL*, 2000, pp. 529–536.

[11] C. Leacock, M. Gamon, and C. Brockett, "User input and interactions on microsoft research ESL assistant," in *Proc. 4th Workshop EdAppsNLP*, 2009, pp. 73–81.

[12] T. Landauer and S. Dumais, "A solution to Plato's problem: The latent semantic analysis theory of acquisition, induction, and representation of knowledge," *Psychol. Rev.*, vol. 104, no. 2, pp. 211–240, Apr. 1997.

[13] B. J. Grosz, S. Weinstein, and A. K. Joshi, "Centering: A framework for modeling the local coherence of discourse," *Comput. Linguist.*, vol. 21, no. 2, pp. 203–225, Jun. 1995.

[14] R. Barzilay and M. Lapata, "Modeling local coherence: An entity-based approach," *Comput. Linguist.*, vol. 34, no. 1, pp. 1–34, Mar. 2008.

[15] M. Sun and J. Y. Chai, "Towards intelligent QA interfaces: Discourse processing for context questions," in *Proc. 11th Int. Conf. IUI*, 2006, pp. 163–170.

[16] J. Burstein, J. Tetreault, and S. Andreyev, "Using entity-based features to model coherence in student essays," in *Proc. Annu. Conf. North Amer. Chapter Assoc. Comput. Linguist. HLT*, 2010, pp. 681–684.

[17] J. M. Williams, *Style: Ten Lessons in Clarity and Grace*. White Plains, NY: Longman, 1997.

[18] J. J. Darragh, I. H. Witten, and M. L. James, "The reactive keyboard: A predictive typing aid," *Computer*, vol. 23, no. 11, pp. 41–49, Nov. 1990.

[19] T. Masui, "An efficient text input method for pen-based computers," in *Proc. SIGCHI CHI*, 1998, pp. 328–335.

[20] H. Komatsu, S. Takabayashi, and T. Masui, "Corpus-based predictive text input," in *Proc. Int. Conf. AMT*, 2005, pp. 75–80.

[21] K. Kurihara, M. Goto, J. Ogata, and T. Igarashi, "Speech pen: Predictive handwriting based on ambient multimodal recognition," in *Proc. SIGCHI CHI*, 2006, pp. 851–860.

[22] K. Tanaka-ishii, "Word-based predictive text entry using adaptive language models," *Nat. Lang. Eng.*, vol. 13, no. 1, pp. 51–74, Mar. 2007.

[23] Y. Liu and K. J. Räihä, "Predicting Chinese text entry speeds on mobile phones," in *Proc. 28th Int. Conf. CHI*, 2010, pp. 2183–2192.

[24] C.-L. Liu, C.-H. Lee, S.-H. Yu, and C.-W. Chen, "Computer assisted writing system," *Expert Syst. Appl.*, vol. 38, no. 1, pp. 804–811, Jan. 2011.

[25] T.-C. Huang, S.-C. Cheng, and Y.-M. Huang, "A blog article recommendation generating mechanism using an SBACPSO algorithm," *Expert Syst. Appl.*, vol. 36, no. 7, pp. 10 388–10 396, Sep. 2009.

[26] J. Baker, "The Dragon system—An overview," *IEEE Trans. Acoust., Speech, Signal Process.*, vol. ASSP-23, no. 1, pp. 24–29, Feb. 1975.

[27] L. R. Rabiner, "A tutorial on hidden Markov models and selected applications in speech recognition," *Proc. IEEE*, vol. 77, no. 2, pp. 257–286, Feb. 1989.

[28] B. H. Juang and L. R. Rabiner, "Hidden Markov models for speech recognition," *Technometrics*, vol. 33, no. 3, pp. 251–272, Aug. 1991.

[29] W. Walker, P. Lamere, P. Kwok, B. Raj, R. Singh, E. Gouvea, P. Wolf, and J. Woelfel, "Sphinx-4: A flexible open source framework for speech recognition," Sun Microsystems, Inc., Mountain View, CA, Tech. Rep., 2004.

[30] J. Hu and M. K. Brown, "HMM based online handwriting recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 18, no. 10, pp. 1039–1045, Oct. 1996.

[31] S. Singh, H. Tu, W. Donat, K. Pattipati, and P. Willett, "Anomaly detection via feature-aided tracking and hidden Markov models," *IEEE Trans. Syst., Man, Cybern. A, Syst., Humans*, vol. 39, no. 1, pp. 144–159, Jan. 2009.

[32] W. An, C. Park, X. Han, K. R. Pattipati, D. L. Kleinman, and W. G. Kemple, "Hidden Markov model and auction-based formulations of sensor coordination mechanisms in dynamic task environments," *IEEE Trans. Syst., Man, Cybern. A, Syst., Humans*, vol. 41, no. 6, pp. 1092–1106, Nov. 2011.

[33] S. M. Thede and M. P. Harper, "A second-order hidden Markov model for part-of-speech tagging," in *Proc. 37th Annu. Meeting ACL*, 1999, pp. 175–182.

[34] T. Hofmann, J. Puzicha, and M. I. Jordan, "Learning from dyadic data," in *Proc. Conf. Adv. Neural Inf. Process. Syst. II*, 1999, pp. 466–472.

[35] T. Hofmann, "Unsupervised learning by probabilistic latent semantic analysis," *Mach. Learn.*, vol. 42, no. 1/2, pp. 177–196, Jan./Feb. 2001.

[36] D. M. Blei, A. Y. Ng, and M. I. Jordan, "Latent Dirichlet allocation," *J. Mach. Learn. Res.*, vol. 3, pp. 993–1022, Mar. 2003.

[37] Z. Zhang, Q. Li, and D. Zeng, "Mining evolutionary topic patterns in community question answering systems," *IEEE Trans. Syst., Man, Cybern. A, Syst., Humans*, vol. 41, no. 5, pp. 828–833, Sep. 2011.

[38] C.-L. Liu, W.-H. Hsaio, C.-H. Lee, G.-C. Lu, and E. Jou, "Movie rating and review summarization in mobile environment," *IEEE Trans. Syst., Man, Cybern. C, Appl. Rev.*, vol. 42, no. 3, pp. 397–407, May 2012.

[39] M. Chandra, V. Gupta, and S. K. Paul, "A statistical approach for automatic text summarization by extraction," in *Proc. Int. Conf. CSNT*, 2011, pp. 268–271.

[40] X. Ren and S. Moriya, "Improving selection performance on pen-based systems: A study of pen-based interaction for selection tasks," *ACM Trans. Comput. Hum. Interact.*, vol. 7, no. 3, pp. 384–416, Sep. 2000.

[41] J. Accot and S. Zhai, "Beyond Fitts' law: Models for trajectory-based HCI tasks," in *Proc. SIGCHI CHI*, 1997, pp. 295–302.

[42] L. Stifelman, B. Arons, and C. Schmandt, "The audio notebook: Paper and pen interaction with structured speech," in *Proc. SIGCHI CHI*, 2001, pp. 182–189.

[43] C. T. Meadow, B. R. Boyce, and D. H. Kraft, *Text Information Retrieval Systems*., 2nd ed. New York: Academic, Jan. 15, 2000, 2000.

[44] C.-C. Chang and C.-J. Lin, LIBSVM: A Library for Support Vector Machines , 2001. [Online]. Available: http://www.csie.ntu.edu.tw/~cjlin/libsvm

[45] S. C. Deerwester, S. T. Dumais, T. K. Landauer, G. W. Furnas, and R. A. Harshman, "Indexing by latent semantic analysis," *J. Amer. Soc. Inf. Sci.*, vol. 41, no. 6, pp. 391–407, Sep. 1990.

[46] M. A. Hearst, "The debate on automated essay grading," *IEEE Intell. Syst.*, vol. 15, no. 5, pp. 22–37, Sep./Oct. 2000.

[47] Y.-Y. Chen, C.-L. Liu, T.-H. Chang, and C.-H. Lee, "An unsupervised automated essay scoring system," *IEEE Intell. Syst.*, vol. 25, no. 5, pp. 61–67, Sep./Oct. 2010.

Chien-Liang Liu received the M.S. and Ph.D. degrees in computer science from National Chiao Tung University, Hsinchu, Taiwan, in 2000 and 2005, respectively.

He is currently a Postdoctoral Researcher with the Department of Computer Science, National Chiao Tung University. His research interests include machine learning, natural language processing, information retrieval, and data mining.

Wen-Hoar Hsaio received the B.S. degree from the Department of Computer Science and Information Engineering, Chung Cheng Institute of Technology, National Defense University, Taoyuan, Taiwan, in 1980 and the M.S. degree from the Department of Computer Science, National Chiao Tung University, Hsinchu, Taiwan, in 1996, where he is currently working toward the Ph.D. degree.

His research interests include information retrieval, Web mining, and machine learning.

Chia-Hoang Lee received the Ph.D. degree in computer science from the University of Maryland, College Park, in 1983.

He was formerly a Faculty Member with the University of Maryland and Purdue University, West Lafayette, IN. He is currently a Professor with the Department of Computer Science, National Chiao Tung University, Hsinchu, Taiwan. His research interests include artificial intelligence, human machine interface systems, natural language processing, and opinion mining.

Hsiao-Cheng Chi received the M.S. degree in computer science from National Chiao Tung University, Hsinchu, Taiwan, in 2010.

He is currently an Engineer with Foxconn International Holdings, Ltd., New Taipei City, Taiwan. His current research interests include information retrieval, natural language processing, and data mining.