Creating Personal Histories from the Web Using Namesake Disambiguation and Event Extraction

Rui Kimura[1], Satoshi Oyama[2], Hiroyuki Toda[3], and Katsumi Tanaka[2]

[1] KDDI Corporation
3-10-10 Iidabashi, Chiyoda-ku, Tokyo 102-8460, Japan
ui-kimura@kddi.com
[2] Graduate School of Informatics, Kyoto University
Yoshida-Honmachi, Sakyo-ku, Kyoto 606-8501, Japan
{oyama,tanaka}@dl.kuis.kyoto-u.ac.jp
[3] NTT Cyber Solutions Laboratories, NTT Corporation
1-1 Hikari-no-Oka, Yokosuka, Kanagawa 239-0847, Japan
toda.hiroyuki@lab.ntt.co.jp

Abstract. We have developed a system for gathering information from the Web, using it to create a personal history, and presenting it as a chronological table. It simplifies the task of sorting out the information for various namesakes and dealing with information in widely scattered sources. The system comprises five components: namesake disambiguation, date expression extraction, date expression normalization and completion, relevant information extraction, and chronological table generation.

Keywords: Web search, namesake disambiguation, event extraction, clustering, machine learning.

1 Introduction

It is said that queries containing a person's name account for 5-10% of all Web searches [1]. Many users consider information about people as search target. Many Internet users search for information about people, and the number will increase as Web relationships become more and more common, as evidenced by the explosion of social networking services such as MySpace[1]. Users are searching not only for the profiles of people, but also for pictures of them, information about events in which they participated, gossip and rumors about them, and anything else related to their history. Gathering such a wide variety of information about a person is troublesome.

Efforts to improve this process have led to several interesting alternatives. The system proposed by Al-Kamha et al. [2] clusters Web pages returned by a search engine for person-name queries by using three independent measures: attributes like phone number, e-mail address, and zip code, similarity between Web pages, and link relationships between Web pages. WebHawk [3], a system developed by

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

L. Baresi, P. Fraternali, and G.-J. Houben (Eds.): ICWE 2007, LNCS 4607, pp. 400–414, 2007.
© Springer-Verlag Berlin Heidelberg 2007

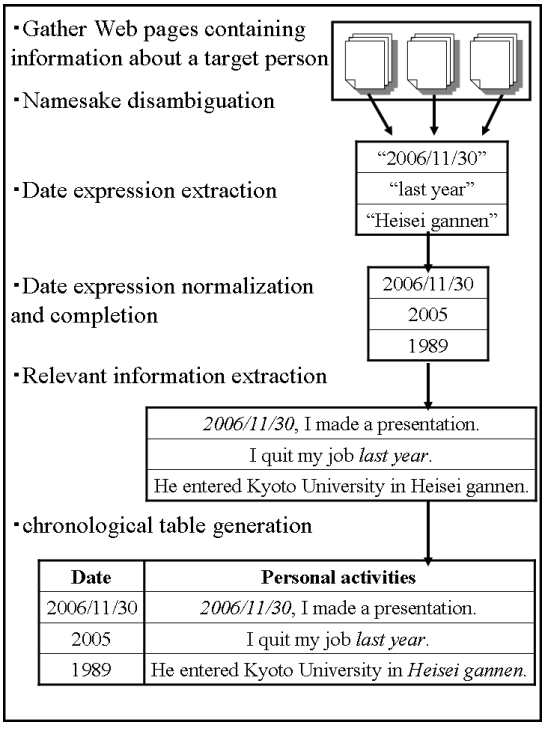

Fig. 1. System overview

Wan et al., extracts personal information, organizations, e-mail addresses, etc. from Web pages and clusters the pages on the basis of the extracted information. It then generates an informative description for each cluster so that the user can quickly identify the target person.

In contrast to these earlier efforts, which extract only basic information, we have developed a system that generates a more comprehensive summary for the target person by extracting information about his or her past activities as well as the basic information. This system enables users to easily obtain a more complete picture of the target person. Moreover, it presents the temporal information obtained about the person in a timeline format, as might be found in a biography or a chronological table. An overview of this automatic generation of a chronological table is shown in Figure 1.

The system uses the name of the target person to search for and gather Web pages containing information about the person. It does this using an existing search engine. If pages are found for different people having the same name, the system clusters the pages by person. The user then specifies the cluster corresponding to the target person. The system then collects the activities for the specified target person from the pages in the corresponding cluster. For the

system to be able to generate a chronological table, a date must be assigned to each activity. The system does this by identifying the expressions in each page that represents a date and then extracting the expressions and corresponding activities. Finally, using the extracted pairs of date expressions and activities, the system places the activities in chronological order and generates a chronological table.

2 Namesake Disambiguation

While the previous systems analyze the original Web pages found by the search engine, our system analyzes only the terms in the snippets shown on the search results pages. It then clusters the snippets to identify the pages belonging to each person. Compared with analyzing the original Web pages, our approach considerably reduces the time taken for downloading the pages and analyzing them.

Our system takes into account the kinds of terms in the snippets and page titles to distinguish between namesakes. Related objects and occupation terms are extracted from the page titles and snippets on the search results pages, and the search results are clustered using the extracted terms to distinguish the namesakes.

The names of people and organizations, i.e., "related objects," are named entities. According to the definition used by IREX (Information Retrieval and Extraction Exercise) [4], a named entity consists of eight kinds of terms, ORGANIZATION, PERSON, LOCATION, ARTIFACT, DATE, TIME, MONEY, and PERCENT. Many efforts have been made to extract named entities from text over the last decade or two. We use the Named Entity Extraction Tool (NExT)[2] developed by Watanabe et al. [5] to extract related objects.

We define occupation terms as terms that can be classified as job, role, or position in Nihongo Goi Taikei - a Japanese lexicon [6], and that consist of more than two letters. We add these terms to a dictionary of ChaSen [7], a Japanese morphological analysis system, and extract them. Example occupation terms are listed in Table 1.

As a baseline method, we extract nouns from page titles and snippets by using ChaSen.

We create a document vector based on the vector space model. We use terms extracted and weight them using term frequency (TF) and inverse document frequency (IDF). For document d and term m_x, the weight is given by

$$w(d, m_x) = tf(d, m_x) \frac{1}{1 + \log(df(m_x))} \tag{1}$$

where $tf(d, m_x)$ is the number of occurrences of term m_x in document d and $df(m_x)$ is the number of documents including term m_x.

[2] http://www.ai.info.mie-u.ac.jp/ next/docs/20040116_NExT.pdf

Table 1. Example occupation terms

Category	Examples
Job	medical director, analyst, interpreter, graduate student, foreign student, golfer, goal keeper, Shinto priest, Member of Parliament
Role	king, chief cabinet secretary, store manager, auditor, assistant, doctor, Member of Parliament, assistant general manager, executive director
Position	staff, instructor, CEO, screenwriter, closer, starting pitcher, home-plate umpire, autocrat, puppet

We make document vectors

$$\boldsymbol{v}_{noun}(d) = (w(d, l_1), w(d, l_2), \cdots, w(d, l_m)) \tag{2}$$

where $w(d, l_x)$ is the weight of term l_x in document d and m is the number of different terms appeared in the document set. We cluster documents using these document vectors and the single linkage clustering method, which is a hierarchical clustering method.

3 Date Expression Extraction

To collect timeline information, we have to extract date expressions indicating when each event happened. Mizobuchi et al. [8] divided terms representing time into time expressions (representing hours, minutes, and seconds) and date expressions (representing years, months, and days). Since we want to create a historical timeline for the target person, we define the smallest time period as a day.

Date expressions are named entities. We could extract them using the NExT named entity extraction tool, which was mentioned above, but this tool cannot extract date expressions written in Japanese hiragana. Moreover, it extracts expressions that are of no use in constructing a timeline, such as "the first year" in "the first year of high school." Therefore, we constructed specialized rules.

We constructed the extraction rules using regular expressions, like those shown in Table 3. Examples of target date expressions are shown in Table 2. As shown in Table 3, the date expressions are categorized as either complete or incomplete. A complete expression has the full complement of year, month, and day, while an incomplete expression lacks one or two of them, indicates a relative time, such as "two years ago" or "four months later," or indicates an ambiguous time, such as "at the end of the year." Other incomplete expressions include the seasons, like "summer," and ambiguous expressions like "early part of October" or "some years later." While "at the end of the year" probably means December, "summer" could mean one of several months. The day is ambiguous in "early part of October," and the number of years in "some years" in "some years later" is unclear. Though these expressions are often used in ordinary texts, we do not extract them due to the ambiguities.

Using regular expressions, we extract date expressions from the clustered Web pages (after removing their HTML tags). Terms that indicate a time transition,

Table 2. Example date expressions to be collected

Completeness	Type	Example
Complete	numeric	2005/4/13
	Western	May 3, 1999
	Japanese	Heisei 18 Nen 12 Gatsu 12 Nichi
	proper noun	New Year's Day of 2003
Incomplete	relative	two years ago
	ambiguous expression	at the end of the year
	year and month abbreviated	on Wednesday, 29th
	year abbreviated	May 3
	month and day abbreviated	Heisei 18
	day abbreviated	May 2003

Table 3. Examples of regular expressions used to extract date expressions

Regular Expressions
`([0-9]+)[/\.\-]([0-9]+)[/\.\-]([0-9]+)`
(*Heisei*|*Syowa*|*Taisyo*|*Meiji*)`\s*([0-9]+|` (*ju*|*hyaku*|*sen*|*ichi*|*ni*|*san*|*shi*|*go*|*roku*|*shichi*|*hachi*|*ku*)`+`|*gan*)(*nendo*|*nen*)
`([0-9]+|`(*ju*|*hyaku*|*sen*|*ichi*|*ni*|*san*|*shi*|*go*|*roku*|*shichi*|*hachi*|*ku*)`+)` `\s*(`*n*endo|*n*en|*g*etsu|*n*ichi)
`([0-9]+|`(*ju*|*hyaku*|*sen*|*ichi*|*ni*|*san*|*shi*|*go*|*roku*|*shichi*|*hachi*|*ku*)`+)` `\s*(`*nen*|*kagetsu*|*getsu*|*nichi*)(*mae*|*ato*)

such as "next year" and "two days ago," are useful, but terms that indicate a time period, such as "for five days," are not, so these latter terms are not targeted.

Terms indicating year, month, or day appearing adjacently are combined, and the combined expression is considered to be one date expression. For example, if "in October" appears next to "on the last day" or, in Japanese, these two terms are connected with the Japanese particle "no," they are combined into "on the last day in October," which is then taken as one date expression.

4 Date Expression Normalization and Completion

After the date expressions, like the examples in Table 2, are extracted, the years, months, and days are normalized. A year is normalized into a four-digit number, a month into a two-digit number, and a day into a two-digit number. A complete date expression is thus normalized into an eight-digit number. or example, "1982/5/3" is converted into 19820503 and "Heisei 18 nen 6 gatsu" is converted into 20060600.

The time transition expressions are incomplete and cannot be normalized. Instead, their meaning is completed by using the context to estimate the date represented. More specifically, the meaning of an incomplete expression is com-

Table 4. Examples of completed date expressions

Incomplete expression problem	Preceding expression information used	Completed expression
May 23 (year missing)	April 13, 2005 (year)	20050523
the fifth day (year and month missing)	May 3, 1999 (year and month)	19990305
in the same month (no time transition)	May 3, 1999 (year and month)	19990500
this year (no time transition)	Heisei 18 Nen 12 Gatsu (year)	20060000
yesterday (time transition)	2005/4/13 (year, month, and date)	20050412
a half year later (time transition)	2005/4/13 (year and month)	20051000

pleted by identifying time expressions appearing prior to the incomplete expression and using the information they contain to complete the expression. Examples of completed date expressions are shown in Table 4.

Three rules are used for completion. The first rule covers expressions in which the year or the year and month are abbreviated. Such expressions are completed by using the corresponding data from a preceding date expression. The second rule covers expressions without a time transition, for example, "at the same month." Such expressions are completed by using a date with the same date in a preceding date expression. The third rule covers expressions with a time transition. If the expression represents l years and m months and n days later, the expression is completed using a date that is l years and m months and n days later than the date represented by the preceding date expression. Similarly, if the expression represents l years and m months and n days ago, the expression is completed using a date that is l years and m months and n previous to the date represented by the preceding date expression. In these third rule calculations, the significant values of both the incomplete date expression and the preceding date expression are considered. Consider the examples shown in Table 4. The incomplete expression "yesterday" is completed using the preceding date expression "2005/4/13." In both expressions, year, month, and day are significant, so the completion calculation is "subtract one day from 2005/4/13." In contrast, only year and month are significant in the incomplete expression "a half year later," while year, month, and day are significant in the preceding date expression "2005/4/13." The completion calculation is thus "add six months to 2005/4."

5 Relevant Information Extraction

Since the Web pages often have information for more than one person with the same name, the system has to identify the information that is relevant to the

target person. Moreover, it has to determine whether the paragraph is information is relevant to the person's history. It does this by using the HTML tree structure of each page and machine learning.

Previous approaches to exacting information from Web pages are generally usable only for Web pages with similar structures, design, or contents or for a few pages with many instances on each page. Our system is designed to extract information from all Web pages containing relevant information. Therefore, it must be able to extract information from various types of Web pages. Moreover, different queries should return different personal histories, so the extracted results must be query dependent. However, since it is virtually impossible to create an extraction rule for each query, i.e., each target person name, the extraction rule must be query independent. We thus use a robust approach to extracting relevant information—the use of a rough context-content model. Using the HTML tree structure of the retrieved pages, the system can guess the context in which each paragraph is used.

While XML tags describe the architecture of a page, HTML tags describe the design. The HTML tag names are not the name of a class, as are XML tags, so the tags around text do not indicate the meaning of the text. Nevertheless, many documents, newspaper stories, magazine articles, and so on have a heading representing the content, so the content can be roughly identify by simply looking at the heading. Moreover, the heading is usually easily distinguishable from the text. For example, the heading is generally located above the text. The same is true of Web pages containing HTML tags. Therefore, estimating whether some text contains information about the target person can be done by checking the heading.

The W3C's HTML Website[3] explains that HTML tags are either for block-level elements or for in-line elements. Block-level elements create larger structures and begin on a new line. They are more important for the HTML paragraph structure than the in-line elements, so we use the tags for the block-level elements and remove those for the in-line elements before extracting relevant information.

The system extracts context C_A of text T_A, which is enclosed by a pair of tags, using the following algorithm. The HTML node to which T_A belongs is N_T, and the HTML node to which C_A belongs is N_C. If N_X is an HTML node and N_{root} is the HTML root node,
PreviousNode(N_X) is defined as follows.

PreviousNode(N_X) =

$$\begin{cases} \text{(i) PreviousSiblingNode}(N_X) \\ \qquad \text{if PreviousSiblingNode}(N_X) \text{ is not null,} \\ \text{(ii) ParentNode}(N_X) \\ \qquad \text{if PreviousSiblingNode}(N_X) \text{ is null} \\ \qquad \text{AND if ParentNode}(N_X) \text{ is not null} \\ \text{(iii) } N_{root} \end{cases} \tag{3}$$

[3] http://www.w3.org/TR/html401/struct/global

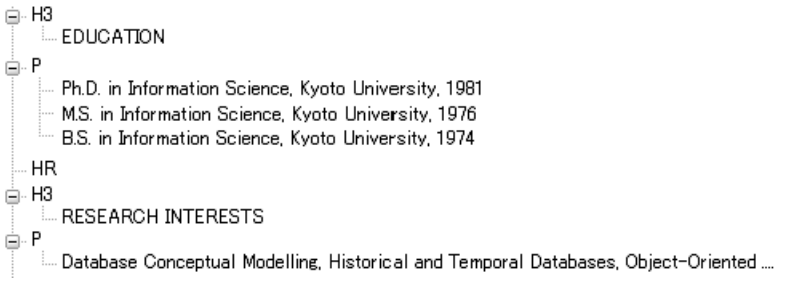

Fig. 2. HTML tree

The node for which the system is searching defined as N_S, and at first N_S is the parent node of N_T. Using the definition of PreviousNode(N_X), N_C is recursively searched as follows.

$N_C =$

$$\begin{aligned} &\text{(i) RootNode} \\ &\qquad \text{if PreviousNode}(N_S)\text{is RootNode,} \\ &\text{(ii) PreviousNode}(N_S) \\ &\qquad \text{if PreviousNode}(N_S)\text{is a text node,} \\ &\text{(iii) the child node of PreviousNode}(N_S) \\ &\qquad \text{if PreviousNode}(N_S)\text{is a node of a heading tag.} \end{aligned} \tag{4}$$

If N_C has not been determined, N_S is set to PreviousNode (N_S) and Formula 4 is repeatedly applied until N_C is determined. If N_C is determined by Formula 4, the context, C_A, is defined as the inner-text of N_C.

Using these algorithms, the system can find a context for each text item on a Web page. Consider the example HTML tree in Figure 2 and a search for the context of the text "Database Conceptual Modelling, ..." First, N_S is set to the previous node in the text, the P tag node. Next, N_S is searched for and the H3 tag node is found. Using Formula 4, the system determines that N_S is a node of a heading tag, meaning that N_C is a child node of N_S, the node for "RESEARCH INTERESTS." In this way, the context of "Database Conceptual Modelling, ..." is determined to be "RESEARCH INTERESTS."

A feature vector is constructed for each identified context-text pair on a page. The nouns on the page are extracted using Chasen, and each one is defined as a feature. The feature value is the number of occurrences of the noun on the page. Although numbers representing dates would be useful for identifying information useful for generating the chronological table, if all the numbers on a page were defined as features, the number of features would be burdensome. Moreover, number features rarely occur more than once in the data set. This causes feature sparseness and low classification accuracy. Therefore, the numbers are defined as one of five features: "one-digit number," "two-digit number," "three-digit number," "four-digit number," and "more-than-four-digit number," as shown in Table 5. All date expressions are defined as "date expression."

Table 5. Aggregated features used to train binary classifier

Type	Feature	Example
numbers	one digit number	3
	two digit number	32
	three digit number	524
	four digit number	1999
	more-than-five digit number	150,000,000
names of people	queried name	Hanako Sato
	part of queried name	Sato
	other person's name	Tanaka
date expressions	date expression	1999/12/1

Since we want to extract information only for the target person, the query name, part of the query name, and the names for other people are good features for extracting relevant information. However, if the names were directly used as features, the rules would greatly depend on the query names used to collect the training data of the machine learning; therefore, when the rules were applied to different data, the accuracy would be low. To make the rules independent of the queries, we abstract the features. The names of people are defined as one of three features, "queried name," "part of queried name," and "other person's name." This aggregation of features also helps prevent overfitting to the training data and results in good generalization to other data. Using these features, we train a binary classifier to classify the segments on a page into relevant and irrelevant ones.

6 Chronological Table Generation

To generate a chronological table for the target person, it is necessary to collect pairs of a date and an activity for the person. The sentences extracted likely include activities for the person. The style of the sentences may vary widely; for example, some sentences may have no date expressions and some may have more than one pair of a date and an activity. If a sentence has no date expression, it is necessary to search among the preceding sentences for a date expression related to the activity. If a sentence has more than one pair of a date and an activity, it is necessary to match the dates with the information. We have created four rules for doing this.

- The system first extracts sentences or parts of sentences that include date expressions. These expressions are normalized and completed, and the system makes a list of candidate texts which are the extracted sentences or incomplete sentences.
- If an extracted date-expression sentence does not end with a noun that is linked to a *sa-hen* verb, the sentence is eliminated from the candidate list.

– If the first word following a date expression is not "ni," "yori," "kara," or a space, the sentence is eliminated from the candidate list.
– If a sentence does not have Japanese words, it is eliminated from the candidate list.

The candidate texts remaining following the application of these rules are used to generate a chronological table.

7 Evaluation

7.1 Namesake Disambiguation

As sample queries, we chose 20 Japanese names from a list of names that has at least three famous people in an article "same last name and first name" of Wikipedia[4]. We did a Google search on each name and used the first 100 results for each one. We regarded a pair of a page title and a snippet search result as one document. For each name, we made an answer set by manually clustering these documents so that each cluster corresponded to one person. The average number of clusters in the answer set was 22.85. The maximum number of clusters in correct data was 48, the same as the number of clusters for "Yutaka Kobayashi," and 12.25% of the clusters had only one document.

The clustering results were evaluated by comparing them to the answer set (correct clusters), which was prepared manually, using precision, recall, and F-measure.

$$F\text{-}measure = \frac{precision * recall * 2}{precision + recall}. \tag{5}$$

We used an F-score measure [9] as the metric clustering accuracy.

$$F\text{-}score = \sum_{c \in C} \frac{|m_c|}{|m|} \max_{r \in R} \frac{2|m_r \cap m_c|}{|m_r| + |m_c|}, \tag{6}$$

where C is the set of clusters in the answer set, c is one cluster in the answer set, m_c is the set of documents belonging to cluster c, R is the set of clusters in the clustering result, r is one person in a classified result, m_r is the set of documents belong to cluster r, and m is a set of documents in the answer set.

We varied the number of clusters, which was used as the stopping condition for single-linkage clustering, between 1 and 99. The precision, recall, F-measure, and F-score for each name were averaged for each number of clusters. We created document vectors consisting of nouns and related objects and/or occupation terms and used them for our evaluation.

Figure 3 shows precision-recall curves for the classification results using our system and the different types of feature vectors as well as for a baseline method

[4] http://ja.wikipedia.org/w/index.php?title=%E5%90%8C%E5%A7%93%E5%90%8C%E5%90%8D&oldid=7825577; dated 08:34, 22 September 2006

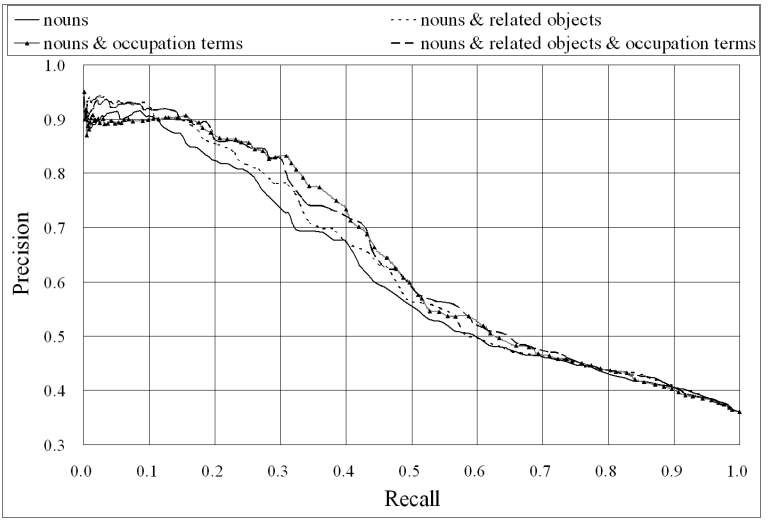

Fig. 3. Clustering accuracy

using only nouns as features. For all three types of feature vectors, our system performed better than the baseline method.

Table 6 shows the highest F-measure and F-score for all cluster numbers (denoted by subscript "max") and those for the actual cluster number in the answer set ("answer"). All the maximum scores were scored using the feature vectors with nouns and related objects and occupation terms. Using the vectors with nouns and occupation terms and the vectors with nouns and related objects and occupation terms produced significantly better results than the baseline method.

7.2 Date Expression Extraction

We did a Google search on five Japanese names (Katsumi Tanaka, Satoshi Oyama, Yuri Ebihara, Junichiro Koizumi, and Hidetoshi Nakata) and used the first 20 results for each one. Then, using the 100 web pages, we evaluated the accuracy of our extraction component. Table 7 shows the number of automatically extracted date expressions, the number that were correct, and the actual number manually found. The recall was 71.3%, and the precision was 93.3%.

While the precision was sufficiently high, it can be improved by eliminating some mistake patterns. One pattern in particular (exemplified by "two days in Cairns") accounted for more than half the mistakes. There were also misrecognition problems. For example, the "6.1.1" in "section 6.1.1" was recognized as "6/1/1." Problems also occurred because the same Chinese ("kanji") character can have more than one meaning. In Japanese, both "moon" and "month" use the same Chinese character, as do "day" and "sun." This multiple usage of characters complicates the extraction of date expressions. Other misrecognized terms are part of a proper noun, which often raise in Japanese text.

Table 6. Clustering accuracy

Method	$F\text{-}measure_{max}$	$F\text{-}score_{max}$	$F\text{-}measure_{answer}$	$F\text{-}score_{answer}$
nouns	0.509	0.637	0.557	0.629
nouns & related objects	0.514	0.631	0.563	0.631
nouns & occupation terms	0.513	0.651	0.572	0.632
nouns & related objects & occupation terms	0.516	0.651	0.575	0.639

Table 7. Precision and recall of date expression extraction

Query	Extracted	Correct	Actual	Precision	Recall
Katsumi Tanaka	473	468	559	0.989	0.837
Satoshi Oyama	223	205	514	0.919	0.399
Yuri Ebihara	323	317	352	0.981	0.901
Junichiro Koizumi	505	435	557	0.861	0.781
Hidetoshi Nakata	293	271	397	0.925	0.683
Total	1817	1696	2379	0.933	0.713

It is difficult to avoid all extraction mistakes when using regular expressions. It is not practical to write rules for all patterns of the various proper nouns that appear in Web pages. We may need to use a morpheme analyzer to determine whether a term actually represents a date.

The recall was lower than precision because we emphasized precision in order to get a precise date for each piece of information. We did not extract uncertain terms, such as "2/3," which can be either a fraction or a date, and "18," which cannot be identified as a year but sometimes means "Heisei 18." Such date expressions could possibly be extracted by using words appearing around them. For example, date expressions tend to be collocated with certain prepositions like "in" ("in May") and "on" ("on May 5th"). They also tend to occur with specific words, like a day of the week, such as "Wednesday."

7.3 Date Expression Normalization and Completion

We did a Google search on each name in Table 7 and used the first 100 results for each one. In the obtained data, there were 5264 date expressions, and 1159 of them were incomplete. On average, each of the 500 pages had about 10.5 date expressions, and about 2.3 of them needed to be completed. This meant that our objective was to complete one-fourth of the date expressions by using the other three-fourths. If the first date expression in a text did not have the year, the three rules defined for completion could not be applied. They could also not

Table 8. Precision of date expression completion

	Rule 1		Rule 2		Rule 3	
	Applied	Precision	Applied	Precision	Applied	Precision
Katsumi Tanaka	103	0.922	6	0.333	8	0.250
Satoshi Oyama	142	0.951	4	1.000	5	0.200
Yuri Ebihara	141	0.702	11	0.727	8	0.625
Junichiro Koizumi	139	0.626	17	0.353	24	0.583
Hidetoshi Nakata	195	0.610	15	0.867	16	0.250
Total	720	0.743	53	0.623	61	0.426

be applied to subsequent date expressions in that text lacking the year. There were 325 of these date expressions.

The date expressions to which the rules could not be applied were classified into "people can complete" (about 61%) and "people cannot complete" (about 39%) depending on whether a person could estimate the date by using the URL, title, and/or text on the Web page. For example, if the URL for the page was "http://blog.example.com/20050602/index.html," a person could determine that it was written on 2 June 2005. If the title of the page was "EURO2004," a person could determine that the events described on the page happened in 2004. The remaining 834 incomplete date expressions were completed by applying the rules. About 71.2% of them were completed correctly, so using only simple completion rules produced relatively good precision.

Table 7 shows the precision of each rule. The precision of the third rule, which was applied to date expressions with a time transition such as "three months ago," was lower than those of the other rules. When constructing our three completion rules, we assumed that a time transition as one from a date of adjacent date expressions; however, in blog posts and news articles, most time transitions are from dates when the post or article was written. This explains why most of the incorrect completions were for blog posts and news articles. Since the names of famous people, which are searched for more often than other names, are frequently used in blog posts and news articles, this is a significant problem.

Since about 61% of the date expressions to which the rules could not be applied were classified into "people can complete," performance could be improved by using more complex rules and additional information, like the URL.

7.4 Relevant Information Extraction

We conducted a preliminary experiment to compare the relevant information extraction performance of three machine learning methods: Naive Bayes [10], C4.5 decision tree learner [11], and SVM [12]. The C4.5 learner had the best classification accuracy. We thus used it to evaluate the classification accuracy of our system.

Table 9. Precision and recall of relevant information extraction

Use context-text model	true	true	false	false
Use aggregation of features	true	false	true	false
Precision	0.593	0.073	0.546	0.103
Recall	0.143	0.045	0.119	0.007
F-measure	0.231	0.055	0.195	0.013

For experimentation, we use 20 Japanese names used for experimentations in disambiguation of namesakes. We downloaded ten random web pages from the first 100 search results for each name. The text elements for each pages were classified manually to produce an answer set for 200 Web pages.

In this evaluation, we cross-validated the results by dividing the answer set into 20 subsets. One name out of 20 names was selected, and the 190 pages corresponding to the other names were used as training data. The ten pages corresponding to the name were used as test data. We repeated this evaluation 20 times, once for each name and calculated micro averages for precision, recall, and F-measure.

To evaluate our system, which uses both a context-text model and feature aggregation, we compared the accuracy for four data sets: one using both the context-text model and feature aggregation, one using only the context-text model, one using only feature aggregations, and one using neither. Due to the computational cost, we counted the number of occurrences in a data set for each feature and used the top 100 features. The results are shown in Table 9.

The scores for our system using both the context-text model and feature aggregation were better than those of the other methods. Especially, feature aggregation was very effective for improving both precision and recall. Even for the best method, however, its recall was not very high. This was probably due to the great difference between the number of positive examples and negative examples. The data used contained only 1.55% positive examples. We plan to incorporate an existing technique for handling imbalanced data, such as artificially generated virtual positive data [13].

8 Conclusion

Our proposed system gathers information from the Web, uses it to create a personal history, and presents the history as a chronological table. It simplifies the task of sorting out the information for various namesakes and dealing with information in widely scattered sources. The system has five components: namesake disambiguation, date expression extraction, date expression normalization and completion, relevant information extraction, and chronological table generation. This system will thus enable efficient creation of personal histories.

Acknowledgments

This work was supported in part by Grants-in-Aid for Scientific Research (Nos. 18049041 and 19700091) from MEXT of Japan and by a MEXT project entitled "Software Technologies for Search and Integration across Heterogeneous-Media Archives."

References

1. Guha, R., Garg, A.: Disambiguating people in search. Stanford University (2004)
2. Al-Kamha, R., Embley, D.W.: Grouping search-engine returned citations for person-name queries. In: Proc. ACM WIDM 2004, pp. 96–103. ACM Press, New York (2004)
3. Wan, X., Gao, J., Li, M., Ding, B.: Person resolution in person search results: Webhawk. In: Proc. ACM CIKM 2005, pp. 163–170. ACM Press, New York (2005)
4. IREX Committee. In: Proceedings of the IREX Workshop, IREX Committee (1999)
5. Watanabe, I., Masui, F., Fukumoto, J.: Improvement of next performance: Elavolating precision and userbility of the named entity extraction tool. In: Proc. NLP 2004, pp. 413–415 (in Japanese) (2004)
6. Ikehara, S., Miyazaki, M., Shirai, S., Yokoo, A., Nakaiwa, H., Ogura, K., Ooyama, Y., Hayashi, Y.: Nihongo Goi Taikei - A Japanese Lexicon (CD-ROM). Iwanami Syoten (in Japanese) (1999)
7. Matsumoto, Y., Kitauchi, A., Yamashita, T., Hirano, Y., Matsuda, H., Takaoka, K., Asahara, M.: Japanese Morphological Analysis System ChaSen version 2.2.1 (2000)
8. Mizobuchi, S., Sumitomo, T., Fuketa, M., Aoe, J.: A method for understanding time expressions. In: Proc. IEEE SMC 1998, pp. 1151–1155. IEEE Computer Society Press, Los Alamitos (1998)
9. Zhao, Y., Karypis, G.: Evaluation of hierarchical clustering algorithms for document datasets. In: Proc. ACM CIKM 2002, pp. 515–524. ACM Press, New York (2002)
10. John, G.H., Langley, P.: Estimating continuous distributions in bayesian classifiers. In: Proc. UAI 1995, pp. 338–345 (1995)
11. Quinlan, R.: C4.5: Programs for Machine Learning. Morgan Kaufmann, San Francisco (1993)
12. Vapnik, V.: Statistical Learning Theory. Wiley, Chichester (1998)
13. Chawla, N.V., Bowyer, K.W., Hall, L.O., Kegelmeyer, W.P.: Smote: Synthetic minority over-sampling technique. JAIR 16, 321–357 (2002)