A Two-Step Approach to Extracting Attributes for People on the Web

Keigo Watanabe
The University of Tokyo
watanabe@mi.ci.i.u-tokyo.ac.jp

Danushka Bollegala [*]
The University of Tokyo
danushka@mi.ci.i.u-tokyo.ac.jp

Yutaka Matsuo
The University of Tokyo
matsuo@biz-model.t.u-tokyo.ac.jp

Mitsuru Ishizuka
The University of Tokyo
ishizuka@i.u-tokyo.ac.jp

ABSTRACT

Personal names are among one of the most frequently searched items in web search engines. Extracting information in the form of attributes and values for a particular person enables us to uniquely identify that person on the web. For example, although namesakes share the same name they usually have different date of births or affiliations. Given a set of documents retrieved for a particular person, we propose two stage approach to extract values for a set of attributes for that person. In the first stage we mark all potential attribute strings in a given text. The second stage then attempts to select the attribute values relevant to a person name. We use a named entity recognition tool to mark all occurrences of named entities in a given document. We then use a rule-based tagger to identify the variants of the given person name. Next, we employ a combination of rules and pre-compiled attribute value candidate lists to extract values for a given set of attributes. The candidate value lists are manually created using resources available on the web such as Wikipedia. The proposed method is evaluated on the test data collection created for the attribute extraction subtask at the second Web People Search Task (WePS). According to the results in the official evaluation, the proposed method is ranked 5-th among the 15 participating systems.

Categories and Subject Descriptors

H.3.3 [**Information Systems**]: Information Search and Retrieval

General Terms

Algorithms

Keywords

Attribute Extraction, People Search, Web Mining

1. INTRODUCTION

A person is associated with numerous attributes on the Web. Accurate extraction of attributes for a particular person is important to uniquely identify that person on the web. For example, in the case of namesakes (i.e. people with identical names), although they share the same name, the other attributes such as affiliation, nationality, date of birth, place of birth, etc. might be different. Consequently, extracting various attributes has shown to be useful for personal name disambiguation. For example, consider the two namesakes of the ambiguous name *Jim Clark*: one is a racing driver whereas the other Jim Clark is the founder of Netscape corporation and also a university professor. The attribute *occupation* can separate the two Jim Clarks because one is a sports car driver and the other is a university professor.

Web People Search Task (WePS) is aimed at searching for people on the web. The first WePS introduced a name disambiguation task where given a collection of web documents retrieved for a particular name, the objective is to identify the documents that belong to different people with the queried name. The problem can be conveniently modeled as a one of document clustering where each cluster represents a different person of the given ambiguous name. It was found that attributes such as date of birth, nationality, affiliation, occupation, etc. are particularly useful as features to identify namesakes [2]. Consequently, in the second WePS [1], an attribute extraction subtask was introduced. Given a web document, the objective of this attribute extraction task is to extract a pre-defined set of attribute values for a given person name. The WePS attribute extraction task focuses on extracting values for the following 18 attributes: date of birth, birth place, other name, occupation, affiliation, work, award, school, major, degree, mentor, location, nationality, relatives, phone, fax, e-mail, and web site. The definition of each attribute can be found in the task description guide [1]. However, not at attributes are equally represented in the WePS dateset. The most frequent attributes are Work (3770), Occupation (3292), and Affiliation (3105). Here, the total number of occurrences are shown within brackets. Attributes such as fax (65), web site (154), and major (173) are the least frequent attributes in the dataset.

A system that attempts to extract attributes for a given person from web documents must solve several sub-problems. First, it must identify the different occurrences of the given person name. This is challenging because of two main problems: namesakes and name aliases on the web. Although a web page might contain the given person name it could be a page for a different person who has the identical name. In attribute extraction task at the second WePS workshop only documents relevant to the person under consideration are given. Therefore the problem of namesake disambiguation does not occur. In fact, the objective of attribute extraction

[*] Research Fellow of the Japan Society for the Promotion of Science (JSPS)

WWW 2009, April 20–24, 2009, Madrid, Spain.
ACM 978-1-60558-487-4/09/04.

[1] http://nlp.uned.es/weps/weps2/WePS2_Attribute_Extraction.pdf

in WePS is to use the extracted attributes to disambiguate people on the web. However, a particular individual can be represented by more than one name on the web. For example, *William Gates* is more commonly called as *Bill Gates* in web contexts. Moreover, abbreviated variants of names are common. For example, *John Fitzgereld Kennedy* has the variants *J.F.K.*, *John F. Kennedy*, and *J. F. Kennedy*. Although it is relatively easy to cover the above mentioned variants using dictionaries of common name aliases (i.e. Bill vs. William, Jim vs. James, etc.) and regular expressions, some name aliases such as *Fresh Price* for *Will Smith* or *Godzilla* for *Hideki Matsui* are difficult to identify automatically [4].

Once the occurrences of the given name is identified in a set of documents, an attribute extraction system must extract attributes and their values. Attribute extraction step can be further divided into two parts – the attribute extraction system must first identify the values for the given set of attributes and then decide which attribute values are relevant to the person under consideration. Attributes such as *e-mail addresses, urls, telephone numbers, fax numbers*, and *birth dates* follow a specific format and are easy to detect. However, attributes such as *relative, mentor, school, award, degree*, and *affiliation* have many variations and are difficult to detect. For example, a mentor of a person can be introduced in a text as the *teacher*, *advisor*, *professor*, *supervisor*, etc. The set of values such attributes can take is open and cannot be completely enumerate using pre-compiled lists. For example, the attribute *mentor* can take any person name as its value. Named entity recognition tools can solve this problem partially. However, most named entity recognition tools only cover more common entities such as personal names, locations, and organizations. They do not annotate awards, majors, nationalities or classify organizations into schools. Moreover, named entity recognition tools are usually trained on noise-free text corpora such as news articles and do not show optimal performance on relatively noisy web documents with numerous markups such as HTML and Javascript. Therefore, identifying which strings can be potential attributes for a person is an important task that an attribute extraction system must perform.

Finally, an attribute extraction system must select the attribute values relevant to the given person. A document might contain information regarding more than one person. Consequently, not all attributes that appear in a document might be relevant to the person under consideration. A simple yet effective heuristic is to associate attributes closest to an occurrence of the given person name. It is likely that a person denote his or her contact information such as e-mail, telephone and fax close to the name in a home page. However, this simple heuristic cannot cover the cases where a name and a relevant attribute appear in distant parts in a document. Moreover, it is not clear how to handle cases where more than one person name appear in a document – should we go beyond an occurrence of a different name and associate attributes or not.

This paper describes the **MIVTU** system that participated in the attribute extraction subtask at the second WePS workshop. According to the official results, MIVTU was ranked 5^{th} among the 15 systems that participated in the attribute extraction task at the second WePS. However, the highest overall F-scores reported by all participating systems is 12.2. MIVTU system reported an F-score of 8.3. This fact suggests that the challenges described in this section are yet to be properly addressed by the participating attribute extraction systems. This paper is organized as follows. In section 2, we briefly overview the previous work on attribute extraction. Then in section 3 we describe the MIVTU system. Finally, we compare the official results for MIVTU and rest of the systems participated in the attribute extraction task at the second WePS in section 4 and conclude the paper.

2. RELATED WORK

Extracting attribute values for an entity has wide applications in information extraction and retrieval. Pasca [5] proposed a method to extract born year of people from web text. For example, from the text *Mozart was born in 1756*, this method extracts the pair (*Mozart*, 1756). The extraction algorithm starts from as few as 10 seed facts, and is capable of extracting facts from over 100 million web documents. Seed facts are searched on web texts and lexical patterns are generated. Then the generated lexical patterns are searched in web texts and new facts are extracted. The process is repeated with newly extracted facts as seeds. Although this method was used to extract birth year of people, a similar boot strapping approach can be followed to extract other types of attributes such as birth place, nationality and occupation.

Bellare et al. [3] proposed a lightly-supervised approach to attribute extraction from the web. They first tag a given text corpus with part-of-speech information and then from each tagged sentence extract all proper noun and noun pairs. They consider each extracted pair as a candidate entity-attribute instance. Each candidate instance is assigned with a set of features. They select the left, right and middle contexts that appear around the entity and candidate attribute as features. They use two learning methods: decision lists by co-training using a mutual information-based measure, and a maximum-entropy classifier by self-training. They evaluate their algorithm on two tasks: extracting the set of attributes for companies, and extracting the set of attributes for countries. However, they do not extract the values for those attributes.

3. METHOD

3.1 Outline

The proposed method is illustrated in Figure 1 and it can be seen as consisting of two fundamental steps. First, we mark potential attribute values in a given text. Second, we decide which candidate values correspond to which attributes of the given person name.

To mark the potential values of attributes we use three approaches: lists of candidate attribute values, a named entity recognizer, and a set of manually created rules in the form of regular expressions. For example, attributes such as nationalities (e.g. Japanese, British) , universities (e.g. The University of Tokyo), majors (e.g. Master of Science, Bachelor of Arts) and professional titles (e.g. professor, general) can be marked using candidate lists. These lists were created manually referring online information sources such as Wikipeida. However, lists cannot completely enumerate all attribute values. In addition to using pre-compiled lists of attribute values, we used a named entity recognition tool to mark three types of named entities: personal names, organization names, and location names. Attributes such as dates, telephone numbers, fax numbers, e-mail addresses and urls usually follow a fixed format and can be efficiently annotate in a text using rules in the form of regular expressions.

Once the given text is annotated following the above mentioned procedure, we mark all potential variants of the given name for which we must extract attribute values. We generate abbreviated forms, last name and first name inter-changed forms, middle name initialized forms, middle name dropped forms, name followed by titles, and combinations of all the above. We then mark those variants in the given text. For example, if the given name is *John Fitzgereld Kennedy* then this process will generate variants such as *J. F. Kennedy*, *John F. Kennedy*, *Kennedy J. F.*, and *John Kennedy*. To find the attributes of the given person, we find the distance for each marked attribute value from a name variant. We then select the closest attribute value as the correct candidate. However, we do

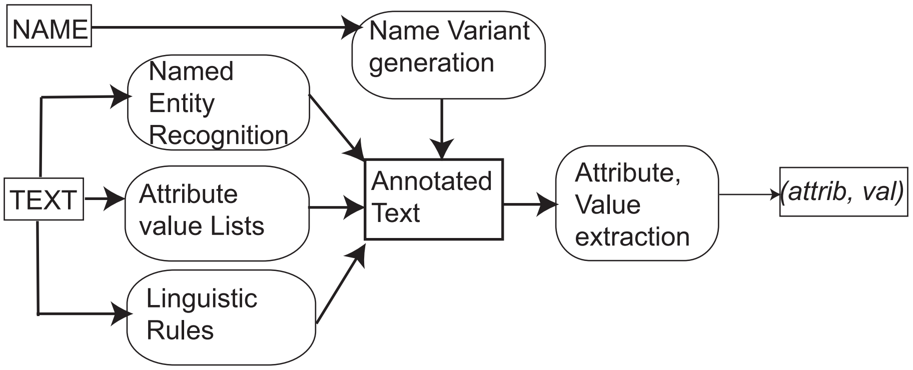

Figure 1: Outline of the proposed system

not go beyond a different person name when computing distances. Moreover, we assign higher confidence score to an extracted attribute value if certain cue phrases appear in close proximity. For example, the cue phrases *born* and *birth* increase the confidence of an extracted date being the date of birth of the person under consideration. Likewise, cue phrases *mentor*, *supervisor*, and *advisor* increase the confidence of a value extracted as the mentor of the person under consideration. The cue phrases are selected manually after reviewing the test data in the WePS attribute extraction dataset. Each sub-component of the proposed attribute extraction system including examples of candidate value lists, linguistics rules, cue phrases, and attribute extraction method will be further explained in the sections to follow.

3.2 Pre-processing

WePS attribute extraction task dataset contains HTML documents for a set of person names. However, named entity recognition tools have difficulties in operating on HTML marked texts. Therefore, we first remove all HTML markups using an external tool[2]. We then use Stanford named entity recognizer[3] and annotate the text for person names, locations and organizations. The remainder of the processing described in the paper use this annotated text version of the dataset and does not use the original HTML version. We use a set of rules to generate probable variants of the given person name. First, we split the given name into first name and last name. We then generate the following variants: first name followed by last name, last name followed by first name, a comma appearing between the two names, a word appearing between the two names, first name initialized and immediately followed by the last name, last name followed by a comma and the first name initialized, and first name initialized and followed by a word and the last name. We also consider all combinations of the above variants with the following titles: Mr., Mrs. Miss., Ms, Rev., Prof., President, Minister, Prime Minister, General, Madame, Lady, Dr., King., Queen, Vice President, Senator, Lawyer, Major, Maj., General, Gen., Maj. Gen., Major General, and Jr. For example, given the name *George Bush* the above mentioned process recognizes the string *president George W. Bush* as a variant of the given name. The process is an over generating one and in practice generates a large number of variants that never occur in the corpus. However, once the candidate variants are generated they can be efficiently matched using regular expressions. Figure 2 shows an example annotation produced by the pre-processing step. In Figure 2 we use the format [TAG_NAME, LENGTH_IN_WORDS] to mark the span of a tag. For example, *Benjamin [VARIANT,1] Snyder* indicates that Benjamin Snyder is a variant of the given name.

3.3 Attribute Extraction

We use the HTML markup removed and annotated text produced by the pre-processing to extract attributes. Next, we describe the extraction procedure for each of the attributes in detail.

Date of birth: We use a set of rules in the form of regular expressions to mark all date strings in the text. We then normalize all date strings to YEAR/MONTH/DAY format. The following Perl versions of the regular expressions are used to mark dates.

```
/((month_exp)\s*(\d+),?\s+(\d+))/gi
/((month_exp)\.?\s*(\d+))/gi
/((\d{1,2})\s*($month_exp)\s*(\d{2,4}))/gi
/((\d{1,2})\s+(\d{1,2})\s+(\d{2,4}))/gi
/((\d+)\/(\d\d?)\/(\d\d?))/g
/((\d+)\/(\d{1,2}))/g
/((\d+)\.(\d\d?)\.(\d\d?))/g
/(\d\d\d\d)/g
```

Here, month_exp is a variable that holds the names of months. Once all dates are marked in the text, we assign confidence scores to dates that appear closer to the given name (or its variants) or have cue phrases such as *born* or *birth*.

Birth place: We use the location markups as given by the named entity recognition tool to identify candidates for the birth place. We then assign confidence scores to locations that appear closer to the given name (or its variants) or have cue phrases such as *birth place* or *born in*.

Other name: The name variant generation procedure described in section 3.2 annotates name variants. We select those variants as other names of the given name. Moreover, we use cue phrases such as *a.k.a.*, *also known as*, *alias*, and *other name* to identify other names for the person under consideration.

Occupation: We created a list of occupations from Wikipedia[4]. The list contains 666 entries. We then select the occupation closest to the given name or any of its variants in the text. Moreover, we tokenized each entry in the occupation list into

[2] http://www.oluyede.org/files/htmlstripper.py

[3] http://nlp.stanford.edu/software/CRF-NER.shtml

[4] http://en.wikipedia.org/wiki/List_of_occupations

Benjamin [VARIANT,1] Snyder and Phedora [ORGANIZATION,3] Blazer Benjamin Snyder and Phoebe Ann Blazer Husband: Benjamin [VARIANT,1] Snyder born 12 DEC 1827 in Dayton, [LOCATION,0] Montgomery [ORGANIZATION,1] Co., OH died 6 JUL 1873 in Montreal, [LOCATION,0] Camden [ORGANIZATION,1] Co., MO.. [LOCATION,0] buried: Freedom [ORGANIZATION,5] Church, Linn Creek, Camden Co., MO [LOCATION,0] married: Phoebe Ann BLASER Bef 1855 in OH Wife: Phoebe Ann BLASER born 25 JUL 1838 in OH died: 20-Feb-1896 in MO [LOCATION,0] buried: Freedom Cemetery -LRB- West side -RRB- Children of Benjamin [VARIANT,1] Snyder and Phoebe [ORGANIZATION,2] Ann Blazer 1. Andrew Jackson Snyder [VARIANT,0] born: JUL 1855 in OH died: 16 OCT 1935 in Tulsa, [LOCATION,0] Tulsa [ORGANIZATION,1] Co., OK. married: Delilah Caroline MOSBY 1878 in IN born: 20 NOV 1866 in Evansville, [LOCATION,0] Vanderburgh [ORGANIZATION,1] Co., IN daughter of Vincent MOSBY and Manerva SAMUELS died: 8 JUN 1910 in Linn [ORGANIZATION,3] Creek, Camden Co., MO [LOCATION,0] buried: Freedom [ORGANIZATION,1] Cemetery - Montreal, [LOCATION,0] ...

Figure 2: Example of an annotated text for the person Benjamin Snyder.

words and sorted the words according to their total frequency within the list. The goal of this is to identify words that are commonly used to describe occupations. If a sequence of words contain any of those high frequency words, we select those sequences as occupations. The most frequent words that occur in occupations are: engineer (11), officer (8), scientist (6), Technologist (5), agent (5), designer (5), Financial (5), and worker (5).

Affiliation: We consider companies and universities as affiliations. We create lists for universities and companies using Wikipedia. Our company names list contains 43040 entries and university list contains 1726 entries. We also find the frequency of words that appear in each of these entity types as we did for occupations. The top 10 most frequent words that appear in company names are: Inc. (16137), Corporation (3932), Ltd. (2277), Limited (2126), Company (1993), LLC (1782), Group (1685), plc (976), and International (835). If a capitalized sequence of continuous words contain those words we mark it as an company. Although words such as *of* (1814), *&* (1814), and *The* (1514) are also highly frequent in company names, we remove such words using a stopwords list because those words are ambiguous and can appear in various contexts not necessarily for companies. Moreover, the named entity recognition tool we used in the pre-processing step also provides some company names. A similar word frequency analysis for university names revealed that the most frequent words that appear in university names to be the following: University (861), College (662), State (234), New (74), Saint (56), and Institute (55).

Work: Works of people are very difficult to extract. Books written by authors and movies created by film directors are such cases. However, what can be a work differs from person to person. It is not feasible to cover all value types for this attribute using lists. MIVTU system does not extract this attribute.

Award: We used Wikipedia to create a list of awards. The list contains 454 entries. Any entry that is found in this list is marked as an occurrence of an award in the given text. We perform a word frequency analysis on this list and found the following words to be the most commonly used in names of awards: Award (85), Prize (62), Medal (58), and Order (41). Any continuous sequence of capitalized words that include these words are marked as awards. However, some awards such as *Common Wealth Award of Distinguished Service* and *National Medal of Science* contain the preposition *of* which is not capitalized. In fact we found *of* to appear in 98 times in award names. Initially, we had removed *of* because it is a common stopword. But we reinstate *of* in order to facilitate the award names that contains it.

School: A list of high schools was created from Wikipedia's "list of" pages. The compiled list contains 25271 entries. Any entry that is found in this list is marked as an occurrence of a school in the given text. The word frequency analysis shows the most frequent words that appear in names of schools to be: School (18618), High (16112), Academy (1828), Christian (1651), HS (1469), Central (684), and Senior (640). First letter capitalized sequences that contain those high frequent words are also marked as schools. However, the confidence score assigned to such partial matches are lower than that for complete entries in the list. Confidence scores are experimentally determined by manual supervision.

Major: We prepared a list of majors by referring to fields of studies offered by some top universities. The compiled list contains 318 entries. Any entry that appear in this list is marked as an occurrence of a major with a high confidence score. The most frequent words that appear in this list are: Studies (33), Engineering (24), Science (22), Management (17), and Education (13). We assign low confidence scores to first letter capitalized continuous word sequences that contain those words.

Degree: A list of degrees was compiled manually using Wikipeida. Our list contains 175 entries. We have both acronym versions of degrees (e.g., M.Eng) and the corresponding full forms (e.g., Masters of Engineering). The word frequency analysis reveals the most frequent words that appear in degree names to be: of (57), Doctor (31), Master (15), Bachelor (12), Administration (8), Science (8), Engineer (8), Medicine (7), degree (7), in (6), Business (5), and Licentiate (5). We ignore common stop words such as *of* and *in* because they are ambiguous and can occur in other contexts other than degrees. If a first letter capitalized continuous sequence of words contain any one or more of those high frequency words then we increase the confidence score assigned to that sequence being a name of a degree. However, the confidence scores assigned in word frequency analysis are lower than the confidence scores assigned when an entire entry in the list get matches in the text. The exact values of the confidence scores are adjusted manually.

Mentor: The value set for the attribute mentor contains only person names. Therefore, we used all person names given by the named entity recognition tool as potential candidates for the attribute mentor if they appear near some cue phrases such as *studied with*, *worked with*, *coach*, *trainer*, *adviser*, *mentor*, *supervisor*, and *spiritual adviser*. We checked the local contexts of the attribute mentor in WePS training data to determine the above mentioned cue phrases. Concretely, we extract a pre-defined window of text from all occurrences of the attribute mentor in WePS training data and the manually go through these contexts to identify the cue phrases. Once person names are marked as candidates for mentors, we then select the candidate that is closest to any of the variants of the given person name as the mentor for that person.

Location: We used the location annotation provided by the named entity recognition tool to mark potential candidates for this attribute. We then select the mention of location that is closest to any of the variants of the given name.

Nationality: We prepared a list of nationalities. The list contains 442 entries. It has multiple entries for certain nationalities (i.e. both *Englishmen* and *British* are marked for United Kingdom). Entries found in this list are marked as nationalities in the text. We then select the nationality tag that is closest to any variant of the given person name in the text as the correct nationality of the person under consideration.

Relatives: The set of values that the attribute *relatives* can take consists of person names. We mark all person names annotated by the named entity recognition tool as candidates of relatives of the person under consideration if a set of cue phrases that indicate various relationships exist in the immediate context of the candidate. We select a window of 10 words around the candidate as its immediate context. Cue phrases are selected from pages describing relationships in Wikipedia. It contains the following entries: "spouse", "brother", "sister", "wife", "husband", "father", "mother", "married","son", "daughter", "late husband", "late wife", "widow", "grand father", "grand mother", "aunt", "uncle", "step father", "step mother", "brother-in-law", "sister-in-law", "son-in-law", "nees", "nephew", "father-in-law", "mother-in-law", "child", "children", "sibling", "parent", "meet", "met", "girl friend", "boy friend", "finance", "ex-girlfriend", "ex-boyfriend", "ex-husband", "ex-wife".

Phone and Fax: We use the following regular expression to mark strings that are likely to be telephone numbers or a fax numbers.

```
(((\+\d{1,3}(-| )?\(?\d\)?(-| )?\d{1,5})|
(\(?\d{2,6}\)?))(-| )?(\d{3,4})(-| )
?(\d{4})(( x| ext)\d{1,5}){0,1})
```

We then mark those candidate strings as a telephone numbers if the cue phrases *tel*, *telephone*, *phone*, *mobile* occur in the immediate context of the candidates. We set a window of 3 words as the immediate context of a candidate. Likewise, a candidate string is marked as a fax number if the cue phrase *fax* occur in its immediate context. We then select the closest candidate to any variant of the given name as the correct attribute value for the person under consideration.

Email: E-mail addresses are marked using the following regular expression.

```
([\w\-\.]+@(\w[\w\-]+\.)+[\w\-]+)
```

We use a stop list for e-mail addresses that occur frequently on web documents such as webmaster@domain or support@domain. This exclusion list of e-mail addresses is compiled manually. Moreover, we found that people have a tendency to include a substring of their first or last names (or both) in their e-mail addresses. Therefore, we increase the confidence of an extracted candidate string if it satisfies those conditions. Finally, the e-mail address candidate that is closest to any of the variants of the given name is selected as the e-mail address of the person under consideration.

Table 1: Overall results for participating systems.

Rank	System	Precision	Recall	F-score
1	PolyUHK	30.4	7.6	12.2
2	CASIANED	8.5	19.0	11.7
3	ECNU_2	8.0	17.6	11.0
4	ECNU_1	6.8	18.8	10.0
5	MIVTU	5.7	15.5	8.3
6	UvA_2	4.4	27.4	7.6
7	UvA_1	2.7	27.3	5.0
8	UC3M_5	8.0	3.6	5.0
9	UvA_5	3.3	2.8	3.1
10	UC3M_1	2.5	2.2	2.3
11	UC3M_2	2.4	2.2	2.3
12	UC3M_3	2.2	2.0	2.1
13	UC3M_4	2.2	2.0	2.1
14	UvA_3	0.7	0.2	0.2
15	UvA_5	0.2	0.0	0.0

Web site: We use the following regular expression to extract urls.

```
(https?://([-\w\.]+)+(:\d+)?
(/([\w/_\.]*(\?\S+)?)?)?)
```

We then select the url that is closest to any of the variants of the given name as the correct web url for the person under consideration.

4. RESULTS AND DISCUSSION

The proposed attribute extraction system is evaluated on the test dataset created for the second WePS workshop. This dataset contains 3468 web documents retrieved for 30 people names. The average number of documents per name is 115.6. Out of those documents 585 were ignored during the annotation process and only the remaining 2883 were used for testing. 2421 documents in the test dataset have at least on attribute value. There were 462 documents without any attribute values. For further details of the annotation process and datasets refer [6]. Each participating system is evaluated by comparing the attributes produced by that system for a particular name against the gold standard attributes created by the annotators. Comparisons are done using precision, recall and F-score. Those evaluation metrics are computed for each individual name in the test dataset as well as for the overall set of attributes extracted by each system. Evaluation metrics are computed using following formulas,

$$\text{precision} = \frac{\text{no. of correctly identified attribute values by system}}{\text{no. of attribute values produced by the system}},$$

$$\text{recall} = \frac{\text{no. of correctly identified attribute values by system}}{\text{no. of attribute values in gold data}},$$

$$F - \text{score} = \frac{2 \times \text{recall} \times \text{precision}}{\text{recall} + \text{precision}}.$$

Table 2: Performance of MIVTU system by each name.

Name	Matches	Over generations	Misses	Precision	Recall	F-score
Benjamin Snyder	42	771	268	5.166	13.548	7.480
Hao Zang	66	747	279	8.118	19.130	11.399
Amanda Lentz	33	1185	279	2.709	10.577	4.314
Otis Lee	26	495	304	4.990	7.879	6.110
Bertram Brooker	56	1247	380	4.298	12.844	6.440
Jason Hart	174	1015	592	14.634	22.715	17.801

Table 1 summarizes the results produced by each participating system in the attribute extraction task at the second WePS. We have arranged the systems according to their overall F-scores. Results shown for UC3M_4 and UC3M_5 are for unofficial runs submitted after the results submission deadline. The proposed system is ranked 5^{th} among the 15 systems shown in Table 1. It reports an overall F-score of 8.3. The best performing system is PolyUHK. It has an F-score of 12.2. Overall, all the systems report low F-scores. The highest precision reported by any individual system is 30.4 (PolyUHK) and the highest recall reported by any individual system is 19.0 (CASIANED). This fact suggests that the task of attribute extraction is indeed challenging and none of the systems successfully overcome the difficulties described in section 1. Among the 18 attributes considered in the task, MIVTU system reported the highest recall for three attributes: date of birth (32.0), birth place (48.5), and affiliation (23.0). All systems had difficulties in extracting the attributes major, mentor and award.

Table 2 shows the performance of the proposed (MIVTU) system per each name used in the evaluations. Best results are reported for *Jason Hart*. From Table 2 we see that MIVTU system generally has better recall values compared to precision values. This is a side effect of it using various lists and over generating candidates. A more conservative approach to tagging candidates might help to overcome this problem. During our experiments we noticed that the named entity tagger itself has a tendency to mark first letter capitalized consecutive sequences of words as named entities even when they were not.

To improve the accuracy of attribute extraction we must improve both steps: marking candidate attributes in text and finding which attribute values are relevant to the person under consideration. The list-based approach that we used to find candidate attribute values has several limitations. First, one cannot enumerate all attribute values using lists. Attributes such as nationalities can be listed up because the number of countries is a closed set. However, attributes such as awards, occupations, birth place, location, affiliation, school and mentor are typical examples of open sets. In addition to using pre-compiled lists, we must have some form of rules to identify such attributes. Moreover, the use of lists can also introduce a level of ambiguity because an entry in a list can appear in a different context in the text. For example, some entries in a list of schools can also be valid entries for affiliation of a person if that person is actually employed in that school. The second step of determining which attribute values are relevant to the person under consideration could be improved if we can merge results from different documents. For example, an attribute such as birth date or birth place should be the same even if it is extracted from different documents for the same person. By enforcing such constraints we might be able to reduce the number of candidate attribute values and thereby make an accurate decision. However, the task guidelines in WePS does not allow this form of cross-document information integration because the objective is to use the set of extracted attributes to cluster the pages. Therefore, the attribute extraction systems must process each document separately.

5. CONCLUSION

We proposed an attribute extraction method to extract the relevant values of a pre-defined set of attributes from a document related to a person. The proposed method consists of two steps: annotating the given document with numerous candidate attribute values, and then selecting the attribute values relevant to the person under consideration. The proposed method was evaluated using the test dataset created for the attribute extraction sub-task at the second Web People Search Task. The proposed method obtained an F-score of 8.3 and was ranked 5^{th} among the 15 systems participated in the task. The results are preliminary and various challenges that must be addressed in order to further improve the performance of attribute extraction were discovered. In future, we plan to improve the proposed attribute extraction method on those lines.

6. REFERENCES

[1] Javier Artiles, Julio Gonzalno, and Satoshi Seline. Weps 2 evaluation campaign: overview of the web people search clustering task. In *proc. of the 2nd Web People Search Evaluation Workshop (WePS 2009) at 18th International World Wide Web Conference*, 2009.

[2] Javier Artiles, Julio Gonzalo, and Satoshi Sekine. The semeval-2007 weps evaluation: Establishing a benchmark for the web people search task. In *proc. of the SemEval at Annual Meeting of the Association for Computational Linguistics*, 2007.

[3] Kedar Bellare, Partha Pratim Talukdar, Giridhar Kumaran, Fernando Pereira, Mark Liberman, Andrew McCallum, and Mark Dredze. Lightly-supervised attribute extraction for web search. In *proc. of NIPS 2007 Workshop on Machine Learning for Web Search*, 2007.

[4] Danushka Bollegala, Taiki Honma, Yutaka Matsuo, and Mitsuru Ishizuka. Automatically extracting personal name aliases from the web. In *proc. of the 6th International Conference on Natural Language Processing (GoTAL 08), Advances in Natural Language Processing Springer LNCS 5221*, pages 77–88, 2008.

[5] M. Pasca, D. Lin, J. Bigham, A. Lifchits, and A. Jain. Organizing and searching the world wide web of facts - step one: the one-million fact extraction challenge. In *proc. of the 21st National Conference on Artificial Intelligence (AAAI-06)*, pages 1400 – 1405, 2006.

[6] Satoshi Sekine and Javier Artiles. Weps 2 evaluation campaign: overview of the web people search attribute extraction task. In *proc. of the 2nd Web People Search Evaluation Workshop (WePS 2009) at 18th International World Wide Web Conference*, 2009.