Four-Factor Ranking Model on Building Oc-C-Gen: Occupation Chronology Generator

Shirley Anugrah Hayati and **Mirna Adriani**
Faculty of Computer Science, Universitas Indonesia
Depok, West Java, Indonesia
`shirley.anugrah@ui.ac.id, mirna@cs.ui.ac.id`

Abstract

`Oc-C-Gen` attempts to generate an occupation chronology given a person name as a query. This study exploits Indonesian news articles as data set and combines the task of temporal information extraction and automatic summarization. We present a new method for extracting and ranking occupation candidates before the final occupation chronology is built. Based on statistics, four important factors for ranking occupation candidates are identified. To evaluate the performance of our approach, we use a collection of ten queries and score extracted candidates with the ranking model we have proposed. For ranking extracted occupation candidates, we use single factor and two factors for ranking. Our approach achieves an F_1 score of 75.6%.

1 Introduction

Information extraction is the process of gaining and converting information from unstructured text into structured data (Jurafsky and Martin, 2009). As a subtask of information extraction, temporal information extraction focuses on identifying event-time or event-event relation. Previous researches about temporal information extraction have been conducted (Ling and Weld, 2010; UzZaman and Allen, 2010; Zavarella and Tanev, 2013), including to tackle the notable TempEval challenge (UzZaman et al., 2013). TempEval challenge addressed the problem of extracting time on text or identifying which event (usually verbs in English) appear before, after, or simultaneously with other time/event.

Meanwhile, timeline summarization is quite related to temporal information extraction since timeline summarization requires temporal information in creating summaries of ordered events. Automatic timeline summarization based on news articles have been extensively studied by NLP researchers (Yan et al., 2011; Nguyen et al., 2014; Tran et al., 2015; Li and Li, 2013). They exploit news dataset to automatically generate a timeline of events, not a person's biographic facts, especially occupation. Occupation is an important aspect of a person's biographic facts which can tell much about a person's historical record. Thus, we present our framework in generating occupation chronology named Oc-C-Gen (Occupation Chronology Generator). Our goal is to automatically generate a person's occupation chronology from news articles based on user-generated query.

An occupation chronology is described as a chronological overview of a person's occupation in which each occupation is arranged based on temporal order. The following shows an example of a part of occupation chronology for query "Joko Widodo".

- Mayor of Surakarta
- Governor of Jakarta
- President of Indonesia

Since we focus on occupation chronology, the content granularity is an occupation phrase, not sentence. For this study, we use Indonesian news data. After user input query, occupation candidates are extracted from the text before building the timeline. Based on statistics, we propose four factors which strongly signifies an occupation. The first one is frequency of occupation unit (phrases or words) extracted from texts. The second factor is the number of capitals in the beginning of occupation. The third factor is phrase length of the occupation, and the final factor is the number of various extraction techniques applied to obtain occupation candidate from texts. These factors are used for ranking occupation candidates before they are selected into occupation chronology.

Our task is quite related to temporal information extraction as well as biographical timeline summarization since we build a query-based, disfluent summary of occupation from a pile of documents (Hovy and Lin, 1998). Several studies related to person's biographical fact summarization or extraction barely touches on the aspect of arranging events or occupation based on chronological order. (Zhou et al., 2004) presented a work on summarizing biographical documents. (Schiffman et al., 2001) created biographical summary from news corpus but specified neither occupation nor chronological order. (Garera and Yarowsky, 2009) extracted biographical facts from encylopaedic pages using pattern-based model, contextual model, and implicit models. The existing research that have come closest to our goal is TimeMachine (Althoff et al., 2015) that generates timeline for all pertinent events, including the occupation of the subject in query. They do not extract such events from documents as a data set, but instead from a knowledge base.

Although our problem also tackles the task of information extraction, we concern more about the big picture of arranging and compressing extracted information into a short chronological occupational summary than labelling correct occupation in a sentence. As mentioned in (Jeong et al., 2015), previous methods produce satisfying outcomes, but most are only suited to their language because they have advanced language resources and tools. Meanwhile, similar resources are not available yet in Indonesian. Therefore, our task needs a different approach from previous temporal information extraction method.

In summary, the main contributions of this work are as follows. This research is the first on building person's occupation chronology from news data set. We investigate phrases surrounding occupation phrases and design a novel framework for occupation candidate extraction. We also propose a new method for ranking occupation candidate.

2 Building Occupation Chronology

We define our problem as follows:

Input: Given a query Q (person name), we obtain a collection of documents related to query.

Output: A chronology C of the queried person's occupation, based on the obtained query-related documents, will be generated. If pertinent information about the subject of query's employer and his/her job capacity is present in the collected documents, said information will be also incorporated in the resultant occupational chronology.

Our methodology consists of three main phases: preprocessing, occupation candidate extraction, and occupation chronology generation. Figure 1 shows an overview of how Oc-C-Gen generates occupation chronology from news data.

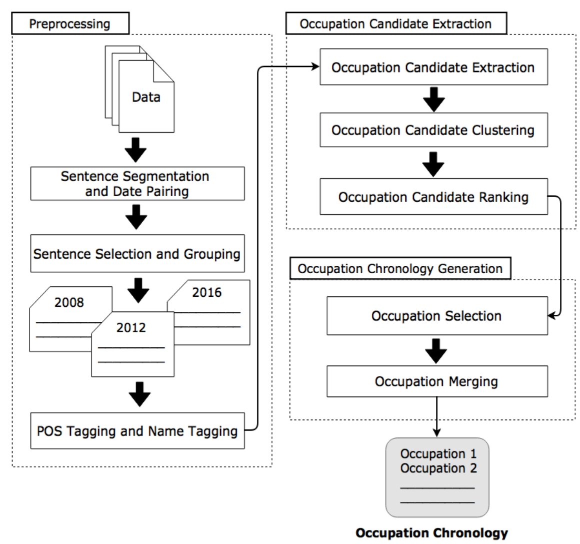

Figure 1: Process of Occupation Chronology Generation

In the preprocessing stage, after user uploads news data set and inputs person name as a query, our methodology segments each news document into sentences which are paired with document publication date if there is no date mentioned in the document. Then sentences which contain the person name query are selected. Irrelevant tokens, such as writer's name and opening phrase, are removed. These sentences are grouped according to the year of document publication date. We then apply part of speech (POS) tags and person named entity tags to the sentences. POS tags and named entity tags are used when we extract occupation candidates. The output of this first stage is a collection of query-related and tagged sentences grouped based on their date.

Occupation candidates are extracted from sentences on each year-based group. Extracted candidates are clustered in order to remove duplication. Unique occupation candidate clusters are ranked based on our proposed ranking model in order to be selected in the next stage, which is occupation chronology generation stage. We will explain more about processes in occupation candidate extraction stage in section 2.1.

In the final stage, top three occupation candidates are selected to represent each occupation that the query person does in a certain year. After that, occupations that are obtained from consecutive years are merged to make sure that the chronology will not have duplicated occupation.

2.1 Occupation Candidate Extraction

Currently, there is no extant system for extracting occupational data from Indonesian text. Therefore, we have designed a new approach in extracting occupation candidates. It is important to note that our task is not limited to specifically extract the person's occupational title, but rather to obtain text unit (words or phrases) which most likely contains information pertinent to the query subject's occupation. These extracted words or phrases are called **occupation candidates**. A total of 9,682 documents from 5 query people are development data to build a statistical database for building occupation extraction method and proposed scoring model.

An occupation can be located before a person name or after a person name. An example of occupation located before a person name is as follow: "Facebook Inc. CEO Mark Zuckerberg have a scheduled meeting with several politicians". If the query is "Mark Zuckerberg", his occupation, "Facebook Inc. CEO", is positioned exactly before his name. On the other hand, in the text "Mark Zuckerberg is Facebook Inc. CEO", query person Mark Zuckerberg's occupation (Facebook Inc. CEO) is located after his name.

In a sentence, distance between an occupational text unit *O* and query person *Q* is defined as follow:

$$Distance(O,Q) = ||min(pos(O) - pos(Q))||$$

where $pos(O)$ is position of occupational text unit in the sentence, $pos(Q)$ is position of query in the sentence, and $min(pos(O) - pos(Q))$ is the minimal of $pos(O) - pos(Q)$.

For example, if "Mark Zuckerberg is Facebook Inc. CEO" is our sentence with "Mark Zuckerberg" as Q and "Facebook Inc. CEO" as O, distance between "Mark Zuckerberg" and "Facebook Inc. CEO" equals 2 because between the query and the occupational text, there is a word "is".

From statistical data, we discover that from all occupation text units which are located before a query person's name, 4480 occupational text units appear exactly before the query (distance = 1), and 317 appears in distance = 2. We further analyze that for occupational text units which are 1-token apart from the query (in other word, distance = 2), the token which separates the query and the occupational text unit is a comma.

For occupational text units which are located after a query person's name, 220 occupational text units are 1-word apart from the query (distance = 2), 102 occupational text units are 2-word apart from the query (distance = 3). We find out that "sebagai (*as*)", "selaku (*as*)", "menjadi (*to become*)", "merupakan (*is*)", and "menjabat (*to officiate as*)" are the most common words to appear before occupational text units.

Therefore, we conclude that there are five ways for occupation candidate extraction.

1. Occupation candidates located exactly before a person name
   For example, in the following snippet "Presiden Indonesia Joko Widodo (*President of Indonesia Joko Widodo*)", the occupation "Presiden Indonesia" is located exactly before the name "Joko Widodo".

2. Occupation candidates positioned before a comma and a person name
   An example for this second case is that in the following snippet, "Presiden Indonesia, Joko Widodo, (*President of Indonesia, Joko*

No	Sentence
S1	**Wakil Presiden Indonesia**, Jusuf Kalla, membuka open house di Istana Wapres.
	(***Indonesian Vice President** Jusuf Kalla held* an open house at Istana Wapres.)
S2	Survey ini dilakukan sebelum **Wapres** Jusuf Kalla menjadi **ketua umum Partai Golkar**.
	(*This survey was conducted before **Vice President** Jusuf Kalla became the **chairman of Golkar Party**.)*
S3	Bagi Kalla yang kini menjabat **wakil presiden** dan **ketua umum Partai Golkar**, koalisi itu sudah menjadi masa lalu.
	(*For Kalla, who now officiates as **vice president** and **chairman of Golkar Party**, the coalition was in the past.*)

Table 1: Sentences for illustration

Widodo,)", the occupation "Presiden Indonesia" is located exactly before a comma and "Joko Widodo".

3. Occupation candidates after a person name that follows the cue words "merupakan" ("*is*") or "menjadi" ("*to become*")
   Given a sentence "Joko Widodo merupakan Presiden Indonesia (*Joko Widodo is President of Indonesia*)", we can obtain occupation candidate "Presiden Indonesia".

4. Occupation candidates following the word "sebagai" or "selaku" ("*as*")
   To illustrate this fourth way, consider the following sentence: "Joko Widodo sebagai Presiden Indonesia mengumumkan kebijakan baru (*Joko Widodo as President of Indonesia announced a new policy*)". The extracted occupation candidate is "Presiden Indonesia" which comes after the word "sebagai (*as*)".

5. Occupations after the word "menjabat" ("*to officiate as*")
   If we have the sentence "Joko Widodo menjabat Presiden Indonesia (*Joko Widodo officiates as President of Indonesia*)", we get occupation candidate "Presiden Indonesia" after the word "menjabat *(officiate as)*".

For each sentence, occupation candidates are extracted by those five techniques. In capturing occupation candidates, we iterate through nouns, adjectives, and numbers present in the now POS-tagged sentence, only stopping if punctuation mark, date, year, month, unrelated person name, or title abbreviation (e.g. "Bu (*Mrs.*)" or "Pak (*Mr.*)") is encountered.

The discovered occupation candidates are then clustered based on similarity to minimize duplication. An occupation cluster contains one or more occupation candidates. Similarity between two candidates are measured using Jaccard similarity coefficient. Note that a **candidate** is a text unit which is made up from one or more words.

$$Sim(W_1, W_2) = |W_1 \cap W_2| / |W_1 \cup W_2|$$

W_1 is a set of words which construct the first candidate $cand_1$ and W_2 is a set of words that make up the second candidate $cand_2$. In other word, Jaccard similarity computes the amount of overlapping words between $cand_1$ and $cand_2$.

If the Jaccard similarity value of a new occupation candidate and an existing cluster is higher than our defined threshold θ, the candidate is added into the cluster. Otherwise it becomes a new occupation cluster. In internal experiments, we observe that the value θ = 0.5 provides the best result, so it is chosen as our threshold value.

As a new candidate is added to a cluster, the words that make up the occupation candidate is normalized using our Indonesian Wiktionary-based[1] abbreviation dictionary. Each occupation cluster is represented by a keyword, which is the most frequent occupation candidate in the cluster. If a new occupation candidate that does not satisfy the threshold θ is found, it is then compared to the keyword. If every word in the new candidate appears in the keyword with the correct order, the candidate joins the cluster. This filter is incorporated to minimize duplication issues that arise from encountering two relatively similar occupations, such as "kiper" (*"goalkeeper"*) and "kiper Arsenal" ("Arsenal *goalkeeper*").

[1] https://id.wiktionary.org/wiki/Wiktionary:Daftar_singkatan_dan_akronim_bahasa_Indonesia

For illustration, we shall define "Jusuf Kalla" as our query. Three sentences in 2004 are retrieved as shown in Table 1 using five extraction techniques described previously. Table 2 presents the extracted candidate occupations and technique used for this extraction. The phrases "Wakil Presiden Indonesia (*Indonesian Vice President*)", "Wapres (*VP*)", and "wakil presiden (*vice president*)" are merged into one cluster after each abbreviation is normalized. "Wapres" is normalized to "wakil presiden" in order to remove the otherwise duplicated cluster. The keyword for this cluster is "Wakil Presiden Indonesia" since "Wakil Presiden Indonesia" and "wakil presiden" are merged. In this example, we also obtain another cluster of "ketua umum Partai Golkar (*chairman of Golkar Party*)".

Sentence	Occupation Candidate	Technique
S1	Wakil Presiden Indonesia	2
S2	Wapres, ketua Partai Golkar	1, 3
S3	wakil presiden, ketua Partai Golkar	5, 5

Table 2: Extracted occupation candidates

2.2 Occupation Candidate Ranking

Given an occupation cluster o and a query q in year Y_n, our ranking model pays attention to four factors that contribute to ranking of the most salient occupation candidates.

- **Frequency** $f(o|q)$**:** We observe that the more frequently an occupation candidate is extracted from sentences in the same year, the more salient it is the real occupation done by the query person in that year. Thus, frequency becomes our first factor. For example, in 2015, the most frequently extracted occupation candidate for query "Jusuf Kalla" is "Vice President", so the most salient occupation of Kalla in 2015 is vice president. Defining f_{max} as the maximum frequency in Y_n, $f(o|q)$ becomes the normalized value of cluster frequency f_o obtained by dividing f_o with f_{max}.

- **Capitalization** $c(o|q)$**:** The second factor is the number of words which is started with capital. Capitalized first letters of words contained *within* a phrase suggest that the phrase contains an organization or institution that employ the subject of the query. Note that we do not take into account capital letters on the first word. If $c_{old}(o|q)$ is equal to the number of words with capitalized first letter (discounting the capitalization of any sentence's first word), $|w|$ is the number of words constructing keyword of o, and $|w_{new}| = |w| - 1$. This results in $c(o|q) = c_{old}(o|q)/|w_{new}|$ as long as $|w| > 0$, with $c(o|q) = 0$ otherwise.

- **Phrase Length Constraint** $p(o|q)$**:** Based on statistics that we have obtained from our development data, occupations are mostly formed using 2 or 5 words. Thus, we define the third factor is justification whether a keyword of an occupation cluster follows the same pattern. $p(o|q) = 1$ of the keyword of o contains 2 to 5 words, otherwise 0.

- **Various Extraction Techniques** $t(o|q)$**:** As mentioned before, there are five different techniques for extracting occupation candidate. Heuristically, if an occupation is extracted by various approaches as opposed to just one, there is a higher chance that the extracted occupation matches the query subject's actual occupation. $t(o|q)$ is the number of different extraction techniques used to get occupation candidate from a text divided by 5 so that $t(o|q)$ is in the range of [0,1]. Using previous example in Table 2, we obtain occupation candidate "Wakil Presiden Indonesia" by using three different techniques, which are technique number 2, 1, and 5. Meanwhile, occupation candidate "ketua Partai Golkar" is extracted by using two different techniques (3 and 5). Therefore, "Wakil Presiden Indonesia" has higher fourth factor score than the score of "ketua Partai Golkar" because there are more techniques to extract "Wakil Presiden Indonesia" than "ketua Partai Golkar".

Overall, a score of an occupation cluster o given the query q is defined as the following:

$$Score(o|q) = \alpha \times f(o|q) + \beta \times c(o|q) + \gamma \times p(o|q) + \delta \times t(o|q)$$

where

- α is coefficient for frequency factor
- β is coefficient for capitalization factor
- γ is coefficient for phrase length factor
- δ is coefficient for capitalization factor various extraction technique factor

Values of all coefficients, factors, and $Score(o|q)$ are in range [0, 1].

Each occupation cluster is ranked based on its score. We select the top 2 occupation cluster with score greater than 0.25. The keywords of selected clusters represent the occupations held by the queried person in the year Y_n. Once again we merge occupation held in later year to identical occupation in previous years to remove duplication. For example, a query person "Jusuf Kalla" is a "vice president" in 2004 and in 2005. Thus, the "vice president" in 2005 is merged with the "vice president" in 2004.

3 Experiments

3.1 Data set

Data are collected from Indonesian news sites, detikcom[2] and Kompas.com[3] spanning a period between 2004 and 2015. Our data set consists of 57,344 news articles related to 15 query people (politicians, athletes, and businessmen). 5 queries are used for development data, and the other 10 are used for testing data.

3.2 Experiment Set-up

We conduct three sets of experiment. The first experiment set only uses one factor in ranking model. The second and third experiment sets incorporate two factors, but the third experiment have one factor have greater weight than the other. We call a factor which has a greater weight than the other as a dominant factor.

Table 3 shows details of coefficients used for each experiment scenario. Active factor means that such factor have non-zero coefficient. α, β, γ, and δ are coefficients related to frequency factor, capitalization factor, phrase length factor, and various technique extraction factor respectively.

Experiment Scenario I				
Active Factor	α	β	γ	δ
F	1	0	0	0
C	0	1	0	0
P	0	0	1	0
T	0	0	0	1
Experiment Scenario II				
Active Factor	α	β	γ	δ
F-C	0.5	0.5	0	0
F-P	0.5	0	0.5	0
F-T	0.5	0	0	0.5
C-P	0	0.5	0.5	0
C-T	0	0.5	0	0.5
P-T	0	0	0.5	0.5
Experiment Scenario III				
Active Factor	α	β	γ	δ
F-C	0.75	0.25	0	0
F-P	0.75	0	0.25	0
F-T	0.75	0	0	0.25
C-F	0.25	0.75	0.5	0
C-P	0	0.75	0.25	0
C-T	0	0.75	0	0.25
P-F	0.25	0	0.75	0.5
P-C	0	0.25	0.75	0
P-T	0	0	0.75	0.25
T-F	0.25	0	0	0.75
T-F	0	0.25	0	0.75
T-F	0	0	0.25	0.75

Table 3: Coefficient variations for experiment scenario where F = frequency factor, C = capitalization factor, P = phrase length factor, T = various technique factor

1. **One-Factor Experiment Set**
   The purpose of this experiment is to evaluate the performance of each factor. Therefore, we set up only one active factor with a value of 1 while assigning a value of 0 to the remaining factors. For instance, to activate the frequency factor, we set $\alpha = 1$ and $\beta = \gamma = \delta = 0$. There are 4 experiments conducted in this first set.

2. **Balanced Two-Factor Experiment Set**
   In this second experiment set, we test combi-

[2] http://www.detik.com
[3] http://www.kompas.com

nations of two factors to compare its performance with single-factor model. We call this experiment as balanced two-factor experiment because we assign 0.5 as weight for two active factors and assign 0 to the remaining factors. 6 experiments are run to represent all combinations of balanced two factors.

3. **Two-Factor Experiment with One Dominant Factor Set**
   For our third experiment set, we attempt to increase the weight of one factor and lower the weight of other factor. The dominant factor's weight is 0.75 and the other factor's weight is 0.25. We administer 12 experiments (3 experiments for 4 different dominant factor) to investigate the best factor combination if we let one factor to be more dominant than the other.

3.3 Evaluation Metric

For evaluation, we calculate the average precision, average recall, and average F_1 score (Steinberger and Jezek, 2009) by comparing the output of our program with gold standard. In this case, gold standard is a person's occupation chronology done by human. Our gold standard is built from the same data set. Initially, we calculate precision P_n, recall R_n, and F_1 score F_{1n} for the result of each query person q_n. Precision, recall, and F_1 score is defined as follows.

$$\mathit{Precision} = \frac{|\mathit{Program\ Output} \cap \mathit{Gold\ Standard}|}{|\mathit{Program\ Output}|}$$

$$\mathit{Recall} = \frac{|\mathit{Program\ Output} \cap \mathit{Gold\ Standard}|}{|\mathit{Gold\ Standard}|}$$

$$\mathit{F1\text{-}score} = 2 \times \frac{Precision \times Recall}{Precision + Recall}$$

Then, average precision is defined as follows.

$$Averageprecision = \sum_{q_n} \frac{P_n}{Q}$$

where Q is the number of queries.

For our case, $Q = 10$ since we have 10 experiment queries. Average recall and average F_1 score are calculated by the same formula but P_n is replaced with R_n or F_{1n}.

3.4 Experiment Results

The result of single-factor experimental scenario shows that the highest precision (71.8%) and F_1 score (69.5%) are achieved when the frequency factor is used. This reveals that frequency is an important contributive factor for ranking occupation candidate. Furthermore, if capitalization is used as the only factor, we obtain the highest recall (88.4%) but very low precision (31.7% lower than the highest precision) as presented in Table 4. Based on this result, we want to know what if we set certain factor as dominant factor as in the third experiment set.

Factor	Precision	Recall	F_1
F	**71.8**	78.4	**69.5**
C	40.4	**88.4**	52.6
P	58.9	82.6	65
T	65.9	75.7	69.3

Table 4: Result of single-factor experiment where F = frequency factor, C = capitalization factor, P = phrase length factor, T = various technique factor

For the balanced two-factor experiment, a combination of frequency and technique variation shows a large improvement for precision, recall, and F_1 score compared to the best single-factor experiment result. This result is not very surprising as the single-factor result already shows that technique variation factor produces the second best F_1 score. We examine in Table 5 that capitalization is not a good partner factor, resulting in low precision regardless which approach it is paired with.

Factor	Precision	Recall	F_1
F-C	47.4	79.3	58.4
F-P	64.1	89.1	70.8
F-T	**76.2**	**90.7**	**74.6**
C-P	40.1	85.4	51.7
C-T	53.9	78.6	60.2
P-T	62.1	82.4	69.5

Table 5: Result of balanced two-factor experiments; F, C, P, T are defined in Table 3 caption; F-C means frequency and capitalization factors are used; the rest abbrv.s follow the same pattern

Table 6 demonstrates the result of the third experiment set. As we make one factor more dominant over the other, we see that the performance increases, especially when frequency becomes dominant factor. Precision rises 6% from the second experiment precision and the F1-score becomes 75.6%. Further observation leads that setting frequency factor as dominant factor produces the best precision, recall, and F1-score. This result agrees with the result of first experiment set which shows that frequency is a good factor partner to produce high precision and F1-score. It is understandable that the more frequently an occupation is connected with a person, the more possible that occupation is the person's real occupation.

Factor	**Precision**	**Recall**	F_1
F-C	65.3	84.3	69.0
F-P	64.1	**90.7**	70.8
F-T	**82.6**	78.4	**75.6**
C-F	42.8	87.1	54.0
C-P	39.7	85.4	51.3
C-T	43.5	85.1	53.9
P-F	64.1	**90.7**	70.8
P-C	40.1	85.4	51.7
P-T	62.1	82.4	69.5
T-F	69.4	80.9	72.9
T-C	55.6	78.9	61.8
T-P	68.8	79.3	72.6

Table 6: Result of two-attribute experiments with dominant factor; F, C, P, T are defined in Table 3 caption; The first letter means that the factor weight is 0.75 (dominant factor). F-C means that frequency factor becomes the dominant factor and capitalization factor; the rest abbrv.s follow the same pattern

4 Conclusion and Future Work

We have introduced a novel method for extracting and ranking occupation candidates to build occupation chronology from Indonesian news dataset. Experiments show that using a combination of two factors with one dominant results better in performance over using a single factor in the scoring model. As we make one factor more dominant, the performance even improves. The highest precision achieved is 82.6% when frequency becomes dominant factor paired with various technique factor whereas the highest recall is 90.7% when frequency is combined with phrase length factor.

Relative to the best F_1 score of single-factor model, the F_1 score improvement of two-factor model with dominant factor is about 6%. In the future, we plan to explore more on three or four-factor scoring.

References

Tim Althoff, Xin Luna Dong, Kevin Murphy, Safa Alai, Van Dang, and Wei Zhang. 2015. Timemachine: Timeline generation for knowledge-base entities. In Longbing Cao, Chengqi Zhang, Thorsten Joachims, Geoffrey I. Webb, Dragos D. Margineantu, and Graham Williams, editors, *KDD*, pages 19–28. ACM.

N. Garera and D. Yarowsky. 2009. Structural, transitive and latent models for biographic fact extraction. In *Proceedings of the Conference of the European Chapter of the Association of Computational Linguistics*, pages 300–308.

Eduard Hovy and Chin-Yew Lin. 1998. Automated text summarization and the summarist system. In *Proceedings of a Workshop on Held at Baltimore, Maryland: October 13-15, 1998*, TIPSTER '98, pages 197–214, Stroudsburg, PA, USA. Association for Computational Linguistics.

Young-Seob Jeong, Zae Myung Kim, Hyun-Woo Do, Chae-Gyun Lim, and Ho-Jin Choi. 2015. Temporal information extraction from korean texts. In *Proceedings of the Nineteenth Conference on Computational Natural Language Learning*, pages 279–288, Beijing, China, July. Association for Computational Linguistics.

Daniel Jurafsky and James H. Martin. 2009. *Speech and Language Processing (2Nd Edition)*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA.

Jiwei Li and Sujian Li. 2013. Evolutionary hierarchical dirichlet process for timeline summarization. In *ACL (2)*, pages 556–560. The Association for Computer Linguistics.

Xiao Ling and Daniel S Weld. 2010. Temporal information extraction. In *AAAI*, volume 10, pages 1385–1390.

Kiem-Hieu Nguyen, Xavier Tannier, and Vronique Moriceau. 2014. Ranking Multidocument Event Descriptions for Building Thematic Timelines. In *Proceedings of the 30th International Conference on Computational Linguistics (Coling 14)*, Dublin, Ireland, August.

Barry Schiffman, Inderjeet Mani, and Kristian J. Concepcion. 2001. Producing biographical summaries: Combining linguistic knowledge with corpus statistics. In *Proceedings of the 39th Annual Meeting on*

Association for Computational Linguistics, ACL '01, pages 458–465, Stroudsburg, PA, USA. Association for Computational Linguistics.

Josef Steinberger and Karel Jezek. 2009. Evaluation measures for text summarization. *Computing and Informatics*, 28(2):251–275.

Giang Binh Tran, Mohammad Alrifai, and Eelco Herder. 2015. Timeline summarization from relevant headlines. In Allan Hanbury, Gabriella Kazai, Andreas Rauber, and Norbert Fuhr, editors, *ECIR*, volume 9022 of *Lecture Notes in Computer Science*, pages 245–256.

Naushad UzZaman and James Allen. 2010. Trips and trios system for tempeval-2: Extracting temporal information from text. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, pages 276–283, Uppsala, Sweden, July. Association for Computational Linguistics.

Naushad UzZaman, Hector Llorens, Leon Derczynski, James Allen, Marc Verhagen, and James Pustejovsky. 2013. Semeval-2013 task 1: Tempeval-3: Evaluating time expressions, events, and temporal relations. In *Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)*, pages 1–9, Atlanta, Georgia, USA, June. Association for Computational Linguistics.

Rui Yan, Xiaojun Wan, Jahna Otterbacher, Liang Kong, Xiaoming Li, and Yan Zhang. 2011. Evolutionary timeline summarization: A balanced optimization framework via iterative substitution. In *Proceedings of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '11, pages 745–754, New York, NY, USA. ACM.

Vanni Zavarella and Hristo Tanev. 2013. Fss-timex for tempeval-3: Extracting temporal information from text. In *Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)*, pages 58–63, Atlanta, Georgia, USA, June. Association for Computational Linguistics.

Liang Zhou, Miruna Ticrea, and Eduard Hovy. 2004. Multi-document biography summarization. In Dekang Lin and Dekai Wu, editors, *Proceedings of EMNLP 2004*, pages 434–441, Barcelona, Spain, July. Association for Computational Linguistics.