A New Surprise Measure for Extracting Interesting Relationships between Persons

Hidetaka Kamigaito[1], **Jingun Kwon**[1], **Young-In Song**[2] **and Manabu Okumura**[1]
[1]Tokyo Institute of Technology
[2]Naver Corporation
kwon.j.ad@m.titech.ac.jp
{kamigaito,oku}@lr.pi.titech.ac.jp
song.youngin@navercorp.com

Abstract

One way to enhance user engagement in search engines is to suggest interesting facts to the user. Although relationships between persons are important as a target for text mining, there are few effective approaches for extracting the interesting relationships between persons. We therefore propose a method for extracting interesting relationships between persons from natural language texts by focusing on their surprisingness. Our method first extracts all personal relationships from dependency trees for the texts and then calculates surprise scores for distributed representations of the extracted relationships in an unsupervised manner. The unique point of our method is that it does not require any labeled dataset with annotation for the surprising personal relationships. The results of the human evaluation show that the proposed method could extract more interesting relationships between persons from Japanese Wikipedia articles than a popularity-based baseline method. We demonstrate our proposed method as a chrome plugin on google search.

Ex.1: **Tim Burton** and **Johnny Depp**
When Tim Burton met Johnny Depp for the first time, he had the impression that Johnny Depp was a hopelessly poor actor.

Ex.2: **Chien-Ming Wang** and **Suzuki Ichiro**
Chien-Ming Wang, a major-leaguer from Taiwan, asked Suzuki Ichiro three autographs before the start of the game.

Ex.3: **Ringo Starr** and **Beatles' members**
In the film Yellow Submarine, after hearing about the crisis in Pepper Land, Ringo Starr, along with Beatles companions John Lennon, George Harrison, and Paul McCartney, went to the bottom of the sea in a Yellow Submarine to save Pepper Land.

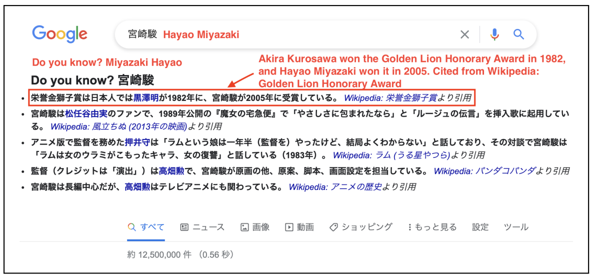
Figure 1: Example sentences that contain interesting relationships between persons.

1 Introduction

Interesting facts are useful information for a variety of important tasks. For example, in data mining, the interesting facts can enhance user engagement in search engines (Fatma et al., 2017). In natural language processing, the interesting facts can improve user experience with automatic conversation systems (Niina and Shimada, 2018). However, if we rely on experts to gather the interesting facts, the cost becomes quite high.

As a solution, several approaches have been developed to extract interesting facts automatically. Lin and Chalupsky (2003) proposed a set of unsupervised link discovery methods that can compute interestingness on graph data represented as a set of entities connected by a set of binary relations. Prakash et al. (2015) extracted interesting sentences about movie entities from Wikipedia articles and ordered them based on their interestingness by utilizing Rank-SVM, trained in a supervised manner. Tsurel et al. (2017) proposed an algorithm that automatically mines trivia facts from Wikipedia by utilizing its category structure. Their approach can rank categories for an entity based on their trivia quality induced from the categories. Fatma et al. (2017) proposed a method for automatically mining trivia facts for an entity of a given domain in knowledge graphs by utilizing deep convolutional neural networks, trained in a supervised manner. Korn et al. (2019) mined trivia facts from superlative tables in Wikipedia articles. Kwon et al. (2020) proposed a method to obtain sentences including trivia facts with utilizing paragraph structures in Wikipedia articles.

However, some of these approaches work only on structured datasets such as knowledge graphs or Wikipedia categories. In addition, while supervised approaches can work on unstructured natural language texts, the applicable domain is restricted due to the lack of annotated datasets. Hence, the current approaches for extracting interesting facts



宮崎駿 Hayao Miyazaki

Do you know? Miyazaki Hayao

Akira Kurosawa won the Golden Lion Honorary Award in 1982, and Hayao Miyazaki won it in 2005. Cited from Wikipedia: Golden Lion Honorary Award

Do you know? 宮崎駿

- 栄誉金獅子賞は日本人では黒澤明が1982年に、宮崎駿が2005年に受賞している。 *Wikipedia: 栄誉金獅子賞より引用*
- 宮崎駿は松任谷由実のファンで、1989年公開の『魔女の宅急便』で「やさしさに包まれたなら」と「ルージュの伝言」を挿入歌に起用している。 *Wikipedia: 風立ちぬ (2013年の映画)より引用*
- アニメ版で監督を務めた押井守は「ラムという娘は一年半（監督を）やったけど、結局よくわからない」と話しており、その対談で宮崎駿は「ラムは女のウラミがこもったキャラ、女の復讐」と話している（1983年）。 *Wikipedia: ラム (うる星やつら)より引用*
- 監督（クレジットは「演出」）は高畑勲で、宮崎駿が原画の他、原案、脚本、画面設定を担当している。 *Wikipedia: パンダコパンダより引用*
- 宮崎駿は長編中心だが、高畑勲はテレビアニメにも関わっている。 *Wikipedia: アニメの歴史より引用*

約 12,500,000 件 （0.56 秒）

Figure 2: A screenshot of our chrome plugin. For the Japanese search query Hayao Miyazaki, top five interesting relationships are presented at the top of the search results. The red texts are translations of them.

are considered limited. In particular, although relationships between persons are important as a target for text mining, there are few effective approaches for extracting interesting relationships between persons.

Figure 1 shows examples of interesting relationships between persons.[1] The first example is a famous film director who initially had a fairly low regard for an actor who is now extremely famous and successful. The second example is about a famous baseball player who asked another famous baseball player for an autograph. The third example relates to famous musicians engaged in something completely unrelated to music. These examples illustrate that surprisingness is an important factor in interesting personal relationships.

In this paper, to extract such interesting relationships, we focus on surprising relationships between persons. We propose a method that extracts relationships between persons from natural language texts and then scores their surprise scores based on the Mahalanobis distance (De Maesschalck et al., 2000), which has been used in the outlier detection task. Our proposed method first extracts all personal relationships from dependency trees for each sentence and then calculates the surprise scores of the extracted relationships on a continuous vector space in an unsupervised manner. As such, our method does not require any labeled dataset for extracting the surprising personal relationships.

The results of our human evaluation show that the proposed method could extract more interesting relationships between persons from Japanese Wikipedia articles than a popularity-based baseline method. Furthermore, as shown in Figure 2, we incorporated our method into a google chrome plugin. You can watch our demo video for this plugin at a shared directory in our google drive.

[1]These examples were extracted from Japanese Wikipedia articles and then were translated into English.

2 Extracting Interesting Relationships between Persons

Figure 3 provides an overview of the entire process of extracting sentences that may include interesting personal relationships about a target person from given documents. The extraction procedure is as follows:

1. Construct dependency trees from sentences in the target documents through an automatic dependency parser.
2. Extract personal relationships that are represented as tuples of persons and their relationships from the obtained dependency trees.
3. Calculate scores for whether the extracted personal relationships are interesting or not.
4. Select top-k personal relationships and sentences that include the target person based on the calculated scores.

The details of each step are described in the following subsections.

2.1 Extracting Personal Relationships

We use a dependency parser for extracting personal relationships from sentences. First, we parse given sentences with the parser and obtain their dependency trees. Next, if a sentence includes more than one person name, we extract pairs of two names e_i and e_j. We also extract a set p_k that includes words $\{w_1, \cdots, w_n\}$ in the shortest path between e_i and e_j on the dependency tree. These elements are represented as a tuple r_l as follows:

$$r_l = (e_i, e_j, p_k). \quad (1)$$

Because r_l is a tuple, it satisfies $r_{l,0} = e_i$, $r_{l,1} = e_j$, and $r_{l,2} = p_k$.

2.2 Representation of Personal Relationships

For calculating a score of interestingness for r_l, we encode e_i, e_j, and p_k into fixed-dimensional continuous vectors by utilizing the skip-gram model (Mikolov et al., 2013). When training the model, we treat a person name as a single word. Hereafter, we represent the vector of a word w_i as E_{w_i}. Thus, the person names e_i and e_j are represented as E_{e_i} and E_{e_j}, respectively.

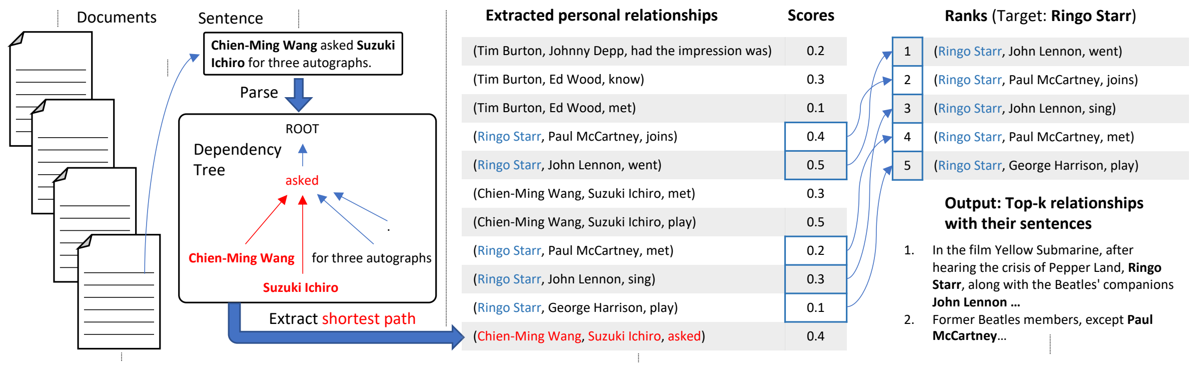

Documents
Sentence
Chien-Ming Wang asked Suzuki Ichiro for three autographs.
Parse
ROOT
Dependency Tree
asked
Chien-Ming Wang
for three autographs
Suzuki Ichiro
Extract shortest path
Extracted personal relationships
Scores
(Tim Burton, Johnny Depp, had the impression was) 0.2
(Tim Burton, Ed Wood, know) 0.3
(Tim Burton, Ed Wood, met) 0.1
(Ringo Starr, Paul McCartney, joins) 0.4
(Ringo Starr, John Lennon, went) 0.5
(Chien-Ming Wang, Suzuki Ichiro, met) 0.3
(Chien-Ming Wang, Suzuki Ichiro, play) 0.5
(Ringo Starr, Paul McCartney, met) 0.2
(Ringo Starr, John Lennon, sing) 0.3
(Ringo Starr, George Harrison, play) 0.1
(Chien-Ming Wang, Suzuki Ichiro, asked) 0.4
Ranks (Target: Ringo Starr)
1 (Ringo Starr, John Lennon, went)
2 (Ringo Starr, Paul McCartney, joins)
3 (Ringo Starr, John Lennon, sing)
4 (Ringo Starr, Paul McCartney, met)
5 (Ringo Starr, George Harrison, play)
Output: Top-k relationships with their sentences
1. In the film Yellow Submarine, after hearing the crisis of Pepper Land, Ringo Starr, along with the Beatles' companions John Lennon ...
2. Former Beatles members, except Paul McCartney...

Figure 3: Overview of our proposed method for extracting interesting relationships between persons from given documents.

To cope with person names e_i with few occurrences, that might cause the sparseness problem, we map person names e_i to clusters, whose number is smaller than the number of person names. We represent a cluster that e_i is assigned to as C_{e_i}. We use k-means as a clustering method to ensure that these clusters are based on the cosine similarity between the vectors.

Unlike the person names, the relationship between two persons, p_k, is represented as a set of words. For encoding the set of words representing the relationship into the continuous vector space, we use smooth inverse frequency (SIF) (Arora et al., 2017),[2] which can encode a sequence of words into a continuous vector by utilizing the frequencies of the words for calculating the weighted sum of the word vectors. Algorithm 1 describes the details of the procedure for obtaining the vector representation of each personal relationship. Through this procedure, we can get V_{p_k}, which is the vector representation of p_k in r_l included in Rel, where Rel is a set of all personal relationships in the corpus.

Algorithm 1 Vector representation for each relationship.

```
Input: All personal relationships $Rel$.
Output: Vectors for each personal relationship
    $\{V_{p_k} | p_k = r_{l,0}, r_{l,0} \in Rel\}$.
    Calculate a weighted sum of the word vectors
    for each $r_l$ based on a word frequency $f(w_{m'})$
    of a word $w_{m'}$ and hyper-parameter $a$.
1: for all relation $p_k$ in $Rel$ do
2:     $V_{p_k} \leftarrow \frac{1}{|p_{k'}|} \sum_{w_{m'} \in p_{k'}} \frac{a}{a+f(w_{m'})} E_{w_{m'}}$
3: end for
    Form a matrix $A$ whose columns are
    $\{V_{p_k} | p_k = r_{l,0}, r_{l,0} \in Rel\}$ and then obtain
    left singular vector $u$ through singular value
    decomposition (SVD).
4: $u \leftarrow SVD(A)$
    Transform the original vectors $V_{p_k}$ with the
    obtained $u$.
5: for all relation $p_k$ in $Rel$ do
6:     $V_{p_k} \leftarrow uu^\top V_{p_k}$
7: end for
```

2.3 Scoring Personal Relationships

In this section, we describe our scoring method for extracting interesting relationships between persons. Our method tries to take into account the following three aspects of the interestingness: **Popularity**, **Surprisingness**, and **Commonness**. The scoring method is based on our assumption that an unusual relationship in a commonly observed pair of two famous persons increases the interestingness, and thus, such a relationship is interesting. The popularity calculates the fame of the persons, the surprisingness calculates the rareness of the relationship, and the commonness calculates how often the pair of the persons commonly appears. The next subsections explain the scores for each aspect in detail.

2.3.1 Popularity

To judge whether the relationships between persons are interesting or not, the reader must know them in advance. From this viewpoint, we consider that the popularity of each person is an important factor in judging whether the relationship between the persons is interesting. Taking this assumption into account, we define $S_{ppl}(e_j)$, the popularity for e_j, as follows:

$$S_{ppl}(e_j) = log(1 + freq(e_j)), \qquad (2)$$

[2] https://github.com/PrincetonML/SIF

where $freq(\cdot)$ is a function that returns the frequency of the input element. $S_{ppl}(e_i)$ can be similarly defined. Note that we use Wikipedia articles for counting the frequency of entities.

2.3.2 Surprisingness

We assume that a surprising personal relationship is a kind of outlier in a set of personal relationships. We use the Mahalanobis distance (De Maesschalck et al., 2000) in the outlier detection task for defining the surprisingness of a personal relationship. Since both the persons and their relationships are represented as continuous vectors, we use a multivariate normal distribution to handle them. If the dimensions of continuous vectors are independent with each other, the variance-covariance matrix of the multivariate normal distribution becomes a diagonal matrix. Under this condition, the Mahalanobis distance is defined as follows:

$$Outlier(\mathbf{x}_i; X) = \sqrt{\sum_{j=1}^{D} \frac{(\mathbf{x}_{i,j} - \hat{\mu}_j)^2}{\hat{\sigma}_j^2}}, \quad (3)$$

where D is a dimension size of $\mathbf{x}$. As explained later, while we consider vector representations of entities as elements of X for the commonness, we consider vector representations of relationships between persons as elements of X for the surprisingness. Both the elements are based on co-occurrence of persons. Thus, these may encounter the sparseness problem.

To deal with the sparseness problem of the elements in X, we use a maximum a posterior probability (MAP) estimation to calculate the mean $\hat{\mu}$ and variance $\hat{\sigma}$. Assuming that each dimension of the continuous vectors obey a normal distribution whose prior distribution of the mean is also a normal distribution $N(\alpha, \beta^2)$ with mean α and variance β^2, the mean $\hat{\mu}$ and the standard deviation $\hat{\sigma}$ is estimated as follows:

$$\hat{\mu} = \frac{\alpha \odot \sigma \odot \sigma + \beta \odot \beta \odot \sum_{i=1}^{|X|} x_i}{|X|\beta \odot \beta + \sigma \odot \sigma}, \quad (4)$$

$$\hat{\sigma} = \sqrt{\frac{1}{|X|} \sum_{i=1}^{|X|} (\hat{\mu} - x_i) \odot (\hat{\mu} - x_i)}, \quad (5)$$

where $|X|$ is the number of elements in X, and $\odot$ is an element-wise product. To use Eq. (3) for calculating surprisingness for a given personal relationship, we need to consider a set $Set_{e_i,e_j,*}$ whose elements are relationships between persons e_i and e_j. However, considering a pair of entities may cause the sparseness problem. To avoid the problem, we use clusters again (as explained in Section 2.2) for representing e_i and e_j to define $Set_{e_i,e_j,*}$ as follows:

$$Set_{e_i,e_j,*} = \{p_k = r_{n,2} | C_{r_{n,0}} = C_{e_i} \wedge C_{r_{n,1}} = C_{e_j} \wedge r_n \in Rel\}. \quad (6)$$

By using $Set_{e_i,e_j,*}$, the surprisingness of a relationship p_k between e_i and e_j, S_{sup}, is calculated as follows:

$$S_{sup}(e_i, e_j, p_k) = Outlier(V_{p_k}; \{V_{p_{k'}} | p_{k'} \in Set_{e_i,e_j,*}\}). \quad (7)$$

When calculating the outlier scores in Eq. (7), we estimate the prior mean α and prior variance β^2 through a maximum likelihood estimation, based on the whole vector representation of personal relationships in the corpus.

2.3.3 Commonness

To determine whether relationships between persons are surprising or not, people must know the ordinary relationships between them in advance.

For example, in Ex.3 of Figure 1, to be surprised by this sentence, the readers must know the common relationships between Ringo Starr and the other members of The Beatles. Since they know that singing, playing a music, etc. are the common relationship among the members of The Beatles, they can be surprised by the phrase "went to the bottom of the sea" in the sentence. Thus, considering how often a pair of persons have a relationship can support our surprisingness. Based on the assumption, our commonness measures how common a pair of two persons.

Since counting the co-occurrence between two persons may cause the sparseness problem, we use continuous vectors for calculating this score. Specifically, we use the minus valued score of Eq.(3), based on the assumption that a pair of two persons is the common pair if it is not an outlier. To use Eq.(3) for calculating commonness, we need to use a set $Set_{e_i,*}$ whose elements are a person who has a relationship with a person e_i. To avoid the sparseness problem, we represent e_i as a cluster again (as explained in Section 2.2) and define $Set_{e_i,*}$ as follows:

$$Set_{e_i,*} = \{e_j = r_{n,1} | C_{r_{n,0}} = C_{e_i} \wedge r_n \in Rel\}, \quad (8)$$

where Rel is a set that includes all relationships between persons in the corpus. By using $Set_{e_i,*}$, commonness S_{com} from e_j to e_i is calculated as follows:

$$S_{com}(e_i|e_j) \tag{9}$$
$$= -\, Outlier(E_{e_i}; \{E_{e_{i'}}|e_{i'} \in Set_{e_j,*}\}). \tag{10}$$

$S_{com}(e_j|e_i)$ is defined similarly. Because $S_{com}(e_i|e_j)$ and $S_{com}(e_j|e_i)$ do not return the same score, we simply use their average for our final score. When calculating the outlier scores in Eq.(10) , we estimate the prior mean α and prior variance β^2 through a maximum likelihood estimation based on the whole word vectors.

2.4 Selecting Top-k Personal Relationships

For ranking personal relationships, we combine all the above three scores. Because these scores have different ranges with each other, we scale them with z-score normalization (Kreyszig, 1979). Let the mean of S_{ppl}, S_{com}, and S_{sup} on all relationships be respectively μ_{ppl}, μ_{com}, and μ_{sup}, and let the variance of S_{ppl}, S_{com}, and S_{sup} on all relationships be respectively σ_{ppl}, σ_{com}, and σ_{sup}. The final score of the interestingness for the target entity e_i is defined as follows:

$$S_{int}(e_i, e_j, p_k) \tag{11}$$
$$= \lambda_{ppl} \cdot \frac{S_{ppl}(e_j) - \mu_{ppl}}{\sigma_{ppl}} \tag{12}$$
$$+ \lambda_{com} \cdot \frac{1}{2} \cdot \Bigg(\frac{S_{com}(e_i|e_j) - \mu_{com}}{\sigma_{com}}$$
$$+ fracS_{com}(e_j|e_i) - \mu_{com}\sigma_{com} \Bigg) \tag{13}$$
$$+ \lambda_{sup} \cdot \frac{S_{sup}(e_i, e_j, p_k) - \mu_{sup}}{\sigma_{sup}}, \tag{14}$$

where λ_{ppl}, λ_{com} and λ_{sup} are weights for adjusting the importance of each score. We tune these weights by using our validation dataset (explained in the next section). Based on $S_{int}(e_i, e_j, p_k)$, we extract top-k relationships that include the target person e_i.

3 Experiments

We conducted human evaluation to determine how well our proposed method can extract interesting relationships between persons. The next subsections describe the details of our experimental settings and the evaluation results.

3.1 Experimental Settings

3.1.1 Dataset

We used sentences in Japanese Wikipedia as our evaluation dataset. We listed articles whose category includes the word "person" as person names and then selected the persons who have more than five relationships from various domains (e.g., anime, manga, novel, actor, music, movie, sports, comedy, and talent) based on their frequencies in Japanese Wikipedia. To remove historical persons, we selected only those who are categorized as "living persons". Finally, we obtained a total of 50 persons for the test dataset and 12 persons for the validation dataset through this process. We next extracted sentences that include personal relationships for the selected persons by using each of the compared methods, that we will describe in the next subsection. We put the top five sentences ranked by each method that include personal relationships for each selected person in the test dataset. If the same sentence was already included in the dataset, we skip it. After this procedure, for each of the compared methods, 250 sentences were included in the test dataset. To provide contextual information, we added the title of the article where the sentences were found to the sentences in the test dataset. The validation dataset was constructed in the same way for the 12 persons.

All personal relationships were extracted with CaboCha,[3] a chunk-based Japanese dependency parser, with the NEologd dictionary (Sato et al., 2017).[4] To filter the personal relationships in compound sentences, we ignored any personal relationships that include multiple predicates. When a sentence lacks its subject, we complement it with the title of the article that contains the sentence. Furthermore, we filtered any sentences starting with a pronoun or conjugation because such sentences are not understandable without the surrounding sentences.

3.1.2 Compared Methods

We evaluated the performance of the proposed methods and several baselines on our test dataset. The following methods were used as the baselines:

- **Rand**: This method randomly selects five personal relationships for each person.

[3]https://github.com/taku910/cabocha
[4]https://github.com/neologd/mecab-ipadic-neologd

- **Pop**: This method selects five personal relationships on the basis of only the popularity score (Eq.(2)).

We used the following as our proposed methods:

- **Pop+Com**: This method selects five personal relationships on the basis of the combined score of the popularity (Eq.(12)) and the commonness (Eq.(13)). Similar to Eq.(11), we tuned the weight parameters λ_{ppl} and λ_{com} on the validation dataset.

- **Pop+Sup**: This method selects five personal relationships on the basis of the combined score of the popularity (Eq.(12)) and the surprisingness (Eq.(14)). Similar to Eq.(11), we tuned the weight parameters λ_{ppl} and λ_{sup} on the validation dataset.

- **Pop+Com+Sup**: This method selects five personal relationships on the basis of a combination of the popularity, the commonness, and the surprisingness (Eq.(11)).

Prior to running these baselines and proposed methods, we obtained word vectors from Japanese Wikipedia articles by utilizing word2vec.[5] In this step, all sentences were tokenized using MeCab[6] with the NEologd dictionary. We further tuned the word vectors by utilizing a retrofitting approach (Faruqui et al., 2015)[7] with Wikipedia's category information to consider similarities between persons. The retrofitting approach can refine word vectors using graph information by making word vectors close to each other when they have a link in the graph. To construct a graph for personal similarities, we linked two words if a Wikipedia category includes the words. Because some person names have several articles due to their ambiguity, we skipped such words in this step.[8] In the end, we reran the retrofitting with the default hyperparameters. Then, we mapped the obtained word vectors of person names to 300 clusters estimated by k-means. When calculating the vectors for each personal relationship, we set a in SIF to 1.0.

[5]https://code.google.com/archive/p/word2vec/

[6]http://taku910.github.io/mecab/

[7]https://github.com/mfaruqui/retrofitting

[8]Note that in Wikipedia, to disambiguate such words, brackets in article titles indicate their ambiguity. Thus, we can skip ambiguous titles based on the brackets.

	$k = 1$	$k = 2$	$k = 3$	$k = 4$	$k = 5$
Rand	49.3	51.4	51.3	51.5	51.1
Pop	51.2	52.5	52.9	52.5	**52.0**
Pop+Com	52.1	52.8	53.3	52.1	51.8
Pop+Sup	54.7†	52.7	53.2	**52.6**	**52.0**
Pop+Com+Sup	**54.9**†	**52.8**	**53.8**	52.0	51.6

Table 1: Evaluation results of rescaled 5-scale scores (%). The bold values indicate the best scores. † indicates that the difference of the score from the best baseline is statistically significant.[10]

We tuned weight parameters in our methods on our validation dataset, which were created for 12 person names in Japanese Wikipedia, and which are not overlapped with the test dataset. We gathered 123 relationships related to the selected persons. Because ranking the degree of interestingness for the gathered relationships would be very costly, we simply attached a label of whether it is interesting or not to them. After that, we estimated the weight parameters by utilizing logistic regression. In Pop+Com, estimated λ_{pop} and λ_{com} were respectively 0.79 and 0.21; in Pop+Sup, estimated λ_{pop} and λ_{sup} were respectively 0.80 and 0.20; and in Pop+Com+Sup, estimated λ_{pop}, λ_{com}, and λ_{sup} were respectively 0.67, 0.17, and 0.16.

3.1.3 Evaluation Metrics

The extracted top five sentences for each method were evaluated in terms of interestingness by six human raters, who rated them on a five-point Likert scale ranging from one to five (Larger is better.). For this rating, we used Lancers,[9] a Japanese cloud sourcing service. We showed personal relationships and their sentences to the raters. For interpretability, we rescaled the rating in the range from 0.0 to 1.0 (Preston and Colman, 2000). In this rescaling, the five scales, 1, 2, 3, 4, and 5, are respectively mapped to 0.0, 0.25, 0.5, 0.75, and 1.0. We averaged the scores of all k-best results for each method.

3.2 Results

Table 1 shows the results of the five-scale scores. Pop+Sup achieved statistically significant improvement over the baselines when $k = 1$. This result can support our expectation that the surprisingness has a strong correlation to the interesting-

[9]https://www.lancers.jp/

[10]We used paired-bootstrap-resampling (Koehn, 2004) with 10,000 random samples ($p < 0.05$).

ness of relationships between persons. In addition, Pop+Com+Sup achieved statistically significant improvement over the baselines when $k = 1$, and outperformed the scores of Pop+Sup when $k = 1, 2, 3$. These results indicate that the commonness can also support the interestingness, especially for a small number of k. When k is larger than 2, all scores are close compared with the scores at $k = 1$. This tendency may suggest that the number of interesting personal relationships is limited for each person.

4 Demonstration System

As shown in Figure 2, our demonstration system presents the top five interesting relationships between persons at the top of the search results based on the current search query. This demonstration system consists of server and client sides. The working process of the system follows the order:

1. In the client side, our google chrome plugin makes a query based on the name of the person input in the google search form.
2. The server-side distributes personal relationships of the person included in the given query to the client-side by loading from the pre-computed personal relationships and their scores.
3. After receiving the result, the client-side shows the result below the search form. If the server does not return any personal relationship, the plugin does not have any action for the search result.

The client-side was implemented on jQuery libraries, and the server-side was implemented on python 3.0 with utilizing http.server module. We chose Pop+Com+Sup as our demonstration system because this model achieved the best result in the human evaluation in the cases of $k = 1, 2$, and 3.

5 Related Work

There have been several approaches for extracting interesting facts. We can divide them into supervised and unsupervised approaches.

The unsupervised approaches have been commonly used for this type of extraction. Merzbacher (2002) proposed a method that mines good trivia questions from a relational database based on predefined rules. Lin and Chalupsky (2003) proposed a set of unsupervised link discovery methods that can compute interestingness on graph data that is represented as a set of entities connected by a set of binary relations. Tsurel et al. (2017) proposed an algorithm that automatically mines trivia facts from Wikipedia by utilizing its category structure. Their approach can rank the entity's categories by their trivia quality, which is induced by the category. Korn et al. (2019) mined trivia facts from superlative tables in Wikipedia articles. They utilized a template-based approach for semi-automatically generating natural language statements as fun facts. Their work had actually been incorporated into the search engine by Google. Kwon et al. (2020) proposed a method to obtain sentences including trivia facts by focusing on a tendency of the Wikipedia article structure that a paragraph containing trivial facts is not similar to other paragraphs in a article.

The supervised approaches have also been used for extracting interesting facts. Gamon et al. (2014) proposed models that predict the level of interest a user gives to various text spans in a document by observing the user's browsing behavior via clicks from one page to another. Prakash et al. (2015) constructed a labeled dataset for movie entities and proposed a method for extracting interesting sentences from Wikipedia articles and ordering them based on interestingness by utilizing Rank-SVM trained with the constructed dataset. Fatma et al. (2017) proposed a method for automatically mining trivia facts for an entity of a given domain in knowledge graphs by utilizing deep convolutional neural networks trained in a supervised manner.

6 Conclusion

In this paper, we proposed a method for extracting interesting relationships between persons from natural language texts in an unsupervised manner.

Human evaluation of the personal relationships extracted from Japanese Wikipedia articles showed that the proposed method improved the interestingness compared to a popularity-based baseline. Through the result, we can conclude that considering the surprisingness of relationships between persons is effective in improving the interestingness of the extracted results.

Furthermore, to demonstrate our proposed method, we incorporated the method into a google chrome plugin, which can work on google search.

As future work, we will investigate ways to extract personal relationships based on more detailed information about a dependency tree.