Extracting Representative Phrases from Wikipedia Article Sections

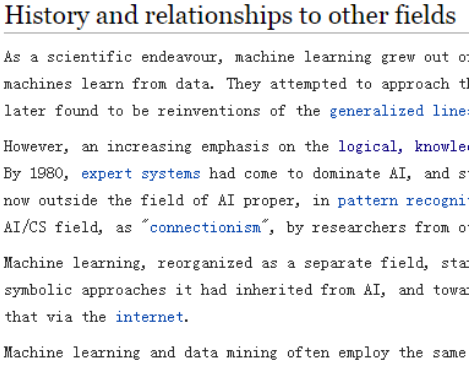
Shan Liu† Mizuho Iwaihara‡
Graduate School of Information, Production and Systems, Waseda University
2-7 Hibikono, Wakamatu-ku, Kitakyushu-shi, Fukuoka-ken, 808-0135 Japan
E-mail: †vickey-liu@toki.waseda.jp, ‡iwaihara@waseda.jp

Abstract Nowadays, Wikipedia has become one of the most important tools for searching information. Since its long articles are taking time to read, as well as section titles are sometimes too short to capture comprehensive summarization, we aim at extracting informative phrases that readers can refer to. Existing work on topic labelling works effectively and performs well on document categorization, but inadequate for granularity of detailed contents. Besides, existing keyphrase construction methods just perform well on very short texts. So we try to extract phrases which represent the target section content well among other sections within the same Wikipedia article. We also incorporate related external articles to increase candidate phrases. Then we apply FP-growth to obtain frequently co-occurring word sets. After that, we apply improved features which characterize desired properties from different aspects. Then, we apply gradient descent on our ranking function to obtain reasonable weighting on the features. For evaluation, we combine Normalized Google Distance (NGD) and nDCG to measure semantic relatedness between generated phrases and hidden original section titles.

Keywords Wikipedia,Co-occurring Word Sets,Gradient Descent

1 Introduction

As a free, open encyclopedia, Wikipedia provides abundant information sources. People are allowed to edit items on Wikipedia, where thousands of new items added every day, and thousands of modifications per hour, resulting in many different authors in different backgrounds. In this case, there are different emphases due to different authors even in one section. Most current approaches to topic construction yield ranked lists of unigrams to represent topics. However, it has long been known that unigrams account for only a small fraction of human-assigned index terms [7]. However, for a person who is unfamiliar with the topic may not easily capture the content quickly. On the other hand, there are only one section title and several subtitles where titles present as phrases, which are too general to present the content of sections thoroughly. Namely, one or two phrases cannot cover the main content of sections. For instance, the following section titled “History and relationships to other fields” in article “Machine Learning” and the main content is: “Machine learning grew out of the quest for artificial intelligence, started to flourish in the 1990s by shifting focus away from AI, and toward methods and models borrowed from statistics and probability theory, and there are overlaps between data mining.”


History and relationships to other fields

As a scientific endeavour, machine learning grew out o
machines learn from data. They attempted to approach t
later found to be reinventions of the generalized line

However, an increasing emphasis on the logical, knowle
By 1980, expert systems had come to dominate AI, and s
now outside the field of AI proper, in pattern recogni
AI/CS field, as "connectionism", by researchers from o

Machine learning, reorganized as a separate field, sta
symbolic approaches it had inherited from AI, and towa
that via the internet.

Machine learning and data mining often employ the same

Figure 1. Section “History and relationships to other fields” in article “machine learning”

But we cannot capture the main content by the section title “History and relationships to other fields” since its over-general. But if there are such phrases, like “artificial intelligence”, “probability theory” and “data mining” etc. we may mainly grasp the section content quickly.

So there is a need to extract n-gram keyphrases to present more information which allow people to take a glimpse of these phrases and capture the general content quickly. A simple way is to extract chunks in sections by statistical probability and latent topic model [3]. However, Wikipedia articles are open to people from all over the world with various writing styles. For instance, it is common that a keyphrase is slightly separated by other words and words in the phrase are in different sequences. But chunks are consecutive and fixed sequences, where non-consecutive phrases are forbidden.To address that, we make improvements over our previous work [11]. We use a traditional method [4] of frequent co-occurring word extraction to obtain a set of words as keyphrase candidates. Then we apply improved feature models for generating representative phrases in one section that can represent the content well and comprehensively. Firstly, we incorporate related articles in Wikipedia that are similar to the target section, so that more quality but hidden content can be extracted. We apply FP-growth [4] to obtain frequent patterns which are sets of words co-occurring frequently.

Meanwhile, we improve the four features that used to measure phrase qualities, which are *coverage*, *phraseness*, *uniqueness* and *potentialness*. Uniqueness and potentialness are proposed by Han et al. (2015) [11]. Coverage and phraseness are proposed by Danilevsky et al. (2014) [7]. Finally, we apply gradient descent to find optimum weighting between the four features. Based on the weight scores we rank candidate phrases. The rest of this paper is organized as follows. Section 2 covers related work. Section 3 sketches out our basic framework, including our improved method and algorithms. Section 4 displays our experiment result, in Section 5, we introduce an evaluation method, and in Section 6, we address a conclusion and future work.

2 Related work

Existing approaches to unsupervised phrase extraction are generally graph-based or unigram-based, which extract ranked representative unigrams first, then combine them into a phrase. Liu et al.[13] transform traditional graph-based ranking method as follows: single random walk into multiple random walks specific to various topics by building a Topic PageRank (TPR) on word graphs to measure word importance with respect to different topics. Graph-based topic labelling [2] by making use of graph-based data structure in DBpedia performs pretty well on topic labelling. Symonds et al. [8] combine an unigram relevance model and linguistic term dependencies to model word associations which are used in query expansion techniques.

Unlike most of these methods, which focus on topic labelling by unigrams or keyphrase extraction on long documents [2, 6] or very short text like microblogs [10], queries [8], we concentrate on Wikipedia article sections, where some of them are long texts while others may be short texts.

Many researchers also apply word sequence segmentation [5] and chunks on keyphrase extraction [3], which are based on statistics or part-of-speech. In this approach, however, keyphrases separated by several other words cannot be captured.

3 Framework

Since section titles in Wikipedia articles are sometimes too short to capture comprehensive summarization, we aim at extracting informative phrases that readers can refer to. So we are working on representative keyphrase extraction on article sections. When given an article with several sections, we want to return several representative phrases for every section that these phrases can reflect the main content of these sections.

The phrases we extracted shouldn't be limited in target section, since phrases in the section may not be enough or hard to judge popularity. So we need external articles to evaluate popularities on similar phrases and provide latent topics. Due to the different emphases and writing styles of different authors, we apply FP-growth [4] to extract co-occurring patterns as order-free word sets. Since these word sets are order-free, we apply hits in search engine to find are a reasonable order that form a sequence phrase for every word set. Then we define four quality measurements to measure these phrases in four different aspects. Finally, we apply gradient descent to find the optimal weights for these four measurements and obtain the overall rank on phrases.

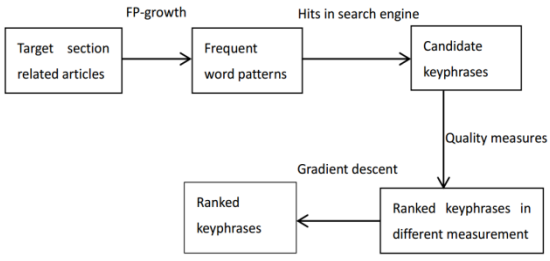


Figure 2. Framework

3.1 Preprocessing

For a target article, we first download all the Wikipedia external related articles which contain the words of the target article title. We preprocess these articles as well as the target article by filtering stop words. For a target section in the target article, we compute

cosine similarities based on TF-IDF for obtaining top-k related articles for every section in the target article.

Because related articles just play a role of support, phrases in the target section should be more significant than those in related articles. Before applying FP-growth [4] to related articles, we concentrate more on the target section. We give more weight on co-occurring words in the target section. In order to obtain hidden topics, we apply LDA (Latent Dirichlet Allocation) [1] onto the target section and related articles, to estimate word-topic distributions and topic-document distributions, which are used to obtain the likelihood of words occurring in the target section and article.

3.2 Frequent patterns

Frequent patterns are those appear frequently in a data set. The frequent patterns do not reflect word orders but co-occurrence. For example, a record contains a set of items, such as milk and bread, which appear frequently together in a transaction data set, is a frequent item set. The customer baskets (records) are as same as articles, where the basket items are as same as words. So if word sets co-occur in many articles, it is likely to be a good candidate keyphrase. For example, there are four baskets: {‘milk’,‘bread’,’cereal’}, {‘milk’,‘bread’,‘sugar’,‘eggs’}, {‘milk’,‘bread’,’butter’}, {‘sugar’,‘eggs’}, if we set the frequency count equals two, we obtain the most frequent patterns {‘milk’,’bread’} and {‘sugar’,’eggs’}. The frequency count is named as support.

Since phrases usually occur once or twice in one section, it is hard to conclude significance of a phrase simply by its frequency in the target section, or in the article. So we need to concentrate more on the words occurred in the target section. Then we try to add more records that contain frequent words occurring in the target sectionas an input to FP-growth[4]. For instance, the word “white”, ”house” and “Wellesley” are frequent in the target section (i.e. with frequencies of 10, 13 and 8 respectively), so we may add more records (i.e. 2, 3 and 1 records respectively) that contain these words. So the adding records are: {“white”, “house”, “Wellesley”}, {“white”, “house”} and {“house”}.

To achieve that, first, we calculate the frequency of each word occurringin the target section and give a rank for each word based on the frequency. Then we apply the following function for ranking words:

$$\#\text{records}(w) = \frac{constant}{\#ranks} * (\#ranks - curRank) \quad (1)$$

Where *#records(w)* means the number of adding records that word *w* should be contained; *constant* is given by experience, here we set to 13; $curRank$ refers to the current rank of word *w* according to the frequency; $\#ranks$ refers to the number of ranks in the target section. In this way, we iteratively add records that are related to the target section.

Then apply FP-growth [4] on the target section, related articles, and adding records together for once, and we obtain the frequent patterns of order-free word sets.

3.3 Candidate keyphrases

When we search information on a search engine, such as Google, Bing, Yahoo etc, the engine will return results with their hit counts to us.

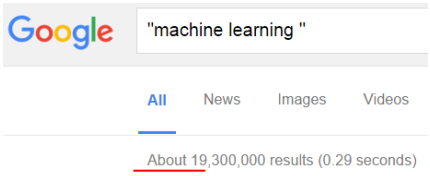


Figure 3. Hits result

The frequent co-occurring word sets are order-free in FP-growth. But what we need are meaningful phrases with sequence as candidate keyphrases, then we enumerate possible permutations and applied to search engine (we chose “Bing” search engine in our research) as a query. We choose the highest hits as a representative ordering of this word set.

3.4 Quality measurements

To evaluate quality of these candidate keyphrases, we introduce the following four measurements from different aspects: *coverage*, *phraseness*, *uniqueness*, *potentialness*. Coverage is from [7], phraseness is originally from [7] but we extended for our case. Uniquness and potentialness are proposed in our previous work[11] but further improved in this paper. In the following definitions, the section corpus C consists of the target section, augmented with thetop-krelated articles.

- **Coverage**[7]

A representative keyphrase should cover many articles. For instance, in the target section titled “early life and education” of arcicle Hirrary Clinton, the phrase “political president” covers four articles, while phrase ”student year” covers eight articles, so the latter is better than the former. The scoring function of coverageis as follows:

$$S_{cov}(p) = \frac{f(p)}{|D|} \quad (2)$$

Where $f(p)$refers to the frequency of phrase *p* occurring in corpus C; |D| refers to the number of articles in section corpus C.

- **Phraseness**

The words in a representative keyphrase are likely to co-occur. Since candidate keyphrases are generated from FP-growth, which find frequent word sets, there is a high chance that words in keyphrases are just combinations of frequent words, even they are far apart in texts, like a frequent word at the beginning of the target section while the other frequent word at the tail. So we restrict words of a keyphrase to co-occur in one sentence. Besides, we need to considerthe correlations between the target section and related articles.

$$S_{phr}(p) = \sum_{i=0}^{n} \frac{n-i}{n} * \frac{f_i(p)}{\prod_{w \in p} f_i(w)} \quad (3)$$

Here, *i* denotes similarity rank by TF-IDF cosine similarities to the target section, *n* denotes the number of top-k related articles, $f_i(p)$denotes the frequency of phrase *p* where *p* occurs in an identical sentence of article *i*, and $f_i(w)$ denotes the frequency of word *w* in article i.

- **Uniqueness** [11]

A representative phrase should be more frequent in the target section rather than other sections in the target article, to extract phrases that represent the target section well among other sections.

$$S_{uni}(w) = \log\left(\frac{|S|}{f(s)} * \frac{f_s(w)}{\sum_{s' \in S, s' \neq s} f_{s'}(w) + 1}\right) \quad (4)$$

Here $|S|$ is the number of sections in the target article, $f(s)$ refers to the number of sections containing word *w*, $f_s(w)$ is the frequency of word *w* in section *s*. Then the score of phrase p is the average score of words in phrase p.

$$S_{uni}(p) = \frac{\sum_{w \in p} S_{uni}(w)}{|p|} \quad (5)$$

Here, |p| refers to the number of words in phrase *p*.

- **Potentialness**[11]

Potentialness is intended to measure how phrases are related to latent topics of the target section. We assume that phrases that are highly related to latent topics of the section are more suitable for section titles. We evaluate the likelihood of words from word-topic distribution and topic-document distribution by Latent Dirichlet Allocation(LDA) [1].

$$S_{pot}(w|s) = \sum_{j=0}^{k} p(w|t_j) * p(t_j|s) \quad (6)$$

Here, *k* refers to the number of topics, $p(w|t_j)$ is the word-topic distribution computed by gibbsLDA, and $p(t_j|s)$ is the topic-document distribution.

$$S_{pot}(p) = \frac{\sum_{w \in p} S_{pot}(w|s)}{|p|} \quad (7)$$

Here, |p| refers to the number of words in phrase *p.*

3.5 Ranking function

By each of the above four features, we can rank candidate phrases. To obtain the combined ranking of phrases, we define a ranking function for candidate keyphrases.

$$S(p) = \theta_0 S_{phr}(p) + \theta_1 S_{cov}(p) + \theta_2 S_{uni}(p) + \theta_3 S_{pot}(p) \quad (8)$$

Here, $S(p)$ refers to the score of phrase *p* in the target section. Let $\boldsymbol{\theta} = [\theta_0, \theta_1, \theta_2, \theta_3]$ denotes the weight vector on the four features.

3.6 Gradient descent

We define a ranking function which combines the four features by a linear function.We apply gradient descent to obtain the optimal parameter settings for the ranking function. Firstly, we formalize the optimizing function as follows:

$$h(\boldsymbol{\theta}) = \theta_0 x_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_3 \quad (9)$$

where $h(\boldsymbol{\theta})$ refers to the score of phrase *p* in the target section, and x_i is the variable on the *i*-th feature.

The cost function is defined as follows:

$$J(\theta) = \frac{1}{2}\sum_{i=0}^{n}(h_\theta(x_i) - 1)^2 \quad (10)$$

We apply batch gradient descent as follows:

For j=1 to m{

$$\theta_i = \theta_i - \alpha \frac{\partial J(\theta)}{\partial \theta_i} \quad (11)$$

} (for all i)

Where *m* is the number of training examples (also corresponding to the number of candidate keyphrases), α is the learning rate. After obtaining the optimum weight vector, we rank candidate phrases by the ranking function $S(p)$.

4. Experiment

4.1 Dataset

We download 17 featured English Wikipedia articles (which are selected as excellent articles in Wikipedia). Since we need to compare our results with original section titles that are hidden, we choose 51 sections which are rich in section titles and subsection titles. External related articles are also downloaded from Wikipedia, in which the words of the article title of the target article occur consecutively.

4.2 Parameters

The section corpus *C* of each target section consists of the target section and top-*k* related articles based on the combination of TF-IDF and cosine similarity. Here, we set k = 15. For gradient

descent, we set the learning rate α=0.0005. We divide our dataset into two parts, where Corpus 1 has seven articles, while Corpus 2 has ten articles to see if the parameters of gradient descent are different in different corpus.

Articles in Corpus 1: {"Hillary Clinton", "Attachment theory", "History of Minnesota", "Domitian", "Political integration of India", "Philosophy of Mind", "Nikita Khrushchev"}.

To verify if the parameters are related to the scale of the training dataset, we test on different scales of candidate keyphrases. We do experiments on four different scales: top-k (here, we let $k = 5, 10, 15, 20$ respectively) candidate keyphrases which are ranked by the ranking function in Section 3.5.

Table 1. Parameters determined from Corpus 1

θ / Top-k	θ_0	θ_1	θ_2	θ_3
5	0.826	0.196	0.279	0.439
10	0.961	0.176	0.246	0.412
15	1.014	0.155	0.226	0.420
20	1.058	0.125	0.206	0.423
Average	0.965	0.163	0.239	0.423

We can see that the obtained parameters under different top-k are performed quite stable.

Articles in Corpus 2: {"Greek mythology", "Barack Obama", "Bryan Gunn", "John Sherman", "Wood Badge", "General relativity", "Society of the Song dynasty", "Richard Nixon", "Uruguayan War", "William the Conqueror"}

We do the same on these 10 articles.

Table 2. Parameters determined from Corpus 2

θ / Top-k	$\boldsymbol{\theta_0}$	$\boldsymbol{\theta_1}$	$\boldsymbol{\theta_2}$	$\boldsymbol{\theta_3}$
5	1.055	0.213	0.263	0.293
10	1.160	0.174	0.235	0.310
15	1.258	0.122	0.215	0.310
20	1.197	0.152	0.235	0.316
Average	1.167	0.165	0.237	0.310

We can see that parameters are stable in different top-k and almost the same to the corresponding parameters. Then we combine the two corpora, to find if the parameters are still stable.

Articles in Corpus 3: all 17 articles

Table 3. Parameters determined from Corpus 3

θ / Top-k	θ_0	θ_1	θ_2	θ_3
5	1.006	0.192	0.267	0.313
10	1.057	0.175	0.248	0.355
15	1.097	0.155	0.241	0.369
20	1.148	0.126	0.224	0.367
Average	1.070	0.163	0.240	0.361

To see more clearly, We put the average result together for comparison.

Table 4. Comparison on different corpus

Corpus	$\boldsymbol{\theta_0}$	$\boldsymbol{\theta_1}$	$\boldsymbol{\theta_2}$	$\boldsymbol{\theta_3}$
1	0.965	0.163	0.239	0.423
2	1.167	0.165	0.237	0.310
3	1.077	0.162	0.254	0.351

Above all, the average results of parameters in different corpus and different top-k phrases are close. So we apply parameters from the larger corpus $\boldsymbol{\theta} = [1.077, 0.162, 0.254, 0.351]$ to the ranking function for every phrase.

4.3 Result

After we apply $\boldsymbol{\theta}$ to the ranking function in equation(9), we can obtain the ranked list of keyphrases for each section. In fact, the result includes 51 sections, we show three sections to inspect their qualities.

Table 5. Result of section "Early life and education" in article "Hillary Clinton"

Rank	Phrase
1	hillary clinton
2	college political
3	clinton political
4	law school
5	college school

Table 6. Result of section "First lady of the United States" in article "Hillary Clinton"

Rank	Phrase
1	white house
2	first house
3	first white house
4	first white
5	white house office

From the results of the two sections above, we can see the extracted phrases are quite similar to target section titles and play the role of supplement, which means our phrase extraction model performs pretty well. Keyphrases, such as "law school", "white house" etc. are quite complete and representative phrases, which means phraseness play a quite significant role. The keyphrases we extracted can distinguish sections even in the same article (Hillary

Clinton), which means uniqueness worked.

5 Evaluation

In this section, we discuss an objective evaluation based on semantic closeness to the hidden section titles, where closeness is based on co-occurences over web documents. We compare various combinations of the four features, where the mehod combining all the features by the parameter $\boldsymbol{\theta}$ is called *Combined Measure* .

5.1 Variations for comparison

We compare the following combinations of the features: Combined Measure$_{\text{-cov}}$ refers to the Combined Measure that ignores the effect of *coverage*, Combined Measure$_{\text{-phr}}$ refers to the Combined Measure that ignores the effect of *phraseness*, Combined Measure$_{\text{-uni}}$ refers to the Combined Measure that ignores the effect of *uniqueness*, and Combined Measure$_{\text{-pot}}$ refers to the Combined Measure that ignores the effect of *potentialness*.

These variations represent the possible settings for parameters $\boldsymbol{\theta} = [\theta_0, \theta_1, \theta_2, \theta_3]$ which we described in Section 3.5.

Table 7. Parameters settings for different variations

Method	$[\boldsymbol{\theta_0}, \boldsymbol{\theta_1}, \boldsymbol{\theta_2}, \boldsymbol{\theta_3}]$
Combined Measure	[1.077, 0.162, 0.254, 0.351]
Combined Measure$_{\text{-cov}}$	[1.077, 0.0, 0.254, 0.351]
Combined Measure$_{\text{-phr}}$	[0.0, 0.162, 0.254, 0.351]
Combined Measure$_{\text{-uni}}$	[1.077, 0.162, 0.0, 0.351]
Combined Measure$_{\text{-pot}}$	[1.077, 0.162, 0.254, 0.0]

Besides, we also arbitrarily combine two of the four features as comparison.

Table 8. Parameters settings for combination of two features

Method	$[\boldsymbol{\theta_0}, \boldsymbol{\theta_1}, \boldsymbol{\theta_2}, \boldsymbol{\theta_3}]$
CovPhr	[1.077, 0.162, 0.0, 0.0]
CovPot	[0.0, 0.162, 0.0, 0.351]
CovUni	[0.0, 0.162, 0.254, 0.0]
PhrUni	[1.077, 0.0, 0.254, 0.0]
PhrPot	[1.077, 0.0, 0.0, 0.351]
UniPot	[0.0, 0.0, 0.254, 0.351]

5.2 Quality measurement

To evaluate how representative of these keyphrases we extracted, we need a golden standard. But it is unrealistic for human to give keyphrases for every section as a golden standard, since different people may give different answers based on their own understanding. So we adopt the idea that the original section and subsection titiles are ideal, and phrases that are semantically close to these titles should be high quality. There are a number of methods for measureing semantic distance, such as using ontological knowledge such as WordNet. But here we adopt semantic distance measured by co-occuring frequencies in web documents, a well-known measure of this direction is the NGD(Normalized Google Distance) [9]. Then we apply the distance results on nDCG(Normalized Discounted Cumulative Gain) [12] to measure the quality of ranking. Not that this measure captures semantic closeness to the original hidden titles, but this does not evaluate comprehensivenss and informativeness of phrases, which original titles sometimes lack, and we reserve these evaluations for future work.

5.3 Relevance score

In our work, we need to calculate the semantic distance between the keyphrases and original titles, so we adapt NGD(Normalized Google Distance) [9] which is a semantic similarity measure derived from the number of hits returned by the search engine for given phrases or words. Phrases with the same or similar meanings in a natural language sense tend to be "close" in units of Normalized Google Distance, while phrases with dissimilar meanings tend to be farther apart. The NGD score between two phrases p_1 and p_2 is

$$\mathrm{NGD}(p_1, p_2) = \frac{\max\{\log f(p_1), \log f(p_2)\} - \log f(p_1, p_2)}{\log N - \min\{\log f(p_1), \log f(p_2)\}} \quad (12)$$

where *N* denotes the total number of web pages indexed by a given search engine; $f(p_1)$ denotes the number of pages containing phrase p_1; $f(p_1, p_2)$ denotes the number of pages containing both phrase p_1 and p_2. *NGD(p_1, p_2) = 0* means phrase p_1 is the same as phrase p_2

As NGD is a normalized distance score between phrases, we define the relevance score between two phrases as follows:

$$\mathrm{rel}(p_1, p_2) = 1 - \mathrm{NGD}(p_1, p_2) \quad (13)$$

After that, we can obtain the relevance score between keyphrases and its target section/subsection titles. Then we combine the relevance score between keyphrase and its section title ($\mathrm{rel}(\mathrm{p}, \mathrm{s})$) and the relevance score of keyphrase and its subsection titles ($\mathrm{rel}(\mathrm{p}, \mathrm{sub}_i)$) to calculate the relevance score between the keyphrase and its target section.

$$rel(p) = \alpha * rel(p, s) + \beta * \frac{\sum_{i=1}^{n} rel(p, sub_i)}{n} \quad (14)$$

Here, *n* refers to the number of subsection titles; *rel(p)* refers to the relevance score between phrase *p* and the target section; *rel(p,s)* refers to the relevance score between phrase *p* and section title; $rel(p, sub_i)$ refers to the relevance score between phrase *p* and i$^{\text{th}}$ subsection titles, α and β are weighting parameters. Since

we pay more attention to section titles than subsection titles, we set α and β to 0.7 and 0.3, respectively.

Now every keyphrase has a relevance score to the target section, we can obtain a ranking of these keyphrases according to the relevance scores.

5.4 Normalized Discounted Cumulative Gain

DCG (Discounted Cumulative Gain)[12] measures the gain of a phrase based on its rank position in the result list. The gain is accumulated from the top of the result list to the bottom with the gain of each result discounted at lower ranks. Namely, if a high relevant phrase ranked low or an irrelevant phrase ranked high in the result list will be discounted.

The ranking list we obtained according to the relevance scores is the ideal ranking list, which used to obtain the ideal (standard) DCG value.

$$IDCG_n = \sum_{i=1}^{n} \frac{2^{rel(p)}-1}{log_2(i+1)} \quad (15)$$

Here, *i* refers to the ranking position of phrase *p* according to the relevance score, and *n* is the number of phrases in the target section. We can also obtain the DCG value by the above equation (15) but based on the positions of phrases in our methods (Combined Measurement and its variations).

Then we apply the value of DCG and IDCG in every section to obtain the normalized result.

$$nDCG = \frac{DCG}{IDCG} \quad (16)$$

nDCG ranges in 0.0 and 1.0. In a perfect ranking algorithm, the DCG will be the same as the IDCG, producing an nDCG of 1.0.

5.5 Evaluation results

After we apply above evaluation method to the 51 sections, we can obtain the nDCG value of each section. Here, we show the result of two sections in the article "Hillary Clinton".

Table 9. Section "first lady of the united states"

Method	**nDCG**
Combined Measure	0.7495
Combined Measure$_{-phr}$	0.7451
Combined Measure$_{-cov}$	0.7656
Combined Measure$_{-pot}$	0.8079
Combined Measure$_{-uni}$	0.7463
CovPhr	0.7682
CovUni	0.6743
CovPot	0.7150
PhrUni	0.7909
PhrPot	0.7613
UniPot	0.6801
Coverage	0.8257
Phraseness	0.7870
Uniqueness	0.5660
Potentialness	0.7300

Table 10. Section"marriage and family, law career and first lady of arkansas"

Method	**nDCG**
Combined Measure	0.9271
Combined Measure$_{-phr}$	0.8929
Combined Measure$_{-cov}$	0.6846
Combined Measure$_{-pot}$	0.9005
Combined Measure$_{-uni}$	0.9201
CovPhr	0.8920
CovUni	0.8556
CovPot	0.8575
PhrUni	0.9390
PhrPot	0.9203
UniPot	0.8818
Coverage	0.8737
Phraseness	0.9307
Uniqueness	0.8880
Potentialness	0.8472

Then we take the average nDCG values of each method as the evaluation score for comparison, as shown in Table 11.

Table 11. Average nDCG of all 51 sections of 17 articles

Method	**Average nDCG**
Combined Measure	0.8393
Combined Measure$_{-phr}$	0.8345
Combined Measure$_{-cov}$	0.8364
Combined Measure$_{-pot}$	0.8229
Combined Measure$_{-uni}$	0.8381
CovPhr	0.8276
CovUni	0.8003
CovPot	0.8463
PhrUni	0.8247
PhrPot	0.8360
UniPot	0.8302
Coverage	0.8234
Phraseness	0.8237
Uniqueness	0.8210
Potentialness	0.8380

From the tables above, we can see the Combined Measure

performs very well among these methods. Although there are other methods maybe slightly better than our combined measure. Since section titles have different characteristics, some of them containing popular phrases, others containing words from the section body, and some of them containing words unique to sections. So the methods which are better than our method varies in different sections and overall results, while our method performs quite well and stable.

6 Conclusions

We proposed a generative model which can automatically extract representative phrases for sections in articles. The performance also shows the ability to represent the section content and complete the insufficiency of section titles. This model can also be extended on normal texts without section title, by setting the weight on uniqueness to 0, and it's flexible on the length of text, while many existing works need the text length limitation, either long document or short texts, like microblogs.

For future work, there are still several problems need to be improved, including relatively bad performance of coverag. We may give a reasonable weighting to the target section and different related articles. We can see from the results that several ranked phrases share quite similar meanings. So we need to find a way to reduce redundant phrases.

7 References

[1] D. Blei, A. Ng, and M. Jordan, "Latent Dirichlet Allocation", Journal of Machine Learning Research, 3, 2003.

[2] I.Hulpu, C.Hayes, D.Greene, "Unsupervised Graph-based Topic Labelling using DBpedia", Proc. of ACM WSDM'13, pages 465-474, 2013.

[3] J. Lau, D. Newman, S.Karimi and T. Baldwin,"Best Topic Word Selection for Topic Labelling", Proc. of the 23rd Int. Conf. on Computational Linguistics (COLING'10), pages 605-613, 2010.

[4] J. Han, J. Pei, and Y. Yin, "Mining Frequent Patterns without Candidate Generation", Proc. of the 2010 Conference on Empirical Methods in Natural Language Processing (EMNLP '10), pages 366-376, 2010.

[5] J. Liu, J. Shang, C. Wang, X. Ren, and J. Han, "Mining Quality Phrases from Massive Text Corpora", Proc. of the 2015 ACM SIGMOD Int. Conf. on Management of Data (SIGMOD'15), pages1729-1744, 2015.

[6] K. Hofmann, M.Tsagkias, E.Meij, M.Rijke, "The Impact of Document Structure on Keyphrase Extraction", Proc. of ACM CIKM'09, pages 1725-1728, 2009.

[7] M.Danilevsky, C. Wang, N. Desai, X. Ren, J.Guo, and J. Han,"Automatic Construction and Ranking of Topical Keyphrases on Collections of Short Documents", Proc.of 2014 SIAM Int. Conf. on Data Mining (SDM'14),2014.

[8] M. Symonds, P.Bruza, G.Zuccon, L.Sitbon, I. Turner, "Is the Unigram Relevance Model Term Independent? Classifying Term Dependencies in Query Expansion", Proc. of the Seventeenth Australasian Document Computing Symposium (ADCS '12), pages 123-127, 2012.

[9] R.L. Cilibrasi, P.M.B. Vitanyi, "The Google Similarity Distance", IEEE Trans. Knowledge and Data Engineering, pages 370-383, 2007.

[10] W. Zhao, J. Jiang, J. He, Y. Song, P. Achananuparp, E. Lim, X. Li, "Topical Keyphrase Extraction from Twitter", Proc. of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (HLT '11), pages 379-388, 2011.

[11] X. Han, R. Wang, M. Iwaihara, "Automatic Construction and Ranking of Keyphrases on Wikipedia Article Sections", DEIM Forum 2015.

[12] Y. Wang, L. Wang, Y. Li, D. He, W. Chen, T.-Y. Liu, "A Theoretical Analysis of NDCG Ranking Measures", Proc. of the 26th Annual Conference on Learning Theory,2013.

[13] Z. Liu, W. Huang, Y. Zheng, and M. Sun,"Automatic keyphrase extraction via topic decomposition", Proc. of the 2010 Conf. on Empirical Methods in Natural Language Processing (EMNLP'10),pages 366-376, 2010.