Estimating Reference Scopes of Wikipedia Article Inner-links

Renzhi Wang† 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:†ouninnyuki.ips@asagi.waseda.jp, ‡ iwaihara@waseda.jp

Abstract Wikipedia is the largest online encyclopedia, and utilized as machine-knowledgeable and semantic resources. Links within Wikipedia indicate that two articles or parts of them related about their topics. Existing link detection methods focus on article titles because most of links in Wikipedia point to article titles. But there are a number of links in Wikipedia pointing to corresponding segments, because the whole article is too general and it is hard for readers to obtain the intention of the link. We propose a method to automatically predict whether the link target is a specific segment and provide which segment is most relevant. We propose a combination method of Latent Dirichlet Allocation (LDA) and Maximum Likelihood Estimation (MLE) to represent every segment as a vector, and then we obtain similarity of each segment pair. Finally we utilize variance, standard deviation and other statistical features to predict the results. We also try Word2Vector model to embed all the segments into a semantic space and calculate cosine similarities between segment pairs, then we utilize Random Forest to train a classifier to predict link scopes. Through evaluations on Wikipedia articles, our method achieved reasonable results.

Keyword Wikipedia, link suggestion, LDA, word2vector, PMI

1. Introduction

Wikipedia articles are edited by various volunteers from all over the world, with different thoughts and styles. One of Wikipedia's characters is featured articles. Wikipedia includes many high quality articles that reach the standard of featured article criteria. These articles are usually edited by experienced authors and checked by Wikipedia's administrators. Featured articles are supposed to be well-written, comprehensive, well-researched, neutral and stable. Wikipedia is structured via a number of links between different articles, which imply that the two linked articles are closely related. Majority of links within Wikipedia are pointing to article titles, and only small fractions point to segment titles. However, when readers browse topic via links, sometimes they are only interested in certain segments while the link itself is pointing to article titles, thus the readers could get lost in such long articles. To avoid such situations, administrators and editors often modify link target text from an article title to a specific segment title. Figure 1 shows an example that an editor modified the link target from the article title to the specific segment. In article "Super Mario 64" there is a link, first pointed to article GameCube, and then the editor corrected the link to the segment "Controller".

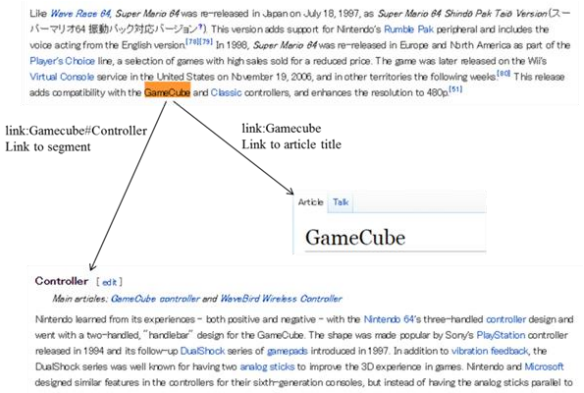

Fig. 1. Wikipedia links

Current link detection methods are focused on article titles [1,2,7,11,12]. They often first generate a candidate set for an article by utilizing existing connections of articles, then rank all articles in the candidate set, and select the most similar article as the final result. But in our case, it is hard to generate a candidate set by using existing connections, due to shortage of links that point to segment titles. Besides, the length of segment texts are usually short, so it is necessary to improve vector representation of segment texts, so that we can obtain better similarity comparisons between segment pairs which helps in candidate set generating process.

In this paper, we discuss the following link suggestion problem: Given a link source, which is a position in a Wikipedia article, we find the most related link target,

which is either a whole article, or a segment in an article. Our previous research utilized LDA based features to train a classifier to predict link target[16]. In this paper our approach employs Latent Dirichlet Allocation (LDA) [3,4] method for topic detection, where segments are represented as vectors of word probabilities. To improve accuracies of conventional LDA-based methods, we have combined LDA with Maximum Likelihood Estimation (MLE) in a nonlinear way, which enables us to compute semantic similarities on the segment level [16]. In this paper, considering about effectiveness of word co-occurrence, we utilize features based on Normalized Pointwise Mutual Information (NPMI)[5], to measure likelihoods of words co-occurring in different segments. We also evaluate similarities between different segments, utilizing word embedding. Word2vec[13,14] is an open source project released by Google which achieves state of the art performances in various natural language tasks. A number of researches proved that word2vec performs very well in calculating text similarity, especially on texts that have specific patterns such as wordvec("king")-wordvec(queen)≈ wordvec("man")-wordvec(woman).

It is hard to predict whether a link should point to a segment, by only using similarities between segment pairs. We compute similarities between one target segment and all the segments in another article, to obtain similarity distributions in one article. We define statistical features based on these similarity distributions. Then we train a classifier to determine whether the link should point to a specific segment rather than the whole article. When we confirm that the link should point to a segment, we compare the similarities between segment pairs to find the most related segment. To solve the imbalanced data problem, we utilize the logistic regression as a filter before final prediction.

In order to improve the result, we propose a stacking model to replace a single classifier. Our evaluation results show that our method is effective.

The rest of this paper is organized as follows. Section 2 shows related work. Section 3 introduces our assumption and proposed method. In Section 4 we describe our dataset in detail, explain our experimental process and we present evaluation results in various situations. In Section 5, we address a conclusion and future work.

2. Related work

Automatically discovering missing links in Wikipedia has been discussed in the literature. Sisay et al. [1] propose a method which can rank pages using co-citation and page title information. They use LTRank to identify similar pages and select top similar articles as the prediction results. This method needs the organizational structures of the articles; LTRank is not suitable for detecting links at the segment level, because there are not enough segment-level links.

Junte proposed a method utilizing TF-IDF and the vector space model to detect document-to-document links and anchor-to-BEP links [19]. Best Entry Point (BEP) is similar to our task. The best entry point here is a specific article belonging to a general article. The difference is that Junte's work is still focusing on the whole article, while our task is to detect the best segment in an article. Junte's research regards the source article as the query and selects top-K similar articles as the result. Then deep-first iteration is repeated to find the final result. This method uses TF-IDF and the vector-space model to compute similarities. But TF-IDF is heavily affected by corpus construction.

David et al. [12] proposed a machine learning-based link detector to detect links between Wikipedia articles. In their method, they did not simply evaluate textual similarity between two articles, but for each article pair, they evaluate five features: link probability, relatedness, disambiguation confidence, generality, location and spread. Then they train a classifier, for predicting whether there should be a link between an article pair. Its result was much better than other similarity-based methods.

3. Proposed method

3.1 Target corpus

As the world's largest encyclopedia, Wikipedia is organized as a large, complex network, where articles are connected by interlinks. Given a target, we assume that interlinks from a target article to another article assists complementing the content of the target article, by incorporating the content of another article. In order to utilize this link structure for incorporating the contents of the linked articles, we construct a suitable corpus for the target article. Given a target article *A*, we regard the union of all the articles that *A* links to and *A* itself as the corpus. Certain links may point to a specific segment, but we include its whole article in the corpus. Wikipedia featured articles are less erroneous and stable, so they are suitable for our experiments. Our dataset consists of randomly sampled featured articles. Since our objective is to suggest a segment-level link, we decompose the articles in the corpus into segments based on their logical structures,

such as paragraphs. The following steps are operated on segments.

3.2 Representing segments as vectors

We argue that linked articles bring additional information to the central article and affect the topics of the central article. The LDA model [3,4] is a popular model that can extract topics from the corpus. Figure 2 shows the structure of LDA model, where documents are regarded as topic distribution and topic is regarded as a word distribution. A document d is sampled from a topic tionθ, and a topic z is represented over words by word distributionϕ.

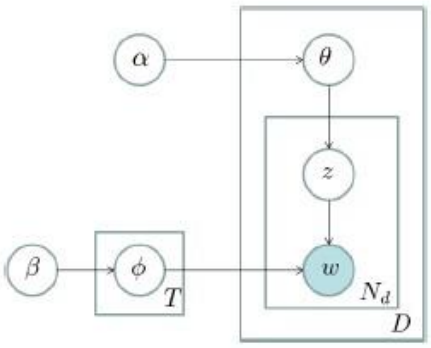

Fig.2. LDA generation process

So we can obtain the probability of a term in a document by the following formula:

$$P_{LDA}\left(w|d,\hat{\theta},\hat{\phi}\right)=\sum_{z=1}^{K}P\left(w|z,\hat{\phi}\right)P\left(z|\hat{\theta},d\right) \quad (1)$$

Here, $\hat{\theta}$ and $\hat{\phi}$ are the posterior estimates of θ and ϕ, respectively. LDA does not perform very well on long tail words, so there are a number of variants over the basic LDA model. One of the variants is to combine LDA and Maximum Likelihood Estimation. The authors of [8,9,15,17,18] utilize linear combination of word-level probability from LDA, document-level probability and collection-level probability to smooth the results. The document-level probability and collection-level probability are obvious parts which can be observed by simple term occurrences. The LDA model can extract the word probability from latent topics which can be regarded as latent part. However, the existing methods adopted a linear combination of obvious part and latent part. But in the assumption of LDA, the word probability of the document is based on the corpus while the document-level probability is based on the current document, so linear combinations may not be the best choices, because it is ad hoc to current corpus. To optimize the combination method, we propose the following nonlinear combination of the obvious part and latent part:

$$p(w|D)=\frac{1}{e^{\alpha N_d+1}}\left[\frac{1}{e^{-\beta N_d+1}}P_{ML}(w|D)+\frac{1}{e^{\beta N_d+1}}P_{ML}(w|Coll)\right] \quad (2)$$

Here, N_d is the number of terms appearing in the segment. The first part of the formula is the probability of the word appearing in the document by term frequencies. The weight proportion of the obvious part and latent part will affect the word probability. This document-level probability is combined with the collection-level probability by the smoothing parameterβ. We adjust the value of β to optimize the obvious part. Smoothing parameter α is to adjust the ratio of the obvious part and latent part. By these two formulae, we can obtain the probabilities of all the words in the corpus. People utilize perplexity to evaluate the language model. The perplexity is smaller when the fitness between the model and data is better. In our experiment, we need to determine the best parameters. We put the experiment data into the formula and select the parameters based on which the perplexity is minimal.

We represent each segment as a vector, whose element is the probability of the segment generating the word. We could use all the words in the corpus as elements of the vector. But to reduce the dimensions of the vectors, we only select words which appear in more than three segments.

Another popular method of present document as a vector is the word2vec model. Its input is large text corpus and output is the word vector for each unique word. Word2vec model embeds all the words into a low dimension space, which all the word is one point at this space and the distance between two words can present its similarity. Document2vec model can present the document to vector, but for new document it needs retrain the model. Our test data is different from training data, so it does not fit for our algorithm. We select the word2vec model as our vector arithmetic.

Pointwise Mutual Information (PMI) is a measure if association used in information theory and statistics. In NLP task, PMI has been often used for finding collocations and association between words. It measures how likely two words are to co-occur. Our object is to compute the similarity between two segments but not words. Mihalcea propose a method to compute the similarity between two sentences based on the normalized PMI of all word pairs in these two sentences. [10] The scoring function is as follow:

$$sim(S_1,S_2)=$$

$$\frac{1}{2}\left(\frac{\sum_{w\in\{S_1\}}(maxSim(w,S_2)*idf(w))}{\sum_{w\in\{S_1\}}idf(w)}+\frac{\sum_{w\in\{S_2\}}(maxSim(w,S_1)*idf(w))}{\sum_{w\in\{S_2\}}idf(w)}\right) \quad (3)$$

Where $maxSim(w,S2_{(1)})$ is the maximum lexical similarity between the word w in segment $S_{1(2)}$ and all the words in segment $S_{2(1)}$ calculated by normalized PMI. $idf(w)$ is the inverse document frequency of the word w calculated from the corpus. This similarity score has a value between 0 and 1, with a score of 1 indicating identical segments, and a score of 0 indicating no semantic overlap between the two segments. By this method, we do not present segments as vectors but directly calculate segment pair similarities.

3.3 Obtaining Similarity Distributions

After representing each segment as a vector, we use cosine similarity to measure the similarity between two segments. We need to evaluate the similarity distribution between segment *SA* in articles *A* and each segment in article *B*. So for *SA* in article *A*, and the link from *SA* that point to article *B*, we compute the similarity between *SA* and each segment in *B*.

3.4 Feature Extraction

A Wikipedia interlink points to an article title or a segment title. As Wikipedia's guideline[1] specifies, a link should point to a segment title when the link source and the link target segment are remarkably similar and describe more details. Simply comparing the similarity between segment *SA* in articles *A* and segment *SB* in article *B*, and the similarity between *SA* and article *B* is not a good idea, because the segment *SB* of the link target is a part of article *B*, hence they are related, and it will greatly affect the results. To solve this problem, we adopt the following assumption.

Assumption: If segment *SA* has a link to segment *SB*, then *SB* should be the most related segment with *SA* in article *B*, and the other segments in *B* are just slightly related with *SA*. In other words, if we rank the similarities between *SA* and each segment in article *B*, then the similarity between *SA* and *SB* should be a prominent outlier. If segment *SA* is linked to article *B*, then all the segments in *B* should be slightly related with *SA*, but there is no obvious outlier.

Based on this assumption, we construct our feature set to incorporate statistical features on segment similarity distributions. We introduce the following features from three aspects.

Range of similarity distribution

Similarities between pairs of segments are measured by segment vectors based on either TIFIDF or LDA. One article consists of multiple segments, so we obtain a vector of segment similarities for one article. We note that similarity values are so diverse between articles. Therefore, we characterize these similarity vectors by descriptive statistics of similarity distributions of the following: The number of the segments in the article, and maximum, minimum and mean of the similarity values of the segments in the article.

Dispersion of similarity distribution

Based on our assumption, if a link points to a segment there will be at least one segment in the link target article which is highly similar to the link source. In an ideal situation, if the link points to an article title, not a specific segment, then the similarities of all the segments in the article toward the link source segment are close between each other, so the dispersion of these similarity values is small. Thus dispersion of similarities within one article is an important clue for determining whether the link should be on the article level or segment level. Thus we introduce the following features: variance, standard deviation, and coefficient of variations. These features are derived from statistical properties of the distribution of segment similarities in one article.

Outliers

According to our assumption, if a link points to a segment title, then it is more likely that there exists an outlier segment, having an outstandingly larger similarity than other segments. We adopt the conventional concept of outliers such that, if the difference between one segment and mean is more than the double of the standard deviation, then we believe the segment is an outlier. If the difference is more than triple of the standard deviation, the segment is a large outlier. In this paper, we use the number of large outliers in one article as a feature.

3.5 Filtering

In Wikipedia the number of link point to segment and point to article title is imbalance. Our previous method used under-sampling method to balance the training data. In this paper, we first train a logistic regression model to predict the probability of the link should be point to segment. Then we filter the test data by filtering the sample with low probability, and using RF model to predict the last samples.

[1] https://en.wikipedia.org/wiki/Help:Link#Section_linking_.28anchors.29

3.6 Prediction

We train a classifier based on the above features to predict whether a target link points to article title or a specific segment. The features we introduced are statistical measures on similarity distributions, which do not have obvious linear relationships. So we utilize the nonlinear classifier model random forest [6] as our classifier.

In most machine learning tasks, ensemble model is much better than unique model. We propose a stacking model to replace unique random forest, only using random forest as the top classifier in the stack top.

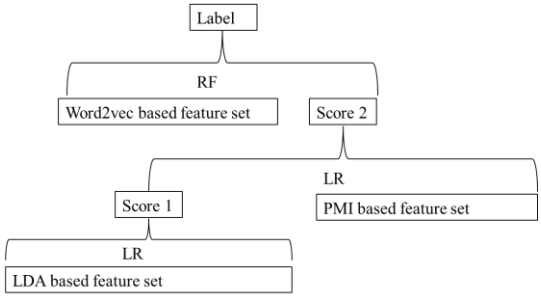

Fig.3. Stacking model

As Fig3 shows, we have 3 features set, we stack these 3 feature set and regard the lower layer result as the higher layer's new feature.

When the target link is predicted as pointing to a segment, we need to determine which segment should be pointing to. In this step, instead of simply selecting the most similar segment as the result, we also require that the selected segment must be an outlier in the similarity distribution.

4. Experiment Evaluation

4.1 Dataset

Featured articles are considered to be high-quality articles in Wikipedia, and they are well organized and maintained. We adopt links from featured articles as our golden standard, assuming that they are appropriately given, so that these links point to segments when there are specifically relevant segments in the target articles. We randomly selected 1000 featured articles as our dataset.

Table 1. Dataset infomation

Dataset	Count
Articles	1000
Segments	7478
Links pointing to segments	1689
Links pointing to article titles	153918

There are totally 1689links pointing to segments and 153918links pointing to article titles. We use these links as our reference data. The ratio between the segment links and article links is nearly 1:100. This ratio is rarely observed in most of articles, because the average number of links in one article is 41, so most of articles just have article-level links. We divide the dataset into small subsets, to control the ratio between positive and negative samples, where positives are segment-level links. We use the random forest classifier for our prediction.

4.2 Experiment preparement

In the step of presenting segment, in word2vec model, we use the whole wikipedia as the training data to train the model and we set the dimension as 50. And we also try the model result trained by others, Because one segment usually contains several words, so we use the sum of all word vectors of words in segments to present the segment vector.

From the dataset, we observe that the ratio between positive and negative samples is imbalanced. We need a careful setup for training, because the classifier tries to adjust the parameters to put all samples into the correct classes. If negative samples are overwhelming majority, the positive samples could be regarded as invalid values by the classifier. There are two approaches to tackle this problem. The first one is randomly sampling negatives to balance positives and negatives, and then train using this sampled set. But its disadvantages are obvious. This sampling process will lose a large number of effective data. The lost negative samples will cause learning errors, degrading precision.

Another solution is resampling positives until positives and negatives are balanced. But it can cause over fitting easily, and it does not help the classifier to learn positive samples, because resampling does not increase new positive samples, instead just balancing the dataset by repeating addition of small positive samples.

In this experiment, we use sampling negatives to balance the dataset. We randomly sample negatives several times, and train the classifier.

We do experiments on unique random forest as the classifier and stacking model as the classifier. We set the default parameters 50 trees in the forest and the deep is 5. For the filter we utilize logistic regression which penalty is L2 to avoid over fitting. We set filter threshold as 0.5.

On the prediction results, we calculate average precision, recall, and F1-score.

4.3 Feature importance

Before we test in the dataset, first we have to measure whether our features are effective to distinguish the posi-

tive class. We use correlation analysis methods to test relationship between each feature and the reference data. Mann-Whitney is a nonparametric test of the null hypothesis that two samples come from the same population against an alternative hypothesis, such that a particular population tends to have larger values than the other. The test results are shown in Table 2.

From this test we can find max similarity, mean similarity, variance, standard deviation, small similarity outlier are related to the reference classification. But in our assumption, the large similarity outlier should be related. For further analysis, we performed Two-Sample Kolmogorov-Smirnov test. This is a nonparametric test for equality of continuous, one-dimensional probability distributions that can be used to compare a sample with a reference probability distribution (one-sample K–S test), or to compare two samples (two-sample K–S test). The results are shown in Table 3.

Table 2. Mann-Whitney test on features

Feature	Significance	decision
Element count	0.454	retain
Max similarity	0.018	reject
Min similarity	0.423	retain
Variance	0.023	reject
Mean similarity	0.002	reject
Coefficient of variation	0.611	retain
Standard deviation	0.023	reject
Small similarity outlier	0.010	reject
Large similarity outlier	0.079	reject

Table 3. Kolmoogorov-Smirnov test on features

Feature	Significance	Decision
Element count	0.227	retain
Max similarity	0.000	reject
Min similarity	0.636	retain
Variance	0.000	reject
Mean similarity	0.001	reject
Coefficient of variation	0.714	retain
Standard deviation	0.000	reject
Small similarity outlier	0.821	retain
Large similarity outlier	0.612	retain

We can see in Table 3 that the small similarity outlier is not very effective, but from these two tests, we can see variance and standard deviation are strongly effective. So the test results support our assumption. These importance results indicate that our features are effective and the classifier is expected to distinguish segment links via these features.

4.4 Baseline

Previous researches are all focusing on linkability to article titles, not targeted to segments. Since there is no preceding work, we choose our baseline based on a simple idea that if words in a link source occur in an article title, then the link should point to the article title. Otherwise, if words in a link source occur in a segment, then the link should point to the segment. If words occur in multiple segments, then the most frequent segment is considered as the target of the link.

4.5 Classification result

Our first step is to determine whether a link should point to an article title or segment. To test our method in different ratios of positives and negatives, we controlled the ratio from 1:1 to 1:100. In each dataset, we use half data to train and the other half data to test and all the samples are random select from the dataset. Results are shown in Tables 4 to 7.

Table 4. Pos:Neg=1:1

Pos:Neg=1:1	Precision	Recall	F1
LDA feature	61.0%	82.4%	70.2%
W2V feature	65.0%	76.5%	71.0%
PMI feature	62.0%	76.3%	68.4%
LDA + W2V feature	53.8%	82.3%	65.1%
LDA + PMI features	53.5%	88.3%	66.6%
W2V + PMI features	63.2%	70.5%	66.7%
LDA+W2V+PMI features	72.2%	76.5%	74.3
Random result	50%	50%	50%
Baseline	61%	57%	59%

Table 5. Pos:Neg=1:10

Pos:Neg=1:10	Precision	Recall	F1
LDA features	13.0%	70.5%	22.0%
W2V features	14.3%	76.5%	24.1%
PMI features	12.0%	74.1%	19.9%
LDA + W2V features	14.4%	76.5%	24.2%
LDA + PMI features	12.1%	76.5%	21.0%
W2V + PMI features	11.7%	82.4%	20.4%
LDA+W2V+PMI features	13.2	88.2%	23.0%
Random result	10%	50%	16.6%
Baseline	12.8%	46.2%	20.0%

Table 6. Pos:Neg=1:50

Pos:Neg=1:50	Precision	Recall	F1
LDA features	2.3%	48.9%	4.5%
W2V features	2.6%	77.8%	5.1%
PMI features	2.2%	66.6%	4.3%
LDA + W2V features	2.5%	75.6%	4.9%
LDA + PMI features	2.3%	67.6%	4.4%
W2V + PMI features	2.3%	73.3%	4.4%
LDA+W2V+PMI features	2.6%	57.8%	5.0%
Random result	2%	50%	3.8%
Baseline	2.1%	45%	4.0%

Table 7. Pos:Neg=1:100

Pos:Neg=1:100	Precision	Recall	F1
LDA features	1.2%	53.2%	2.3%
W2V features	1.2%	56.1%	2.4%
PMI features	1.0%	61.0%	1.9%
LDA + W2V features	0.9%	53.9%	1.8%
LDA +PMI features	0.9%	56.0%	1.7%
W2V + PMI features	0.9%	48.0%	1.7%
LDA+W2V+PMI features	0.9%	65.2%	1.8%
Random result	1%	50%	1.9%
Baseline	1.1%	41%	2.1%

The second step is to determine which segment is most relevant to the link source. In our method, for a link in the source segment the most similar segment is selected, based on cosine similarities between two segments. After that, we calculate whether the most similar segment is an outlier in the entire similarity distribution. If the outlier is satisfied, we select the segment as our prediction result. The result is shown in Table 8.

Table 8. Predicting the most related segment

Average segment count=8	Accuracy
LDA features	21.3%
W2V features	32.0%
PMI features	40.0%
Random result	12.5%

4.6 Discussion

From Table 4 to Table 8, the results show that our method is most accurate to predict links pointing to segments. Our method performs better than the baseline. Correlation analysis shows that our statistics features on similarity distributions are effecitive, to capture patterns when segment-level links occur.

Our method works well when the positive and negative samples are balanced to 1:1. However, even though we already use the sampling method to balance the dataset, the influence of imbalance data is still strong. We can find that when the ratio of postives and negatives is more than 10:1, precision goes down significantly. We still need to improve to deal with imbalanced datasets, since most of the real world data is imbalanced.

As our dataset is imbalanced, we can observe that the sampling process is very important in our method. Table 9 shows that if we ignore the sampling process, the F1 score decreases a lot.

Table 9. Pos:Neg=**1:10**

Pos:Neg=1:10 W2V features	Precision	Recall	F1
under sampling	14.3%	76.5%	24.1%
Without under sampling	10.5%	46%	17.1%

As Table 10 shows, the results of LDA features and W2V (Google news) features are basically same, word2vec model get higher precision but lower recall. And the model trained by Wikipedia performs better than trained by Google news, it maybe because our experiment data is also from Wikipedia. But in selecting the most related segment process, our method performs best.

We also compare the result of filtering and without filtering. The Table 11 shows that filtering performs well in our task. It increases precision a lot, but decreases recall less, and finally increasing the F1 score.

Table 12 shows when we select different filter threshold, the result will change a lot. If the filter is strict, false positive samples increase a lot and recall decrease. If the filter is general, the false negative samples will increase, it will decrease the precision value. In our experiment, we set the threshold as 0.5, it gets best result.

Table 10. Pos:Neg=1:10

Pos:Neg=1:10	Precision	Recall	F1
LDA features	13.0%	70.5%	22.0%
W2v(Google news)	13.2%	68.2%	22.1%
W2v(Wikipedia)	14.3%	76.5%	24.1%
Baseline	12.8%	46.2%	20.0%

Table 11. Pos:Neg=**1:10**

Pos:Neg=1:10	Precision	Recall	F1
With filtering process	16.5%	64.4%	27%
Without filtering process	14.3%	76.5%	24.1%

Table 12. Pos:Neg=**1:10**

Pos:Neg=1:10	Precision	Recall	F1
Without filtering process	14.3%	76.5%	24.1%
Filter threshold <0.6	25.0%	11.7%	17%
Filter threshold <0.5	22.8	47.1%	31%
Filter threshold <0.4	15.5%	52.9%	24%
Filter threshold <0.3	18.0	52.9	26%

Besides set a filter, we also stack the features into a stacking model. Usually the top layer should be the most accuracy features so we set W2V features in top layer. Table 13 shows the result of stacking model. Compare the single features the stacking model actually improve the result

Table 13. Pos:Neg=**1:10**

Pos:Neg=1:10	Precision	Recall	F1
Only W2V features	14.3%	76.5%	24.1%
Top=W2V features Mid=LDA features Bottom=PMI features	14.8%	82.4%	25.2%
Top=W2V features Mid=PMI features Bottom=LDA features	14.3%	82.3%	24.3%

5. Conclusion and future work

In this paper, we proposed an text similarity based algorithm to determine whether links in Wikipedia articles point to a related segment, or article title. We believe that our research is the first work of link detection on segment level. Our approach is combining the LDA model with MLE with a nonlinear combination. But it can still be improved, we currently use document length to smooth the obvious part. Further improvements over parameter optimizations can be expected. We also compare LDA based method with word2vec model and PMI based method. We introduced statistical features on segment similarity distributions, to train a classifier for predicting whether a link points to segments or article titles. Our method performs well when the dataset is balanced. In future work, we plan to design a strong feature to improve accuracies when the dataset is much imbalanced. We also try to utilize category hierarchies when selecting the most related segment.

6. Reference

1. Adafre, S. F., & de Rijke, M. Discovering missing links in Wikipedia. In Proceedings of the 3rd international workshop on Link discovery pp. 90-97(2005).
2. Besnik, F., Katja, M., Avishek, A.: Automated News Suggestions for Populating Wikipedia Entity Pages. In: Proc. CIKM '15 Proceedings of the 24th ACM International on Conference on Information and Knowledge Management pp 323-332(2015)
3. Blei, D.M., Moreno, P.J.: Topic segmentation with an aspect hidden markov model. In: Proceedings of SIGIR (2001)
4. Blei, D.M., Ng, A.Y., Jordan, M.J.: Latent Dirichlet allocation. Journal of Machine Learning Research 3, 993–1022 (2003)
5. Bouma, G. (2009). Normalized (pointwise) mutual information in collocation extraction. *Proceedings of GSCL*, 31-40.
6. Breiman L. Random forests[J]. Machine learning, 45(1): 5-32 (2001)
7. Knoth, P., Novotny, J., & Zdrahal, Z. (2010, August). Automatic generation of inter-passage links based on semantic similarity. In Proceedings of the 23rd International Conference on Computational Linguistics (pp. 590-598).
8. Lavrenko, V., Croft, W.B.: Relevance-based language models. In: SIGIR 2001,pp. 120–127 (2001)
9. Liu, X., Croft, W.B.: Cluster-based retrieval using language models. In: Proc. 27th International ACM SIGIRConf. Research and Development Information Retrieval, pp. 186–193(2004)
10. Mihalcea, R., Corley, C., & Strapparava, C. Corpus-based and knowledge-based measures of text semantic similarity. In *AAAI* (2006, July). (Vol. 6, pp. 775-780).
11. Milne, D., Ian, H.W.: An Effective, Low-Cost Measure of Semantic Relatedness Obtained from Wikipedia Links. In: Proc. AAAI Workshop on Wikipedia and Artificial Intelligence: an Evolving Synergy, Chicago, pp. 25–30 (2008)
12. Milne, D., Ian, H.W.: Learning to link Proceeding. In: CIKM '08 Proceedings of the 17th ACM conference on Information and knowledge management, pp. 509-518(2008)
13. Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, Jeffrey Dean: Distributed Representations of Words and Phrases and their Compositionality, NIPS '13, Pages 3111–3119 (2013)
14. Tomas Mikolov, Kai Chen, Greg Corrado, Jeffrey Dean: Efficient Estimation of Word Representations in Vector Space, ICLR '13 Proceedings of Workshop at International Conference on Learning Representations (2013)
15. Xing, W., Croft, W.B.: LDA-Based Document Models for Ad-hoc Retrieval. In: Proc.29th ACM SIGIR Conf., pp. 178–185 (2006)
16. Wang, R., & Iwaihara, M. (2016). Suggesting Specific Segments as Link Targets in Wikipedia. In Digital Libraries: Knowledge, Information, and Data in an Open Access Society (pp. 394-405).
17. Wang R, Wu J, Iwaihara M. Finding co-occurring topics in wikipedia article segments. International Conference on Asian Digital Libraries. Springer International Publishing, pp.252-259(2014).
18. Zhai, C., Lafferty, J.: A study of smoothing methods for language models applied to ad hoc information retrieval. In: Proc. 24th ACM SIGIR 2001, pp. 334–34 (2001)
19. Zhang, J., & Kamps, J.: A content-based link detection approach using the vector space model. In International Workshop of the Initiative for the Evaluation of XML Retrieval pp. 395-400(2009)