NCU IISR System for NTCIR-12 MobileClick2

Wen-Bin Han
National Central University,
Taoyuan, Taiwan
b84414@gmail.com

Hung-Hsiang Wang
National Central University,
Taoyuan, Taiwan
whh99u@gmail.com

Richard Tzong-Han Tsai*
National Central University,
Taoyuan, Taiwan
thtsai@csie.ncu.edu.tw

ABSTRACT

This paper describes our approach to the NTCIR-12 MobileClick task. First of all, we do some extra process on the baseline. Next, we try to use a totally different method from baseline which is machine learning. Finally, tune the two types into better situation and apply them to test data. Our system achieves an nDCG@3 score of 0.7415, nDCG@5 score of 0.764, nDCG@10 score of 0.8059, nDCG@20 score of 0.8732 and a Q-measure score of 0.9004, outperforming the baseline a little bit.

Team Name

IISR

Subtask

iUnit Ranking Subtask (English)

Keywords

Information Ranking, Multi-aspect iUnit Score, MobileClick, Machine Learning

1. INTRODUCTION

In this paper, we describe our approach to the MobileClick iUnit Ranking Subtask. This subtask is to rank a set of pieces of information(iUnits) based on their importance for a given query. In our approach, we first try the baseline formula provided by the Organization and improve it with some extra features such as emphasizing ranks of pages and removing smoothing. In addition to the baseline, we attempt to use Machine Learning, which is totally different from the concept of baseline.

The reminder of this paper is organized as follows. Section 2 describes our methods and implementation. Section 3 describes the evaluation results and discusses error analysis Section 4 concludes this paper.

2. METHOD

In this section, we describe our methods, which consist of two modules: Improved-Baseline and Machine Learning. Before acquiring the importance of each iUnit by the methods mentioned above, we remove the tags and advertisers in html documents and extract content of html pages first in order to make the following work easier. Then, we take these methods:

2.1 Improved-Baseline

The concept of baseline is to calculate how importance the word is. The baseline is:

$$LogOdds(u) = \sum_{w \to w_u} (P(w|q) - P(w|o))$$

$$P(w|q) = \ln\left(\frac{n_{\{D_q,w\}} + smoothing}{n_{\{D_q\}} + w_{num}}\right)$$

$$P(w|o) = \ln\left(\frac{n_{\{D_o,w\}} + smoothing}{n_{\{D_o\}} + w_{num}}\right)$$

$n_{\{D_q\}}$: number of words in the pages for the given query.
$n_{\{D_q,w\}}$: number of the word in the pages for the given query.
$n_{\{D_o\}}$: number of words in the pages for the other queries.
$n_{\{D_o,w\}}$: number of the word in the pages for the other queries.
w_num: number of different type of words in all pages.
$smoothing$: default for 1.

If some of words of iUnit appears more frequent in its documents for the given query than in other documents for the other queries, it represents those words are important (highly relevant) for this query. Based on the notion of the baseline, we take some process as follows to make the result more precise.

2.1.1. Natural Language Processing (NLP)

Because the concept of baseline is to calculate how importance the word is, there are many tenses for a verbs in documents, if we intend to count numbers of words, we need to do stemming so as to get the correct number. Besides, for the reason that some of the words in iUnits and html documents are common and don't have any specific meaning such as "the", "be", and "his", we remove those out and retain those meaningful words.

2.1.2. Filtering Infrequent Words

Counts of some kind of words are not up to the threshold, which means those words are too rare to be involved.

2.1.3. Making negative scores to zero

Each iUnit will get its score calculated by using the baseline; however, there will be some scores are negative. Ideally, the score should not be reduced by words which are not much important (more frequently appear in other documents). Instead, they should not matter.

2.1.4. Take Mean

Observing the relation between the predict answers and the iUnit, we notice that the longer iUnit is, the higher OddsRatio is, which means comparing all iUnits in different length is unfair. Therefore, after calculating OddsRatio, divide the scores by the count of words of iUnit.

2.1.5. Ranks of Page

Html page dataset is derived from Bing search, which possesses its own searching algorithm. Based on the rank sorted by Bing, we give it a weight. Word appears in top pages will occupy more score of OddsRatio, vice versa.

$$\frac{freq_{word\ of\ page}}{Rank_{page}^{0.01}}$$

$freq_{word\ of\ page}$: number of the word shown in specific page.
$Rank_{page}^{0.01}$: the order of the pages for the given query.

2.1.6. Do not use smoothing

Before using the baseline, we need to know what will happen if we remove smoothing. Maybe the trend will be more probability to reflect the truth. Therefore, we take a try to see what will happen if there is no smoothing done.

$$LogOdds(u) = \sum_{w \to w_u} (P(w|q) - P(w|o))$$

$$P(w|q) = \ln(\frac{n_{\{D_q,w\}} + 1}{n_{\{D_q\}} + 1})$$

$$P(w|o) = \ln(\frac{n_{\{D_o,w\}} + 1}{n_{\{D_o\}} + 1})$$

$n_{\{D_q\}}$: number of words in the pages for the given query.
$n_{\{D_q,w\}}$: number of the word in the pages for the given query.
$n_{\{D_o\}}$: number of words in the pages for the other queries.
$n_{\{D_o,w\}}$: number of the word in the pages for the other queries.

2.2 Machine Learning

Aim to make machine to predict the answer through the relation between query and iUnits, we trying to use machine learning, which is a different method from the baseline with the tool Liblinear-SVM. We classify our method into two category: pointwise and pairwise, and depict them individually in detail.

2.2.1. Pointwise

More features there are, more precisely it can predict. We use the Word2Vec dictionary as the features, which was pre-trained by Wikipedia 2014 corpus and English Gigaword fifth edition corpus. There are four kinds of dimension of word2vec: 50, 100, 200, and 300. For balanceing consuming time on calculating and precision, we chose the 100 dimension one. Taking the weight of training data as label, we set 101 values as features which are composed of the 100-dimension vector and the OddsRatio scores from the Improved-Baseline method. Each word has its 100-value vector, and closer the distance between two words, more frequently they appear together. First of all, sum the vectors of words of query and iUnit for each dimension, and each query or iUnit has its own vector. Next, subtract the vector of iUnit from the vector of query, and the new vector represents the relation between iUnit and query. For the reason knowing how the method is, we work on training data with 5 folds, which splits the training data into five parts, four for training and one for predicting, and assemble the five parts of predicting result into final predict answer.

2.2.2. Pairwise

The methods we have tried are all about the relation between query and iUnit; nevertheless, iUnits should simultaneously be ranked by relation between two iUnits for the same query. Same as the Pointwise method, we also use the 100-dimension word2vec dictionary, and get the distance between a query and an iUnit (a pair). In the beginning, we want to compare two pairs (query with iUnit A and query with iUnitB) which is closer, so our features include distances between iUnit. In SVM, training data format is several features followed by one label. Our labels divide into three kind: 0, 1, and 2, which mean the weight of iUnit A given by organizer is larger than, equals, and is smaller than iUnit B. There are 201 features for each instance, the first 100 features are 100-dimension vector of iUnit A minus query and the next 100 features are 100-dimension vector of iUnit B minus Query. The last feature is the OddsRatio of iUnit A minus the OddsRatio of iUnit B gained from 2.2.1 Improved-Baseline. Depending on the predict answer, we can get the eventually ranked list.

3. EVALUATION

In order to acknowledge which method is suitable for the task, we analyze our methods individually and conduct more experiment on promising ones.

3.1 Improved-Baseline

3.1.1. Natural Language Processing (NLP)

We found that doing NLP will make predict more uncertainty. It is conferred that removing stopwords may keep the whole percentage not measured. Also, the Stopword list perhaps is not proper.

3.1.2. Filtering Infrequent Word

Although we have filtered out infrequent words, there is not much progress in ranking. The percent of infrequent word is not really high.

3.1.3. Making Negative scores to zero

With this method, the Q-measure will get higher. We referred that if there are some words more frequently appearing in other query dataset, it should not reduce the OddsRatio score; instead, it shouldn't matter. Therefore, negative score ought not to appear.

	nDCG@3	nDCG@5	nDCG@10	nDCG@20	Q
Improved-Baseline (3.1.2 combine 3.1.6)	0.7415	0.764	0.8059	0.8732	0.9004
Pairwise with three class(>, <, =)	0.7499	0.7661	0.8056	0.8727	0.8977

Table1: Evaluation of two methods. nDCG@k: the top k-th rank of nDCG score.

3.1.4. Take Mean

As the ground-truth, the weight document implies that iUnit containing more information will have more votes. According to the rule, if we concentrate on word rather than sentence, then we just get a bad perfomance.

3.1.5. Ranks of Page

After working on the training data, there is not much difference in using it. We deduce that tokens in iUnits are shown evenly in overall pages than centralized in some pages.

3.1.6. Do not use smoothing

Without smoothing, making negative score to zero doesn't influence much; instead, removing infrequent words improves it.

After several experiences on training data, we figure out that do not use smoothing with filtering infrequent words can get more precise rank on training data. Wherefore, we take this module to test data.

3.2 Machine Learning

3.2.1. Pointwise

Through machine learning, our Q-measure score is near to the score of Organization, but not exceeds it. We consider that the word2vec dictionary is not trained by the same kind of article; thence, deviation will occur. Besides deviation, prediction also has chance to be wrong. We also have done NLP and taken mean of each dimension score when using Pointwise method; however, these did not make the prediction better.

3.2.2. Pairwise

The score from three labels model surpass the score from two labels model and approximately equals to the score of Organization's method. However, we don't have much time to improve it, or it will have opportunity to be better.

4. CONCLUSION

We describe our approach to the NTCIR-12 MobileClick task in this paper. Our approach comprises Improved-Baseline and Machine Learning. For Improved-Baseline, we can get higher score with filtering infrequent words and removing smoothing. For Machine Learning, the pairwise method, which is comparing two pairs of iUnit and query with three labels, is better than the pointwise method.

5. REFERENCES

[1] Jeffrey Pennington, Richard Socher, Christopher D. Manning. 2014. GloVe: Global Vectors for Word Representation.
DOI= http://nlp.stanford.edu/projects/glove/

[2] Chia-Tien Chang, Yu-Hsuan Wu, Yi-Lin Tsai, Richard Tzong-Han Tsai. Improving iUnit Retrieval with Query Classification and Multi-Aspect iUnit Scoring: The IISR System at NTCIR-11 MobileClick Task. Dec, 2014.
DOI=http://research.nii.ac.jp/ntcir/workshop/OnlineProceedings11/pdf/NTCIR/MobileCLICK/02-NTCIR11-MOBILECLICK-ChangC.pdf

[3] Chih-Chung Chang and Chih-Jen Lin. LIBSVM -- A Library for Support Vector Machines.
DOI = https://www.csie.ntu.edu.tw/~cjlin/libsvm/