Contextual Advertising for Web Article Printing

Shengwen Yang, Jianming Jin, Parag Joshi, Sam Liu

HP Laboratories
HPL-2010-79

Keyword(s):

printed ad, web printing, article extraction, contextual advertisement matching

Abstract:

Advertisements provide the necessary revenue model supporting the Web ecosystem and its rapid growth. Targeted or contextual ad insertion plays an important role in optimizing the financial return of this model. Nearly all the current ad payment strategies such as "pay-per-impression" and "pay-per-click" on web pages are geared for electronic viewing purposes. Little attention, however, is focused on deriving additional ad revenues when the content is repurposed for alternative mean of presentation, e.g. being printed. Although more and more content is moving to the Web, there are still many occasions where printed output of web content or RSS feeds is desirable, such as maps and articles; thus printed ad insertion can potentially be lucrative.

In this paper, we describe a cloud-based printing service that enables automatic contextual ad insertion, with respect to the main web page content, when a printout of the page is requested. To encourage service utilization, it would provide higher quality printouts than what is possible from current browser print drivers, which generally produce poor outputs - ill formatted pages with lots of unwanted information, e.g. navigation icons. At this juncture we will limit the scope to only article-related web pages although the concept can be extended to arbitrary web pages. The key components of this system include (1) automatic extraction of article from web pages, (2) the ad service network for ad matching and delivering, and (3) joint content and ad printout creation.

External Posting Date: August 6, 2010 [Fulltext] Approved for External Publication
Internal Posting Date: August 6, 2010 [Fulltext]
To be published and presented at ACM Document Engineering Conference, 2010, Manchester, UK, September 21-24, 2010

Contextual Advertising for Web Article Printing

Shengwen Yang, Jianming Jin
HP Labs China
No.1 Zhong Guan Cun East Road
Beijing 100084, China
{sheng-wen.yang,jian-ming.jin}@hp.com

Joshi Parag, Sam Liu
HP Labs Palo Alto
1501 Page Mill Road
Palo Alto, CA, 94304, USA
{parag.joshi,sam.liu}@hp.com

ABSTRACT
Advertisements provide the necessary revenue model supporting the Web ecosystem and its rapid growth. Targeted or contextual ad insertion plays an important role in optimizing the financial return of this model. Nearly all the current ad payment strategies such as "pay-per-impression" and "pay-per-click" on web pages are geared for electronic viewing purposes. Little attention, however, is focused on deriving additional ad revenues when the content is repurposed for alternative mean of presentation, e.g. being printed. Although more and more content is moving to the Web, there are still many occasions where printed output of web content or RSS feeds is desirable, such as maps and articles; thus printed ad insertion can potentially be lucrative.

In this paper, we describe a cloud-based printing service that enables automatic contextual ad insertion, with respect to the main web page content, when a printout of the page is requested. To encourage service utilization, it would provide higher quality printouts than what is possible from current browser print drivers, which generally produce poor outputs - ill formatted pages with lots of unwanted information, e.g. navigation icons. At this juncture we will limit the scope to only article-related web pages although the concept can be extended to arbitrary web pages. The key components of this system include (1) automatic extraction of article from web pages, (2) the ad service network for ad matching and delivering, and (3) joint content and ad printout creation.

Categories and Subject Descriptors
H.3.5 [**Information Storage and Retrieval**]: Online Information Service – *commercial services, web-based services.*

General Terms
Documentation, Experimentation

Keywords
printed advertisement, web printing, contextual ad matching

1. INTRODUCTION
Deriving revenue from contextual ads inserted into web pages is the primary sustainable business model for supporting the Web ecosystem and its rapid growth. There are several ways in which advertisers can be charged when ads are placed in a web page. There are the "pay-per-impression" ads where they are simply displayed onto the web page, and there are the "pay-per-click" ads where the advertiser pays only after each click, which usually leads to the advertiser's site. The publishers (content owner) and the ad service provider would share the ad revenue.

As a vertical application to web printing, there is an untapped but potentially lucrative market of targeted advertisement and discount coupon placement when the web page is printed. Even though traditional personal and commercial printed contents are moving to the Web, we still believe there will be substantial amount of printing of web content in the future, such as maps and articles. As most of the current online advertisement models are based on ad networks for ad insertion and distribution on web pages, we propose to develop a cloud-based printing service which enables the automatic repurposing of web content for printing as well as the contextual ad insertion when a printout is requested, thus realizing the additional ad revenue for printing service providers from the web printing.

Another motivation of repurposing web content for printing is that web printing experience is generally quite unsatisfactory. It suffers from the objective that most pages are created for display purpose and to interact with the users, so the browser is lacking the technology to produce well formatted printout from complex HTML/CSS files. As a result, the printouts usually contain ill formatted pages with non-informative content such as navigation menus. There are, however, some article sites that provide a hyperlink that leads to a print-friendly version of the page, but many print-worthy sites still lack such feature. To encourage end users to access a cloud-based printing service (so ads can be inserted), the service would provide a much higher quality printout than from the browser. The high quality printout is created by extracting only the informative content of the page and aesthetically laid out along with newly inserted ads to produce a document-style output, e.g. PDF files, as illustrated in Figure 1.

Note that the inserted ads or coupons are new and not related to the original ads on the Web page. This is essentially a secondary insertion and is controlled by the print service provider. Thus the new ad revenue is also collected by the print service provider, and possibly shared with the publisher depending on the business model. Since articles on the Web are good candidates for printing, this paper limits the focus to article pages although the concept can be extended to other types of web pages as well.

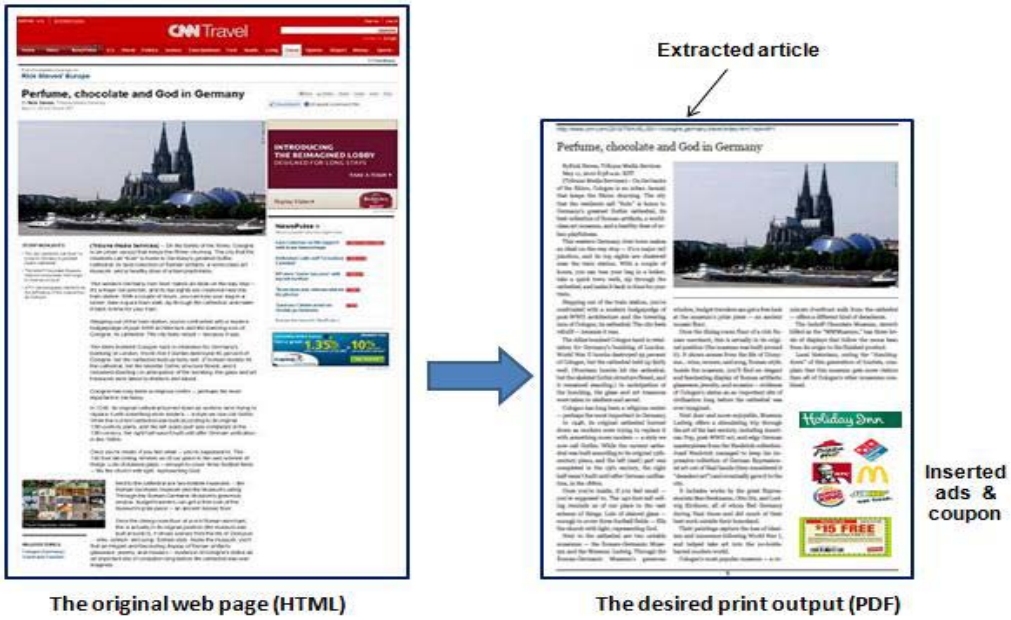

Figure 1. Example of article extraction and ad insertion for printing

The key technologies for this service include: (1) automatic extraction of article from web pages, (2) ad network for ad matching and delivering, and (3) combined content and ad printout creation. The service requires an end-to-end solution in order to control the entire pipeline, which includes fetching web page HTML/CSS files, extracting the article, inserting the ads, and creating the final document to be printed. The implementation of the service can be either server or client centric, depending on how the tasks are distributed between the server and the client. Other potential applications of this service include free or subsidized mobile printing, discounted commercial brochure printing for direct mailing, magazines, newsletters, etc. End users would find insertion of targeted coupons and advertisements in such printing scenarios desirable.

2. ARCHITECTURE OVERVIEW

The cloud-based printing service requires a client plug-in and backend server to perform various tasks that are necessary for delivering the final repurposed document to the printer. There are three key analysis components of the system that require significant computations: (1) automatic article extraction from Web pages, (2) article semantic extraction for ad selection, and (3) final document layout and creation. Both components (1) and (3) can be executed either on the client or the server, but component (2) must be executed on the server since it involves the ad database. There are several possible implementations depending on the service requirements, such as resource costs, e.g. servers, software maintainability, and revenue return from ads. We will describe two possible configurations here: the server-centric solution where all of the analysis components are performed by the server and the client-centric solution where some of the analysis is shifted to the client side.

2.1 Server-Centric Solution

In the server-centric implementation, all the analysis tasks are performed by the backend server, as illustrated in Figure 2. First, when a print request is initiated by a user, the client needs to send to the server the URL of the page to be printed. Once the server receives this information, it would perform article extraction, article semantic extraction, ad or coupon selection, and the final printed document creation. Once these tasks are completed, it returns this document to the client for printing.

Since the server-centric solution requires all analysis to be done by the backend server, the server however may not be able to provide real-time response during high traffic. Instead of scaling up the performance by adding more servers which may render the service cost prohibitive, the server can archive the prior analysis result of the URL to be reused for future requests. Another enhancement to the scaling solution is to pre-crawl targeted websites to analyze and archive article Web pages. The advantage of server-centric implementation is that the client plug-in can be kept fairly simple – no update of the plug-in is necessary when better analysis technology is deployed in the network.

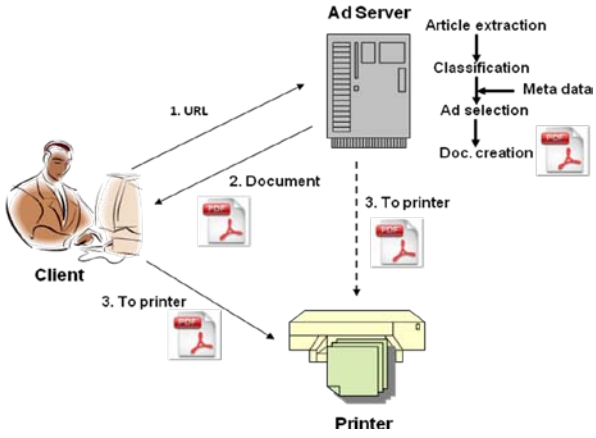

Figure 2. Server-centric architecture

2.2 Client-Centric Solution

As discussed earlier, one potential drawback of server-centric solution is the cost of scaling up the service to accommodate high number of requests. Instead of using the aforementioned archiving and pre-crawling schemes to circumvent the problem, an alternative solution is to dedicate some of the analysis tasks to the client, as illustrated in Figure 3. Both the article extraction and the final document creation can be done at the client end. Under this scenario, upon the initiation of the print request, the client plug-in would extract the article from the Web page, and then the article content would be delivered to the ad server for semantic extraction and ad selection, which is then sent back to the client for printing. This implementation only requires the server to perform the article semantic analysis and ad selection.

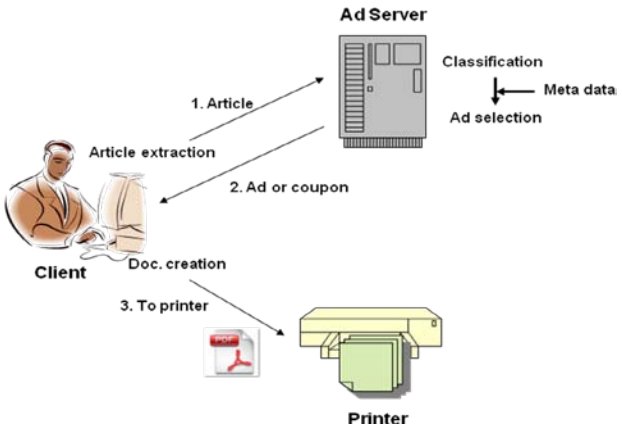

Figure 3. Client-centric architecture

3. AUTOMATIC ARTICLE EXTRACTION FOR WEB PRINTING

Web pages are generally created without their printability in mind. The main content of interest, e.g. the article itself, is mixed with ads and other auxiliary information related to the interworking of the Web, such as navigation menus. Current web printing usually results in undesirable outputs, plagued by poor page layout, too many pages generated, and unwanted content, e.g. ads and icons. The problem mainly stems from the use of complex HTML/CSS files to organize information on web pages, but evidently the browser print engine has difficulty in rendering the print pages with acceptable quality. Ideally, only the main informative content should be extracted and aesthetically laid out to produce a document or magazine style output, e.g. PDF files, as illustrated in Figure 1.

To improve printing of web articles, there are solutions proposed to address this particular problem [1], all of which apply the DOM (Document Object Model) data structure for content analysis and extraction. Our solution for article extraction is similar to the method described in [1]. In a nutshell, the article extractor first creates the DOM from HTML/CSS, from which it is analyzed to cluster contiguous paragraphs together. The cluster with the largest number of paragraphs, in terms of character count, is usually chosen as the article text block. Within this text block, additional analysis is performed to prune out unwanted content, e.g. icons and link-lists, and to discriminate between ad and article images. The outcome of the article extractor should only consist of the following components: the article text body, title, associated relevant images and captions, and possibly the site logo if available. The article extractor can be implemented either at the client or the server side depending on the platform architecture choice. Please refer to the references mentioned above for details.

4. CLASSIFICATION OF WEB ARTICLES FOR AD MATCHING

The ad insertion system relies on a database of advertisements and coupons along with the necessary meta-data or features for contextual matching [2][3]. The database has advertisements organized into large set of categories. Each category is pre-determined manually based on the available ad pool and web articles such as, restaurants, movies, grocery, etc, in addition to meta-data related to location or demographics (if available). The relevance matching of advertisement and article is not just measuring the content theme between the article and ads, but also considering the location and demographic information, if available. The advertisements and coupons are pre-classified into these categories based on discriminative keywords using linguistic analysis from a large training set, as illustrated in Figure 4. In this paper, we present some experimental results on article classification from real news sites for advertisement matching. A more sophisticated contextual advertisement matching model is still under development in which more complex set of factors is included, e.g. topic category, keywords, location, demographics, revenue, etc.

4.1 Keyword Extraction

After the main text-body is extracted we perform keyword extraction using various Natural Language Processing techniques. The keyword extraction component first performs Named Entity Recognition where names of people, places and organizations (commercial, governmental or non-governmental organizations) are extracted. To perform Named Entity Recognition and recognize other proper nouns we have leveraged GATE software [9]. Additionally, we have developed several heuristics to improve the keyword extraction process. We count frequencies of all recognized entities. Also since larger documents have larger frequencies, we normalize the frequencies appropriately. Using features such as, normalized frequencies, we identify relative importance of the entities. Moreover, we also use other NLP techniques such as, PLSA (Probabilistic Latent Semantic Analysis) to identify key topics and associated key-words.

Given a large corpus, however, the number of keywords generated becomes too large and unwieldy to apply in the classifier, so we also include a keyword or feature selection stage to determine the final set of vocabulary for the classifier. We are currently evaluating both generative and discriminative text classification techniques, such as the Bayesian and Support Vector Machine (SVM).

Once a document or web page have been successfully classified into one of the ad categories, it is not enough simply rely on its semantic class. The final ad or coupon selection from the database also takes into account other variables such as geography and demographic meta-data if available. For example, if the article is about London, obviously any ad or coupon related to hotel information should correspond to that particular location.

Contextual ad insertion can be quite complex, especially if it is optimized for content semantics along with end-user specific info. This is on-going investigation, and we will provide more technical details when the final solution is field tested.

4.2 Data Set Collection

Since news content is generally the most popular reading materials on the Web, we use web news articles as an example for advertisement insertion. In this experiment, we collect news articles from the USA Today website (http://www.usatoday.com/), which is one of the most visited news site. For each news article, its category can be determined by the URL pattern. For example, the category of the article "Fishermen, hotels feel oil spill's effect" (http://www.usatoday.com/money/economy/2010-05-13-gulfecon13_CV_N.htm) is "money-economy". Currently, we collected roughly 36000 news articles by crawling different topic categories (and sub-categories) of the website to obtain the training data. These articles comprise of 36 categories, with each category containing 1000 articles. For each crawled article, its title and main content are extracted by the method mentioned in section 3.

The dataset is preprocessed by tokenization, stop word removal, and word stemming. Each document is then represented as a token vector, where each element is the TF-IDF (Term Frequency – Inverse Document Frequency) of the token. Those token vectors are further processed with a feature selection algorithm to reduce the dimension, resulting in a vocabulary of 8160 tokens. Here we use the information gain feature selection method [4].

4.3 The Classifier

We use SVM (Support Vector Machine [5]) as the classification method in the experiment because of its proven performance for text classification tasks [6]. Basically, SVM is a classifier for binary classification tasks, but it can be extended to address the multi-class classification tasks by combining the results of multiple binary classifiers [7]. There are two kinds of commonly used policies for the classifier: one-vs-one and one-vs-rest. The former policy builds a binary classifier for any two classes, a positive class and a negative class. The predicted label of an unlabeled document will be determined by the voting result of all of these binary classifiers. The latter policy builds a binary classifier for each class, taking the class as the positive and the remaining classes as negative. Given an unlabeled document, each classifier will output a real value indicating the probability of the document being in the class, and the final label of the document will be determined by the outputs of these classifiers. Considering the number of binary classifiers to be trained, and the size of dataset, we adopt the LIBLINEAR library [8] for its effectiveness and efficiency in dealing with large-scale text data. And we use a 10-fold cross-validation to evaluate the performance of the multi-class news article classification task, where the standard *precision*, *recall*, and *F-measure* are used as metrics.

4.4 Experimental Results

Figure 5 shows the experimental results of the news article classifier for ad matching. From the figure we can see that articles from half of the categories can be classified very well, with an F-measure of over 90%. For the remaining categories, 9 of them have an F-measure between 80%~90%, 8 of them have an F-measure between 70%~80%, and the worst case has an F-measure of 59.8%. Overall, the F-measure is 87.6%, which is considered quite effective.

5. Content Layout

After the article has been extracted and the appropriate ads or coupons are selected, all these content parts need to be assembled to create the final document. Our advertisement selection algorithm is also dependent on layout constraints and available space on the print media. The advertisements or coupons would be placed while considering the layout of the main content and to minimize impact on the aesthetics of the main content layout and

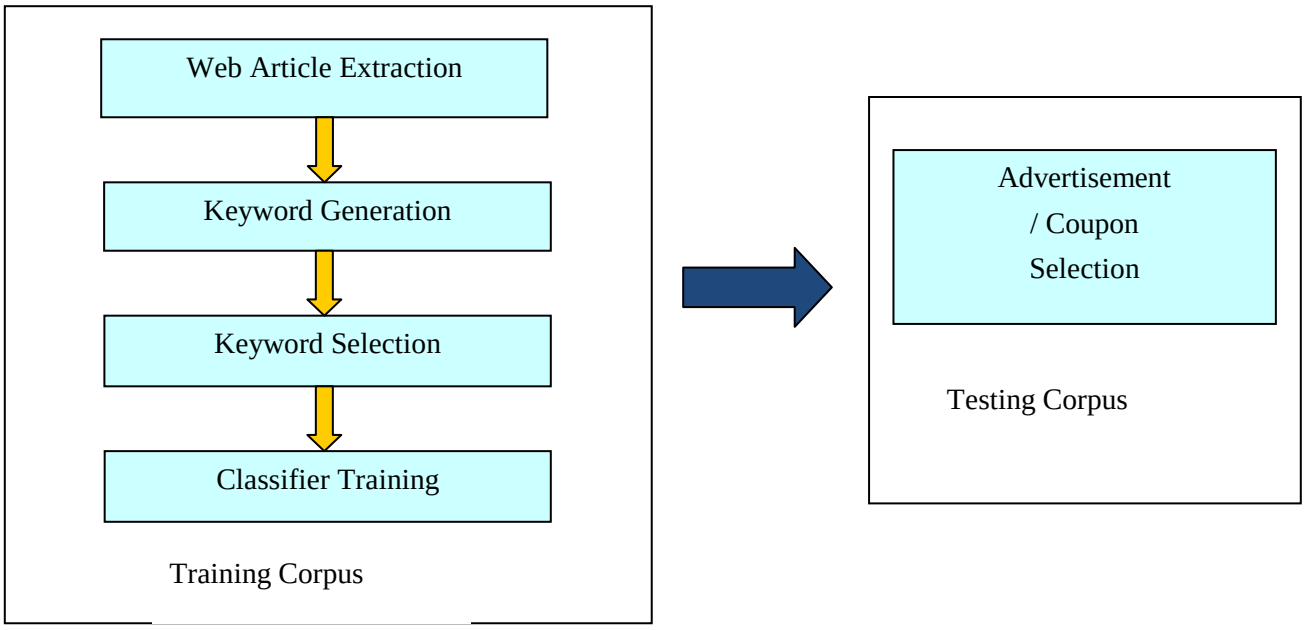

Figure 4. Experimental results on the multi-class news classification

have much superior print quality than those from the Web browser print engine, which are recognized to be quite poor. The ad and coupon insertion needs to be contextual for revenue optimization, using the semantics of the web content and meta-data such as geography and demographic. Since web articles are good candidates for printing, this paper limits the discussion to article pages although the concept can be extended to other web page types as well. The end-to-end printing pipeline requires a client plugin and backend ad server to include the following key components: (1) automatic Web article extraction, (2) article classification for ad selection, and (3) final document layout and creation. This paper presents the result of using SVM classifiers for news article categorization for ad matching. We achieve very good performance using such a classifier, with an average F-measure of over 87%. For future work, we intend to incorporate more factors, e.g. location, demographics, revenue, etc, in the matching model to further improve the relevancy between the articles and the selected ads.

7. REFERENCES

[1] Luo, P., Fan, J., Liu, S., Xiong, Y.H., and Liu, J. 2009. Web article extraction for web printing: a DOM+visual based approach. In *Proceedings of the 9th DocEng*, 2009. 66-69.

[2] Broder, A., Fontoura, M., Josifovski, V., and Riedel, L. 2007. A semantic approach to contextual advertising. In *Proceedings of the 30th SIGIR*, 2007. 559-566.

[3] Yih, W.T., Goodman, J., and Carvalho, V.R. 2006. Finding advertising keywords on web pages. In *Proceedings of the 15th WWW*, 2006. 213-222.

[4] Yang, Y.M. and Pedersen, J. O. 1997. A comparative study on feature selection in text categorization, In *Proceedings of the 14th ICML*, July 08-12, 1997. 412-420.

[5] Burges, C.J.C. 1998. A tutorial on support vector machines for pattern recognition. Data Mining and Knowledge Discovery 2, 2(June 1998), 121–167.

[6] Joachims, T. 1998. Text categorization with support vector machines: learning with many relevant features. In *Proceedings of the 10th ECML*, 1998. 137-142.

[7] Allwein, E.L., Schapire, R.E., and Singer, Y. 2001. Reducing multiclass to binary: a unifying approach for margin classifiers. Journal of Machine Learning Research 1, (September 2001), 113-141.

[8] Fan, R.E., Chang, K.W., Hsieh, C.J., Wang, X, R., and Lin, C.J. LIBLINEAR: a library for large linear classification. Journal of Machine Learning Research 9, (June 2008), 1871-1874.

[9] H. Cunningham, D. Maynard, K. Bontcheva, and V. Tablan. GATE: A framework and graphical development environment for robust NLP tools and applications. In Proceedings of the 40th Anniversary Meeting of the Association for Computational Linguistics, 2002.

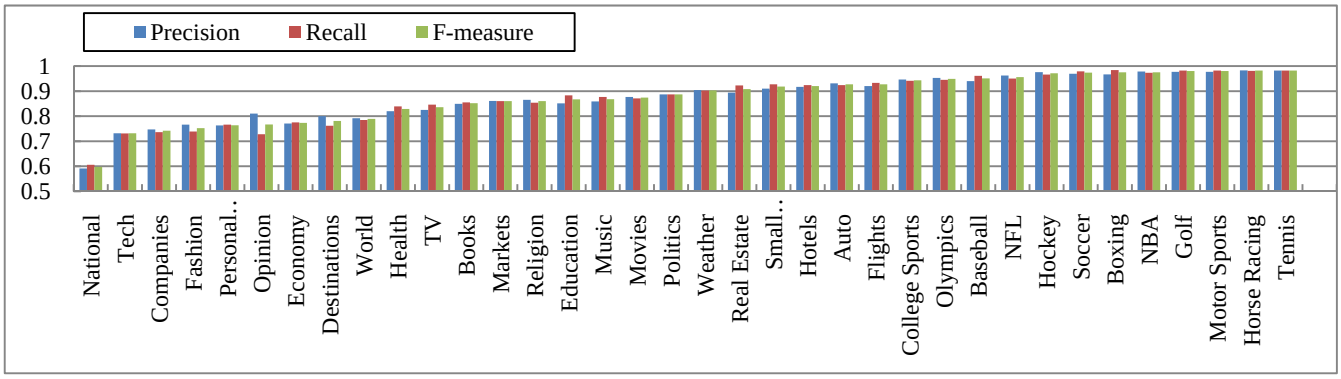

Figure 5. Experimental results on the multi-class news classification