Relevance Estimation with Multiple Information Sources on Search Engine Result Pages

Junqi Zhang†, Yiqun Liu†, Shaoping Ma†, Qi Tian‡
†Department of Computer Science and Technology, Institute for Artificial Intelligence, Beijing National Research Center for Information Science and Technology, Tsinghua University, Beijing 100084, China
‡Department of Computer Science, University of Texas at San Antonio, San Antonio 78249, USA
zhangjq17@mails.tsinghua.edu.cn, {yiqunliu, msp}@tsinghua.edu.cn, qi.tian@utsa.edu

ABSTRACT

Relevance estimation is among the most important tasks in the ranking of search results. Current relevance estimation methodologies mainly concentrate on text matching between the query and Web documents, link analysis and user behavior models. However, users judge the relevance of search results directly from Search Engine Result Pages (SERPs), which provide valuable signals for reranking. Morden search engines aggregate heterogeneous information items (such as images, news, and hyperlinks) to a single ranking list on SERPs. The aggregated search results have different visual patterns, textual semantics and presentation structures, and a better strategy should rely on all these information sources to improve ranking performance. In this paper, we propose a novel framework named Joint Relevance Estimation model (JRE), which learns the visual patterns from screenshots of search results, explores the presentation structures from HTML source codes and also adopts the semantic information of textual contents. To evaluate the performance of the proposed model, we construct a large scale practical Search Result Relevance (SRR) dataset which consists of multiple information sources and 4-grade relevance scores of over 60,000 search results. Experimental results show that the proposed JRE model achieves better performance than state-of-the-art ranking solutions as well as the original ranking of commercial search engines.

KEYWORDS

Multimodal; Information Retrieval; Ranking; Relevance

ACM Reference Format:
Junqi Zhang†, Yiqun Liu†, Shaoping Ma†, Qi Tian‡. 2018. Relevance Estimation with Multiple Information Sources on Search Engine Result Pages. In *The 27th ACM International Conference on Information and Knowledge Management (CIKM '18), October 22–26, 2018, Torino, Italy.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3269206.3271673

1 INTRODUCTION

With the explosive growth of the Web, search engines play an ever more crucial role in information retrieval in our daily lives. The core problem of search engines is to meet users' information needs by locating relevant Web pages. Relevance estimation is usually adopted in the ranking of search results because most search engines follow the Probability Ranking Principle (PRP) [28]. According to PRP, search results are ranked according to their relevance scores. Hundreds or even thousands of features are used in commercial search engines to indicate the relevance of Web pages, such as text matching features, Web graph features, user interaction features and so on. Cascade ranking methods, such as learning to rank algorithms (LTR), are then adopted to combine these features to obtain the final ranking list displayed on SERPs.

To make SERPs provide a more intuitive, direct access to useful information, besides organic results (one blue hyperlink with short snippet contents), most modern search engines also present heterogeneous search results which provide much richer information on SERPs. As shown in the lower left part of Figure 1, news verticals aggregate a couple of news results, of which one is shown in details and illustrated with an image while others only contain the title information. Queries searching for famous people or places always get image verticals, which consist of several images directly showing the person or the place. From these examples, we can see that users can directly judge the relevance from the visual pattern, title, snippet and presentation structure of a search result. Thus, it is essential to incorporate these information sources into the ranking process.

However, there have been almost no researches in how to adopt the visual and structure information of search results in the ranking process to better estimate the relevance. In this paper, we estimate the relevance of search results by jointly considering visual, textual and structure information displayed on SERPs. Inspired by the recent progresses in computer vision and nature language processing tasks, we propose a novel framework named Joint Relevance Estimation model (JRE), which is composed of four subnets: Visual Pattern Learning Network (VPN), Title Semantics Learning Network (TSN), Snippet Semantics Learning Network (SSN), and HTML Tree Structure Learning Network(HSN). VPN learns the visual patterns from the screenshots of search results based on the Convolutional Neural Network (CNN). TSN and SSN explore semantic information from titles and snippets respectively using Long Short Term Memory network (LSTM) [17]. HSN exploits the presentation structures from HTML source codes using the $2D$-Convolution operation.

We also introduce a jointly learned hierarchy of inter-modality and intra-modality attention mechanism. Because different information sources contribute differently to relevance estimation, an inter-modality attention mechanism promotes the importance of

CIKM '18, October 22–26, 2018, Torino, Italy

ACM ISBN 978-1-4503-6014-2/18/10…$15.00
https://doi.org/10.1145/3269206.3271673

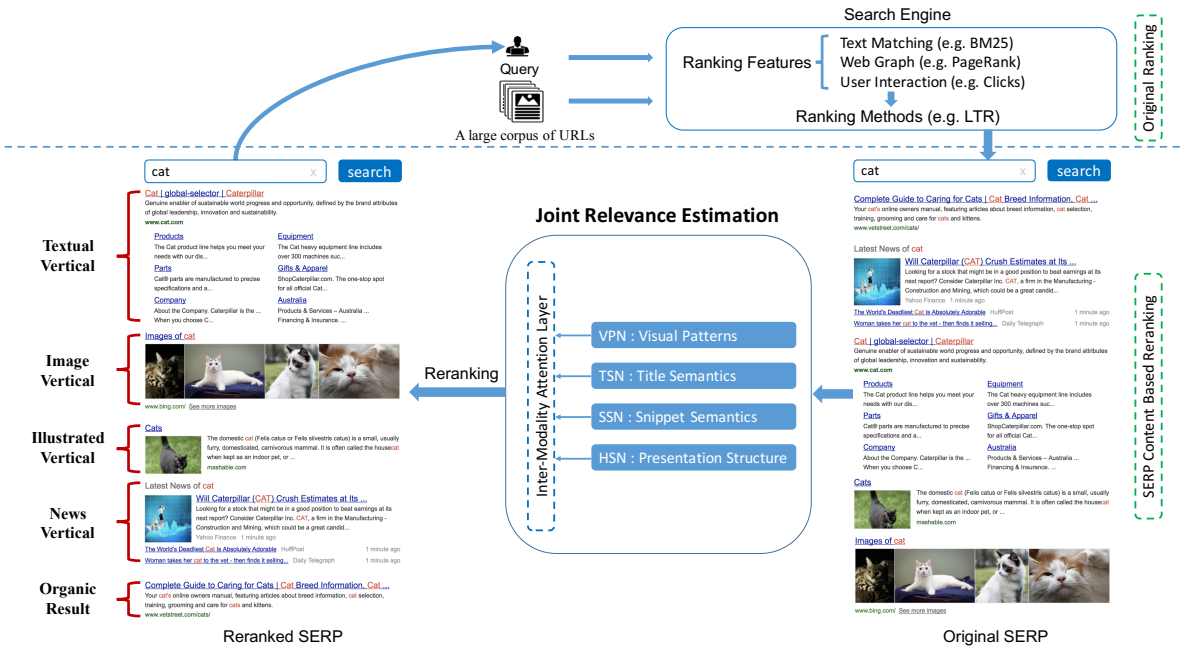

Figure 1: The original ranking process of search engines and the reranking process of JRE.

the most effective modalities. For visual and textual information sources, two kinds of intra-modality attention mechanisms are designed. To better learn the visual patterns, we introduce a result-type guided attention mechanism to VPN. All search results are grouped into 19 categories according to their presentation styles. The feature maps extracted from different types of search results are refined by corresponding attention maps. For textual contents, query terms usually appear several times in titles and snippets. Words located close to query terms may attract more user attention because query terms are usually presented with some salient patterns (e.g. in bold font). Thus, a query guided attention sliding window is introduced to assign different weights to words according to their distances from query terms.

We train and evaluate JRE on a newly constructed Search Result Relevance dataset SRR [1] (details will be introduced in Section 3). With this dataset, we investigate whether visual patterns, textual semantics, and presentation structures help in the estimation of search result relevance. Specifically, the contributions of this paper are three-folds:

- We propose the JRE framework, which incorporates visual patterns, textual semantics and presentation structures of search results into the relevance estimation process. The estimated relevance scores are then used to rerank top-ranked search results to better meet users' information needs.
- A jointly learned hierarchy of inter-modality and intra-modality attention mechanism is designed. The inter-modality attention mechanism applies different weights to different information sources. The intra-modality attention mechanism consists of a novel result-type guided attention mechanism and a query guided attention sliding window, which help better exploit visual and textual information respectively.
- We construct a benchmark Search Result Relevance dataset SRR with 6,338 queries and corresponding top 10 search results for the relevance estimation task. The dataset is the first-of-its-kind and public available. It contains the screenshot, title, snippet, HTML source code and a 4-grade relevance score of each search result.

[1] http://www.thuir.cn/data-srr/

2 RELATED WORK

2.1 Search Result Ranking

The ranking process of commercial search engines can be divided into two steps as shown in the upper right part of Figure 1. First, they design a large number of ranking features as indicators for relevance. A lot of works have been proposed to extract different types of features given the query and a large corpus of URLs [45]. Direct text matching methods compute matching scores of queries and Web documents according to their term frequencies, such as BM25 [34] and vector space model [28]. Link analysis (e.g. PageRank [30]) employs link relations as the proxy of Web page importance based on the Web graph. Click models such as UBM [10], DCM [15], DBN [3] and PSCM [40], exploit user behavior based on experimental hypotheses. Besides, document statistics (e.g. the number of words in various fields), document classifier (e.g. navigational destination vs informational), query features (e.g. click-through rate of the query), topical matching (e.g. topic level similarity), timeliness features (e.g. freshness of a Web page) and spatial features (e.g. location information) are also widely used [45].

Second, cascade ranking methods take the extracted features as input to get the final ranking list. Learning to rank algorithms (LTR) are popular ranking functions, which can be divided into three categories: pointwise (GBRT [14]), pairwise (RankSVM [19], RankBoost [13]) and listwise (AdaRank [43], LambdaMART [42]). Gradient Boosting Regression Tree (GBRT) formalizes LTR as a

regression problem. RankSVM formalizes LTR as a binary classification problem on document pairs and solves it using Support Vector Machines, while RankBoost solves it by means of boosting. AdaRank directly optimizes the performance measures (e.g. NDCG) within the framework of boosting. LambdaMART is a state-of-the-art learning to rank algorithm which is based on boosted regression trees. Finally, search results are ranked according to their relevance scores and displayed on SERPs.

2.2 Visual Pattern and Representation

Visual patterns have strong influence on users while viewing SERPs, Web pages and advertisements. Heterogeneous search results attract users' attention in a completely different way from organic results. Users may be attracted by vertical results and the browsing process on the SERP will be affected [4, 41]. Visual patterns of advertisements also matter for propensity of user response and affect the click through rate (CTR) [1]. Some works have also paid attention to visual appearance of Web pages [12]. They show that the structured layout conveys useful visual information which indicates the relevance of a Web page.

43 statistical visual features were designed in [1] to represent the visual patterns, such as the contrast of gray level image and Hues, lightness, and standard deviation of the image. However, statistical features are not suitable to capture the high level patterns of images. The Convolutional Neural Network (CNN) is a better choice to explore the semantic information of images through multiple convolutional layers. CNN has made great success in many computer vision tasks like image classification [7], object detection [33], and image captioning [44]. The fast development is drived by both advanced structures like AlexNet [22], VGGNet [36], GoogLeNet [38], ResNet [16], and large public image repositories, such as Pascal VOC [11], ImageNet [9], MSCOCO [23] and VisualGenome [21]. In this work, we adopt CNN to capture the complex visual patterns of search results. However, there are great differences between search result screenshots and images in the aforementioned datasets in visual patterns, contents and shapes. Thus, we construct a new dataset consisting of search result screenshots and pre-train CNN on it to adapt the network for the distinct characteristics of screenshots.

2.3 Text-based Result Ranking

Text matching is of central importance to many information retrieval tasks. There are two kinds of text matching methods: traditional term-based approaches and neural networks. Traditional term-based approaches view queries and documents as a set of terms (words or phrases). Each term is weighted based on the statistics of occurrence in the document. Tf-idf and BM25 [28] are widely used weighting schemes, which have achieved great success across a range of collections and search tasks.

However, statistics of term frequency convey little semantic information of nature language. Deep neural networks have shown potential in dealing with this problem. CNN and RNN (Recurrent Neural Network) have been widely used in nature language processing tasks, which perform well in exploring high level structure and semantic information. A lot of works have been proposed to estimate the relevance between a query and a document according to their neural representations. ARCI [18] and MatchPyramid [31] use CNN to explore the sequential and hierarchical structure of natural language sentences. MV-LSTM [39] captures the contextualized local information by a Bidirectional Long Short Term Memory network (Bi-LSTM). Different from these works, we use different structures for the two matching sentences. For titles or snippets, the semantic information is explored by a single LSTM incorporated with a query guided attention mechanism. For the base sentence to be matched with (the search task representation), we set word-level weights to the output of another LSTM.

2.4 Structure Information on SERPs

Previous works show that the structure and layout of Web pages have a strong impact on the quality that users perceive [27]. Structure features derived from HTML source codes of Web pages, such as the numbers of lists and DOM elements, are utilized to indicate the Web page quality. A Web page can be partitioned into multiple segments or blocks. The importance of blocks in a Web page is often not equivalent [37].

Though the block of a search result is much smaller than a Web page, it also has its own structure and layout, which influence the user's judgement of relevance. The structure of some search results is simple, such as organic results which are made up of two main parts: the title block and the snippet block. Others may have complex structures, such as map verticals which have title block, snippet block, zoomed map block and input block. The structure of search results can be derived from HTML source codes of SERPs. For most search engines, the 'div' HTML tag is a high level element containing other blocks. The 'table' tag may contain multiple rows and columns of data. The 'p' tag represents a text block, while 'img' denotes an image block. These tags are constructed as a HTML tree. Thus, we can utilize the tree structure of HTML codes to capture structure information of search results.

2.5 Attention Mechanism

Attention mechanism has shown effectiveness in various computer vision and natural language processing tasks. In image captioning [24, 44], the language generator attends to different regions in images while predicting captions, so that the words in captions are related to objects in images. In machine translation, the attention mechanism tells the decoder what is now translated [2, 26], and the words correspond to each other in different languages.

Besides the above intra-modality attention, there have also been some works adopting inter-modality attention. An attention mechanism applying different weights to textual and visual modalities in the multimodal image search task is proposed in [5].

Inspired by these approaches, we introduce two types of attention mechanisms in this work: the inter-modality and intra-modality attention mechanisms.

3 THE SRR DATASET

We construct a new dataset called Search Result Relevance (SRR) to train and evaluate the proposed models. Search log data on September 3rd, 2017 collected from a popular commercial search engine are adopted, which contain $13,836,079$ different queries. $37,936$ distinct queries with frequency between 100 to $1,000$ are retained, which are usually regarded as torso queries with "median" frequencies and usually the most important concerns for ranking algorithm design [46]. Google Chrome Driver is then used to grab

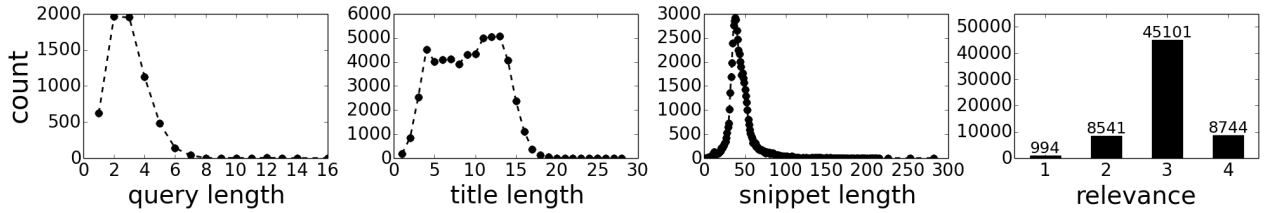

Figure 2: Statistics of SRR dataset.

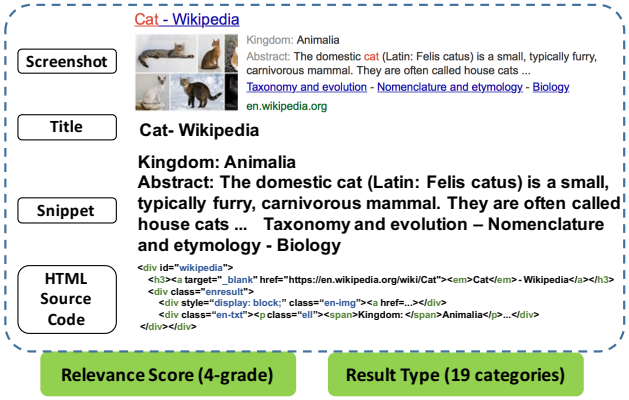

Figure 3: An example of SRR dataset.

data. For each query, we crawl the screenshot and HTML document of the corresponding SERP. The screenshot of each search result is then cropped, and the title, snippet, HTML source code are extracted from the SERP page. The obtained search result content data of the 37,936 queries constitute SRR-raw dataset.

After collecting SRR-raw, we further assess the search results in terms of *Relevance* and *Result-Type* through crowd-sourcing to construct SRR. 6,338 queries are sampled from the original 37,936 queries in SRR-raw. The top 10 search results of each query are retained, resulting in 63,380 different search results.

For *Relevance*, we use the typical 4-grade relevance criteria: irrelevant (R=1), marginally relevant (R=2), relevant (R=3) and highly relevant (R=4) [25]. To make sure the data annotations are reliable, we ensure that each search result is annotated by three assessors. Assessors are asked to give a relevance score based on the search result screenshot and the corresponding query. The most voted relevance score of three assessors is adopted as the final label. If three scores are all different (only 1171 of the 63380 search results), we choose the one closest to the average score. The Cohen's Weighted κ [8] is 0.5887, which indicates that the annotated relevance scores are of reasonable quality.

For *Result-Type*, based on "result types" provided by the search engine, we manually divide the search results into 19 categories according to the presentation styles. Since the result type is easy to distinguish, each search result is annotated by one assessor. The descriptions of result types are shown in Table 1.

An example of SRR dataset is shown in Figure 3. Figure 2 gives the length distributions of queries, titles and snippets, as well as the distribution of relevance labels. The average size of search result screenshots is 549×128 pixels. Due to the limited space, we will give more details along with the dataset.

Table 1: The descriptions of 19 result types.

Organic Result : One blue hyperlink with short snippet contents.
Illustrated Vertical : Consisting of the title, snippet and an illustration on the left of the search result.
Encyclopedia Vertical : Search results from encyclopedia Web sites, usually have similar layout with Illustrated Verticals.
Image Vertical : Composed of one row of images.
Video Vertical : Composed of one row of video snapshots.
Multi-row Image Vertical : Composed of multiple rows of images.
Multi-row Video Vertical : Composed of multiple rows of video snapshots.
Tutorial Vertical : Providing instructions to some questions, usually containing diagrams with multiple steps.
Forum Vertical : Search results from forum websites, usually having an image on the left and a list of hyperlinks on the right.
Map Vertical : Consisting of a zoomed map and an input box.
News Vertical : Aggregation of multiple news results, of which one is shown in details and usually illustrated with an image while others only have title information.
Question Answering Vertical : Aggregation of multiple answers from a Community Question-Answering site, of which one is shown in details while others only have title information.
Textual Vertical : Hyperlinks of different channels from a Web site and corresponding snippets.
Download Vertical : Direct download links of certain softwares described by the query.
Direct Answer Vertical : Directly showing the required information described by the query
Application Vertical : Embedded applications which can be directly interacted on SERPs, such as music or express inquiry services.
Navigation Vertical : Giving a catalog of TV serials, books and so on.
Shopping Vertical : Shopping search results from E-commerce Web sites.
Others : Search results belonging to none of the above categories.

4 JOINT RELEVANCE ESTIMATION MODEL

The goal of Joint Relevance Estimation (JRE) model is to explore the visual patterns, textual semantics and presentation structures jointly from screenshots, titles, snippets and HTML source codes of search results. The multiple information sources of a search result are defined as $m = \{v, t, s, h\}$, where v, t, s and h refer to screenshot, title, snippet and HTML source code respectively. Given m, VPN, TSN, SSN, and HSN can predict a relevance score respectively. The final relevance score of JRE is jointly learned from the predictions of the four subnets.

Inter-Modality Attention Mechanism. Different information sources contribute differently to the relevance estimation. Thus, we build an inter-modality attention layer on top of the four subnets. Each information source is assigned with an attention weight, which can

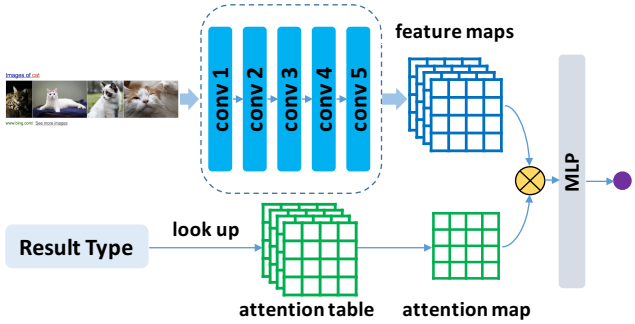

Figure 4: The structure of Visual Pattern Learning Network.

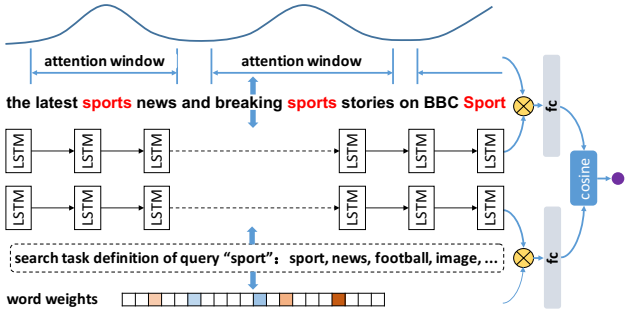

Figure 5: The structure of Title Semantics Learning Network and Snippet Semantics Learning Network.

be automatically learned in the training process. The final predicted relevance of JRE is defined as:

$$\mathcal{R}_{JRE} = \sum_{I \in \{VPN,\ TSN,\ SSN,\ HSN\}} w_I \times \mathcal{R}_I \tag{1}$$

where $\mathcal{R}_I$ is the relevance score predicted by each subnet, and w_I is the corresponding attention weight.

Optimization. JRE is initialized by the pre-trained four subnets and fine-tuned end-to-end. All the subnets and JRE utilize *CrossEntropy* as the loss function, which is defined as:

$$\mathcal{L}(r, t; \theta) = \frac{1}{N}\sum_{j=1}^{N}(-t_j \log r_j - (1 - t_j)\log(1 - r_j)) + \lambda||\theta||_2^2 \tag{2}$$

where r_j and t_j are predicted and target relevance scores of the *j-th* search result in the N training samples respectively, θ includes all the parameters in the neural network, λ denotes the *L2* regularizer coefficient.

4.1 Visual Pattern Learning Network

Visual Pattern Learning Network (VPN) learns visual patterns of search result screenshots to predict relevance. As discussed in Section 2.2, visual patterns have strong influence on users while viewing SERPs. The user's judgement can be largely affected by "how does the search result look like". Thus, it is important to estimate the relevance from the visual aspect. The structure of VPN is shown in Figure 4.

4.1.1 Visual Representation. We employ CNN to generate visual features from each search result screenshot. The CNN architecture is based on AlexNet for simplicity. The convolutional layer is defined as:

$$conv_i(z; \mathcal{K}_i) = MaxPooling(\ ReLU(\ \Phi(\ z\ )\ )\ ) \tag{3}$$

where $\Phi(\cdot)$ denotes the convolutional operation, z is the input features, $\mathcal{K}_i$ is the kernels to be learned in the *i-th* convolutional layer.

VPN takes a search result screenshot v as input, and projects it into feature maps through five convolutional layers. The projected feature maps are denoted as $\hat{v} \in \mathbb{R}^{h \times w \times c}$, where h, w, c are the height, width and channel number respectively.

$$z_0 = v \tag{4}$$

$$z_l = conv_l(z_{l-1}; \mathcal{K}_l),\ l = 1, 2, 3, 4, 5 \tag{5}$$

$$\hat{v} = z_5 \tag{6}$$

4.1.2 Result-Type Guided Attention Mechanism. Different types of search results have different visual patterns. For example, image verticals are composed of images while textual verticals contain structured textual links. The informative parts of different verticals vary a lot. Thus, VPN generates a result-type guided attention table $AT \in \mathbb{R}^{h \times w \times t}$ to better distinguish them and concentrate on the most informative parts of the screenshots, where t is the number of result types. AT consists of t attention map $AM \in \mathbb{R}^{h \times w}$. The visual feature maps $\hat{v}$ are refined through re-weighting each location by the corresponding weight in AM via element-wise production. The result-type guided visual feature maps v_{attn} is then projected to a *relevance* score through a multilayer perceptron (MLP):

$$v_{attn} = \hat{v} \otimes AT\{i\} \tag{7}$$

$$\mathcal{R}_{VPN} = \delta(\psi(v_{attn})) \tag{8}$$

where $\delta(\cdot)$ is the sigmoid function, $\psi(\cdot)$ is implemented as a multilayer perceptron, i refers to the result type id, $\otimes$ represents element-wise production, and $AM = AT\{i\}$.

4.2 Textual Semantics Learning Network

Title Semantics Learning Network (TSN) and Snippet Semantics Learning Network (SSN) explore semantic information of titles and snippets respectively. Apart from the visual aspect, the textual contents provide abstracts of search results which are also vital bases for users' judgements. TSN and SSN are trained separately because we believe that the title and snippet are organized with different language patterns. They share the same structure as shown in Figure 5.

4.2.1 Textual Representation. When predicting the semantic matching score as relevance, the search task ST need to be defined. Queries are usually regarded as search tasks. However, information contained in queries is limited. So besides the query, we also design a more comprehensive semantic representation of the search task based on pseudo relevance feedback. For each query q, the most important k keywords from the query as well as titles and snippets of the top 10 search results are filtered out:

$$text = \{q, t_i, s_i | i = 1, \cdots, 10\} \tag{9}$$

$$T = tfidf(text) \in \mathbb{R}^{21 \times N} \tag{10}$$

$$W = Normalization(AvgPool(T)) \in \mathbb{R}^{N} \tag{11}$$

where t_i, s_i refer to the title and snippet of the i-th search result respectively, N is the number of distinct words in $text$, $tfidf$ computes the tf-idf score of each word, $AvgPool$ is the the average pooling operation, and $Normalization$ controls the tf-idf weights between 0 and 1. Then k keywords with the maximum weights in W are set as the search task representation ST^k, and the corresponding weights are W^k. The query is denoted as ST^q, while the corresponding weights are W^q with all elements set to 1.

The title or snippet and the corresponding ST are then fed into LSTM. The output of LSTM is denoted as $h \in \mathbb{R}^{n \times d}$, where n is the sentence length, d is the number of hidden units in LSTM. The textual representations are defined as:

$$h^t = LSTM(t),\ h^s = LSTM(s),\ h^{st} = LSTM(ST) \tag{12}$$

Algorithm 1: Query Guided Attention Sliding Window

Input: $AW = [a_1, a_2, \ldots, a_l]$, $Q = [q_1, q_2, \ldots, q_m]$ and a sentence $S = [s_1, s_2, \ldots, s_n]$, l is the window size which is odd, m is the query length, n is the sentence length.
Output: $W = [w_1, w_2, \ldots, w_n]$ as the weights of terms in S.
$W = [1, 1, \ldots, 1] \in \mathbb{R}^n$;
for *each term* s_{index} *in* S **do**
  if $s_{index} \in Q$ **then**
    for $i \leftarrow 1$ *to* l **do**
      if $index - \frac{l+1}{2} + i \geq 1$ *and* $index - \frac{l+1}{2} + i \leq n$ **then**
        $w_{index-\frac{l+1}{2}+i} = w_{index-\frac{l+1}{2}+i} \times a_i$
      end
    end
  end
end

4.2.2 Query Guided Attention Sliding Window. As shown in Figure 5, we design a query guided attention sliding window AW to assign different weights to words according to their distances from query terms, where $AW \in \mathbb{R}^l$, l is the window size. AW slides along a title or snippet to detect the query terms and applies weights to words inside it. The closer a word is to query terms, the more weight it gets. After the attention window sliding process, the sentence weight W is obtained. The overall attention sliding window algorithm is summarized in Algorithm 1.

The relevance score of TSN (SSN) is the cosine similarity between the search task representation and the title (snippet).

$$h^{t/s}_{attn} = \psi(AvgPool(h^{t/s} \times W^{t/s})) \tag{13}$$

$$h^{st}_{attn} = \psi(AvgPool(h^{st} \times W^{st})) \tag{14}$$

$$\mathcal{R}_{TSN\,/\,SSN} = cosine(h^{t/s}_{attn}, h^{st}_{attn}) = \frac{h^{t/s}_{attn} \odot h^{st}_{attn}}{||h^{t/s}_{attn}|| \times ||h^{st}_{attn}||} \tag{15}$$

where $W^{t/s}$ is the output weights of Algorithm 1, W^{st} is either W^q or W^k, $\psi(\cdot)$ represents a full-connected layer, and $\odot$ refers to dot production.

4.3 HTML Tree Structure Learning Network

In SERPs, the search result is constructed with particular HTML tree structure according to its result type. Each visible item has a corresponding HTML tag, such as "div", "h3", "p", "a", while each HTML tag has a class attribute indicating the content of the element. For example, "box" is a high level element which may contain both texts and images, while "txt" refers to texts and "img" represents the image. The presentation structure can be exploited directly from the HTML source code of each search result.

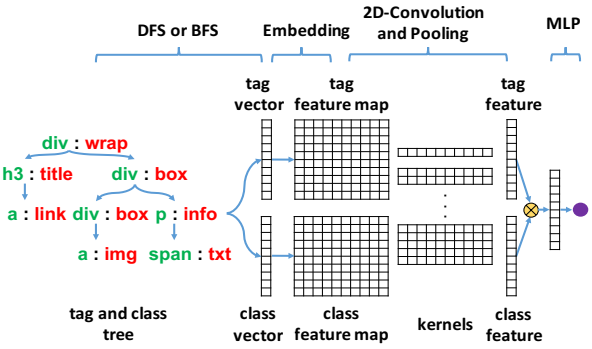

Figure 6: The structure of HTML Tree Structure Learning Network. Words in green are HTML tags while red refers to the HTML class.

A tag tree and a class tree are constructed from the HTML source code at the first. The tree structure cannot be used for learning directly. So we convert the two trees to vectors through Depth-First-Search (DFS) or Breadth-First-Search (BFS) algorithm. Each tag or class is embedded into a vectorial representation. Then after 2D-convolution and pooling using different kernels, the tag feature f_{tag} and class feature f_{class} are obtained. The whole HTML tree structure representation f_{html} is element-wise production of f_{tag} and f_{class}. The HTML feature f_{html} is then projected to a *relevance* score through a multilayer perceptron (MLP).

$$f_{html} = f_{tag} \otimes f_{class} \tag{16}$$

$$\mathcal{R}_{HSN} = \delta(\psi(f_{html})) \tag{17}$$

where $\delta(\cdot)$ is the sigmoid function, $\psi(\cdot)$ is implemented as a multi-layer perceptron.

5 MODEL TRAINING

5.1 Initialization

To adapt VPN for the visual patterns of search result screenshots, we pre-train AlexNet to classify result types on 167,009 sampled screenshots of `SRR-raw`. The sampled data are split into the training, validation and testing sets at a ratio of 8 : 1 : 1. The classification accuracy of the 19 result types is 98.79%, indicating the strong ability of CNN to distinguish visual patterns. The five convolutional layers in VPN are initialized with weights in the pre-trained AlexNet.

For textual modality, we employ Skip-Gram algorithm [29] to train word embeddings on Wikipedia[2] dataset and fine-tune it on titles and snippets in `SRR-raw`.

5.2 Parameter Settings

We split `SRR` into the training, validation and testing sets at a ratio of 8 : 1 : 1. Adaptive Moment Estimation [20] method is used to

[2]https://dumps.wikimedia.org/zhwiki/latest/zhwiki-latest-pages-articles.xml.bz2

Table 2: Performance of different query guided attention windows.

Size	Weight	NDCG@3	NDCG@5	NDCG@10	MSE
3	1.2, 1.3, 1.2	0.8165	0.8549	0.9293	0.0344
	1.8, 2.0, 1.8	**0.8244**	**0.8611**	**0.9323**	**0.0337**
	3, 4, 3	0.7853	0.8268	0.9166	0.0460
5	1.1, 1.2, 1.3, 1.2, 1.1	0.8180	0.8551	0.9299	0.0343
	1.6, 1.8, 2.0, 1.8, 1.6	**0.8197**	**0.8604**	**0.9309**	**0.0336**
	2, 3, 4, 3, 2	0.8126	0.8530	0.9281	**0.0336**
No Attention		0.8021	0.8435	0.9242	0.0351

train all the models. The initial learning rate is set to 0.0001. Search result screenshots are resized to 550×130 pixels according to the average size. Titles and snippets are truncated to 20 and 100 words respectively for training efficiency. The word embedding dimension is set to 200. The hidden representation dimension of LSTM is set to 1000. For HSN, we only consider the 10 major elements in the HTML source code of each search result, because tiny HTML elements have little impacts on presentation structures. Thus, the length of tag and class vector is set to 10. Each HTML tag or class is embedded into a 200-dimension vector. Five different sizes of kernels are utilized in the 2D-convolutional operation, which are $\{kernel \in \mathbb{R}^{h \times 200 \times 256} | h = 1, 2, 3, 4, 5\}$. The 4-grade relevance score $\mathcal{R}$ is normalized to $(\mathcal{R} - 1)/3$, where $\mathcal{R} \in \{1, 2, 3, 4\}$.

5.3 Evaluation Metrics

All evaluation and empirical analysis are reported by mean Normalized Discounted Cumulative Gain (NDCG) [35] and Mean Squared Error (MSE):

$$MSE\ (r,label) = \frac{1}{n}(r\text{-}label)^T(r\text{-}label) \tag{18}$$

$$NDCG@k = N_k^{-1} \sum_{j=1}^{k} \frac{2^{r_j} - 1}{\log_2(1+j)} \tag{19}$$

where r is the estimated relevance scores, *label* is the relevance labels, n is the number of search results, N_k denotes the maximum of $\sum_{j=1}^{k} \frac{2^{r_j}-1}{\log_2(1+j)}$, r_j refers to the relevance label of the search result ranked at the $j\text{-}th$ position.

6 EXPERIMENTAL RESULTS

6.1 Baselines

There have been no multimodal neural models utilizing SERP contents to estimate relevance. Thus, we adapt state-of-the-art LTR and neural text matching methods for SERP contents as two types of baselines.

6.1.1 Learning To Rank. To train LTR methods, 19 different statistical features are extracted from visual modality (screenshots), textual modality (titles and snippets) and HTML modality (HTML source codes). Details about the visual features are described in [1, 6]. The textual features are designed following the methodology used by Microsoft LETOR data [32]. The two HTML features are designed to reflect the structure complexity of search results.

Visual Features: gray level contrast, contrast of dominant hues, number of dominant bins in gray level histogram, number of dominant hues, standard deviation of gray level images, standard deviation of hues, colorfulness.

Table 3: Performance of different search task representations.

Model	NDCG@3	NDCG@5	NDCG@10	MSE
TSN - ST^q	**0.8275**	**0.8635**	**0.9332**	0.0336
TSN - ST^{10}	0.8244	0.8611	0.9323	0.0337
TSN - ST^{20}	0.8260	0.8622	0.9323	**0.0334**
SSN - ST^q	0.8148	0.8477	0.9255	0.0350
SSN - ST^{10}	**0.8191**	**0.8559**	**0.9282**	0.0351
SSN - ST^{20}	0.8154	0.8508	0.9264	**0.0344**

Textual Features: Document Length, tf, idf, tf-idf and BM25 scores of titles and snippets

HTML Features: number of nodes in a HTML tree, number of node types in a HTML tree.

Three types of LTR methods are adopted in our experiments:
Pointwise: GBRT
Pairwise: RankSVM and RankBoost
Listwise: AdaRank and LambdaMART

For RankSVM, we use the implementation of SVM^{rank} [3], while RankLib[4] is used for RankBoost, AdaRank, and LambdaMart.

6.1.2 Neural Text Matching. Three neural text matching models are compared to in our experiments: MV-LSTM [39], ARCI [18] and MatchPyramid [31]. The implementations of the three models in Match-Zoo[5] are adopted. *MSE* is used as the loss function, which preforms better than *CrossEntropy* in our training process. To make a fair comparison with our models, the word embedding dimension is set to 200 and the hidden representation dimension of LSTM in MV-LSTM is set to 1000 (which is originally set to 50). The effects of different search task representations are evaluated to make a comprehensive comparison and we report the best performance.

6.2 Attention Window Selecting for TSN and SSN

Two different sizes and three different weights of query guided attention sliding windows are tested on TSN to apply different levels of attention. ST^{10} (described in 4.2) is utilized as search task representation in this experiment. We also test the performance of TSN without attention. Results in Table 2 show that attention window of size 3 and weight $[1.8, 2, 1.8]$ performs the best. We will adopt this attention window setting in the following experiments. We did not try more window settings and the performance may be further improved by better attention window. A conclusion can also be drawn from the results that inappropriate attention window (e.g. $[3, 4, 3]$) is worse than no attention.

6.3 Search Task Selecting for TSN and SSN

Three kinds of search task representations are defined: ST^q, ST^{10}, ST^{20} as described in 4.2. The query guided attention sliding window is set to $[1.8, 2, 1.8]$. According to the results, we will use ST^q as the search task in TSN and ST^{10} in SSN. It is an interesting finding that "title with ST^q" and "snippet with ST^k" preform better. This may be explained by the fact that titles of top search results contain many exact matching query terms.

[3] http://www.cs.cornell.edu/people/tj/svm_light/svm_rank.html
[4] https://sourceforge.net/p/lemur/wiki/RankLib/
[5] https://github.com/faneshion/MatchZoo

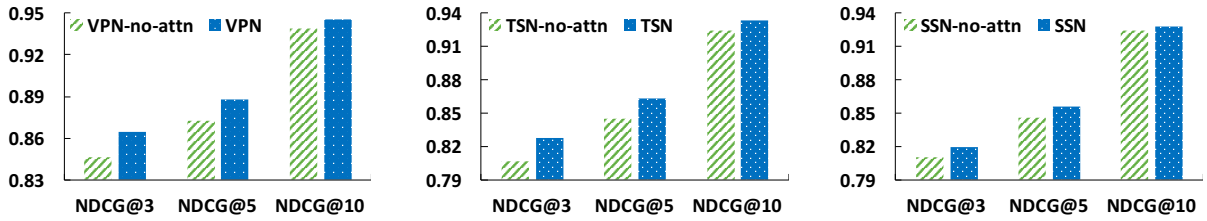

Figure 7: The effectiveness of intra-modality attention mechanism. 'no-attn' refers to models without attention mechanism. The performances of the models with attention mechanism are significant better ($p < 0.05$) than that of the models without it.

Table 4: Experimental results of different models. *, $^+$ and $^\triangle$ denote the significant improvement compared to the original rankings of the search engine, the best LTR baseline LambdaMART and the best neural text matching baseline MatchPyramid-title respectively (p-value$< 10^{-4}$).

Model	NDCG@3	NDCG@5	NDCG@10	MSE
Search Engine	0.8025	0.8409	0.9243	-
GBRT	0.8140	0.8553	0.9267	0.0388
RankSVM	0.8200	0.8584	0.9293	-
RankBoost	0.8181	0.8587	0.9278	-
AdaRank	0.7981	0.8410	0.9207	-
LambdaMART	**0.8306**	**0.8651**	**0.9324**	-
MV-LSTM-snippet	0.8087	0.8434	0.9255	0.0357
MV-LSTM-title	0.8165	0.8507	0.9280	0.0346
ARCI-snippet	0.8110	0.8430	0.9264	0.0338
ARCI-title	0.8196	0.8553	0.9303	0.0340
MatchPyramid-snippet	0.8108	0.8472	0.9262	0.0338
MatchPyramid-title	**0.8331**	**0.8645**	**0.9333**	**0.0326**
HSN	0.8073	0.8438	0.9261	0.0379
SSN	0.8191	0.8559	0.9282	0.0351
TSN	0.8275	0.8635	0.9332	0.0336
VPN	**0.8647**	**0.8878**	**0.9451**	**0.0297**
JRE	**0.8715**$^{*+\triangle}$	**0.8949**$^{*+\triangle}$	**0.9478**$^{*+\triangle}$	**0.0296**

6.4 Comparison

We compare our models with all the baselines and original ranking lists of the search engine. Since there are three kinds of search task representations for each text matching model, the best performance among the three is reported. All the methods focus on reranking of the top 10 search results, thus have high values of NDCGs. From the experimental results, some observations can be made:

Comparison with the original ranking list. All the methods except for AdaRank achieve superior performance than original ranking lists of the search engine in terms of *NDCG@3,5,10*, which indicates that there is still large room for improvements in search engines. Either a single information source or multiple of them help in reranking search results.

Comparison among information sources. VPN preforms the best in the four subnets of JRE, which verifies the importance of visual patterns in predicting relevance, especially in reranking top results. TSN perform better than SSN. Although information contained in snippets is much richer, titles are more refined. HSN achieves almost the same performance with DFS or BFS encoding algorithm, and is just slightly better than the original ranking of the search engine. Though HTML source codes provide valuable information of presentation structures, the information they contain is limited. Converting a HTML element tree to a vector will also lose a lot of structure information. We may try to find some better ways to incorporate the HTML tree-based structure information and we would like to leave this as future work.

Comparison with baselines. LambdaMART performs the best among LTR methods while MatchPyramid performs the best among neural text matching models. TSN is better than MV-LSTM and ARCI on titles. SSN achieves superior performance than all text matching baselines on snippets. This shows the effectiveness of our proposed attention guided text matching subnets. Though LTR methods utilize multiple information sources, the statistical features are not capable to indicate the relevance of search results. This shows the advantage of neural networks in exploring high level semantic information. When different information sources are jointly explored, JRE achieves superior performance than all the other models. The improvements of JRE over original ranking lists of the search engine are 8.60%, 6.42%, and 2.54% in terms of *NDCG@3,5,10* respectively (all improvements are statistically significant according to t-test with p-value $< 10^{-4}$).

7 DISCUSSION

7.1 The Effectiveness of Attention Mechanism

7.1.1 Inter-Modality Attention Mechanism. The automatically learned inter-modality attention can better incorporate different information sources, which jointly achieve the best performance. The ability of each information source in relevance estimation can be reflected by the respective performance of VPN, TSN, SSN and HSN to a certain extent. To give an ablation study of each modality, we prune out each subnet from JRE. As shown in Table 5, when VPN is discarded, there is significant loss in performance, while the whole model suffers little without HSN. TSN and SSN have nearly the same impact on JRE.

Table 6 shows the attention weight of each subnet in all frameworks. In all circumstances, VPN accounts for the biggest weight and HSN accounts for the smallest. TSN and SSN contribute equally to the model.

From the contrastive results, it is clear that visual patterns are the most important signals in relevance estimation, and semantic information of texts takes the second place. Due to the limitation of representation ability of the HTML tree, the presentation structure of the search result has not been effectively excavated, which should be explored in the future.

7.1.2 Intra-Modality Attention Mechanism. As shown in Figure 7, the introduced intra-modality attention mechanisms can improve the performance of VPN, TSN and SSN by a large margin. This

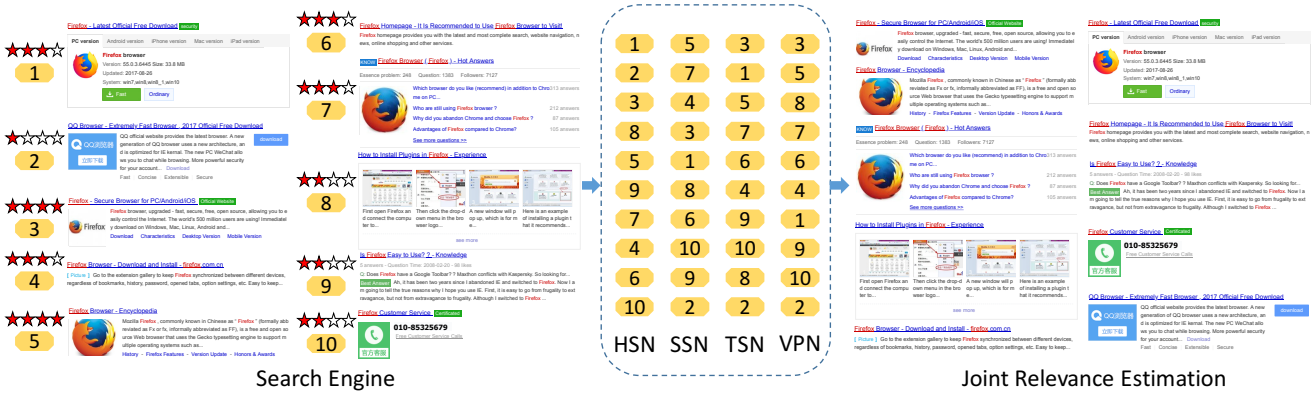

Figure 8: A qualitative example of query "Firefox". The number in the yellow icon is the original ranking position of the search result. The number of red stars represents the relevance label. The yellow icons in the middle show the ranking list of each subnet. All search results are translated into English. Best viewed in electronic form.

Table 5: Performance when discarding any subnet of JRE.

Model	NDCG@3	NDCG@5	NDCG@10	MSE
JRE without VPN	0.8330	0.8663	0.9337	0.0339
JRE without TSN	0.8668	0.8908	0.9473	0.0299
JRE without SSN	0.8668	0.8919	0.9473	0.0303
JRE without HSN	0.8703	0.8945	0.9475	0.0298
JRE	**0.8715**	**0.8949**	**0.9478**	**0.0296**

Table 6: Weights of each subnet in different frameworks.

Model	VPN	TSN	SSN	HSN
JRE without VPN	-	0.3532	0.3547	0.2921
JRE without TSN	0.3581	-	0.3250	0.3169
JRE without SSN	0.3771	0.3082	-	0.3147
JRE without HSN	0.3720	0.3101	0.3179	-
JRE	0.2778	0.2355	0.2438	0.2429

indicates that by paying more attention to the most informative parts of search results, VPN can learn the visual patterns more efficiently. The experimental results also show that words around query terms contain more related information and may attract more attention when users reading the texts. It is consistent with our observation.

7.2 Qualitative Analysis

To intuitively examine the effectiveness of JRE, we show the search results of query "Firefox" in Figure 8. The original ranking list of the search engine is shown in the left. The top 10 search results are partitioned into two columns according to their ranking positions. When users searching for "Firefox", they are more likely to expect the official Web site or some knowledge about Firefox browser, or they have some troubles in using Firefox and they want the solutions. But the search engine puts a download result at the top, which only helps for some people. The search result ranked at the $2th$ position is an advertisement, which is the most unaccessible. The official Web site result, encyclopedia result and search results solving problems are all not ranked to higher positions.

As for JRE, the ranking lists of the four subnets are shown in the middle of Figure 8. HSN puts the $8th$ search result from a lower position to a higher-ranked one, which has a more complicated structure. The $4th$ and $6th$ search results are ranked behind perhaps due to the simple structure. SSN puts the $5th$, $7th$, $4th$, and $3th$ search results at the top positions, whose snippets are more semantically similar to words in the search task definition, such as "Firefox", "browser", "download", "free" and "secure". The $2th$ search result is ranked at the bottom as its snippet is not related with the search need at all. TSN concentrates on titles. The $2th$, $8th$, $10th$, and $9th$ titles are less related with "Firefox", thus ranked behind. VPN puts the $3th$, $5th$, $8th$, and $7th$ search results at the top, which are all reliable.

Finally, JRE incorporates all the information sources and provide an adjusted satisfactory ranking list. The official Web site result, encyclopedia result and the search results solving problems are ranked at the top. Useless search results and the advertisement are ranked behind.

8 CONCLUSIONS AND FUTURE WORK

As heterogeneous verticals account for more and more in search results, exploring their contents becomes vital in relevance estimation. In this paper, we jointly learn the visual patterns, textual semantics and presentation structures from different information sources of search results, including screenshots, titles, snippets and HTML source codes. Meanwhile, inter-modality and intra-modality attention mechanisms are introduced to better utilize information from different sources. The proposed JRE model achieves the best performance among all the approaches, and significantly improves the original ranking of the search engine by 8.60%, 6.42%, and 2.54% in terms of *NDCG@3,5,10* respectively. The contribution of each information source is also investigated. Besides, we have also constructed a new SRR dataset containing different information sources of search results as well as annotated relevance labels. In the future, we would like to consider the query topic in relevance estimation because users' preference of search results is related with search needs. Besides, the structure information in HTML source codes can be better utilized to incorporate different information items such images and texts. A more effective fusion method of different information sources can be proposed.

ACKNOWLEDGMENTS

This work is supported by Natural Science Foundation of China (Grant No. 61622208, 61732008, 61472206) and National Key Basic Research Program (2015CB358700). We also benefit from discussions with Jiafeng Guo, Min-Yen Kan and Maarten de Rijke.

REFERENCES

[1] Javad Azimi, Ruofei Zhang, Zhou Yang, Vidhya Navalpakkam, Jianchang Mao, and Xiaoli Fern. 2012. The impact of visual appearance on user response in online display advertising. 457–458.

[2] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural Machine Translation by Jointly Learning to Align and Translate. *Computer Science* (2014).

[3] Olivier Chapelle and Ya Zhang. 2009. A dynamic bayesian network click model for web search ranking. In *Proceedings of the 18th international conference on World wide web*. ACM, 1–10.

[4] Danqi Chen, Weizhu Chen, Haixun Wang, Zheng Chen, and Qiang Yang. 2012. Beyond ten blue links: enabling user click modeling in federated web search. In *Proceedings of the fifth ACM international conference on Web search and data mining*. ACM, 463–472.

[5] Kan Chen, Trung Bui, Chen Fang, Zhaowen Wang, and Ram Nevatia. 2017. AMC: Attention guided multi-modal correlation learning for image search. *arXiv preprint arXiv:1704.00763* (2017).

[6] Haibin Cheng, Roelof Van Zwol, Javad Azimi, Eren Manavoglu, Ruofei Zhang, Yang Zhou, and Vidhya Navalpakkam. 2012. Multimedia features for click prediction of new ads in display advertising. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 777–785.

[7] Dan C Ciresan, Ueli Meier, Jonathan Masci, Luca Maria Gambardella, and Jürgen Schmidhuber. 2011. Flexible, high performance convolutional neural networks for image classification. In *IJCAI Proceedings-International Joint Conference on Artificial Intelligence*, Vol. 22. Barcelona, Spain, 1237.

[8] Jacob Cohen. 1968. Weighted kappa: Nominal scale agreement provision for scaled disagreement or partial credit. *Psychological bulletin* 70, 4 (1968), 213.

[9] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. 2009. Imagenet: A large-scale hierarchical image database. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*. IEEE, 248–255.

[10] Georges E Dupret and Benjamin Piwowarski. 2008. A user browsing model to predict search engine click data from past observations.. In *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, 331–338.

[11] Mark Everingham, SM Ali Eslami, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. 2015. The pascal visual object classes challenge: A retrospective. *International journal of computer vision* 111, 1 (2015), 98–136.

[12] Yixing Fan, Jiafeng Guo, Yanyan Lan, Jun Xu, Liang Pang, and Xueqi Cheng. 2017. Learning Visual Features from Snapshots for Web Search. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*. ACM, 247–256.

[13] Yoav Freund, Raj D. Iyer, Robert E. Schapire, and Yoram Singer. 1998. An Efficient Boosting Algorithm for Combining Preferences. In *Fifteenth International Conference on Machine Learning*. 170–178.

[14] Jerome H. Friedman. 2001. Greedy Function Approximation: A Gradient Boosting Machine. *Annals of Statistics* 29, 5 (2001), 1189–1232.

[15] Fan Guo, Chao Liu, and Yi Min Wang. 2009. Efficient multiple-click models in web search. In *Proceedings of the Second ACM International Conference on Web Search and Data Mining*. ACM, 124–131.

[16] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 770–778.

[17] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation* 9, 8 (1997), 1735–1780.

[18] Baotian Hu, Zhengdong Lu, Hang Li, and Qingcai Chen. 2014. Convolutional neural network architectures for matching natural language sentences. In *International Conference on Neural Information Processing Systems*. 2042–2050.

[19] Thorsten Joachims. 2002. Optimizing search engines using clickthrough data. In *ACM Conference on Knowledge Discovery and Data Mining*. 133–142.

[20] D Kinga and J Ba Adam. 2015. A method for stochastic optimization. In *International Conference on Learning Representations (ICLR)*.

[21] Ranjay Krishna, Yuke Zhu, Oliver Groth, Justin Johnson, Kenji Hata, Joshua Kravitz, Stephanie Chen, Yannis Kalantidis, Li-Jia Li, David A Shamma, et al. 2017. Visual genome: Connecting language and vision using crowdsourced dense image annotations. *International Journal of Computer Vision* 123, 1 (2017), 32–73.

[22] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. 2012. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*. 1097–1105.

[23] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. 2014. Microsoft coco: Common objects in context. In *European conference on computer vision*. Springer, 740–755.

[24] Jiasen Lu, Caiming Xiong, Devi Parikh, and Richard Socher. 2016. Knowing when to look: Adaptive attention via A visual sentinel for image captioning. *arXiv preprint arXiv:1612.01887* (2016).

[25] Cheng Luo, Yiqun Liu, Tetsuya Sakai, Fan Zhang, Min Zhang, and Shaoping Ma. 2017. Evaluating Mobile Search with Height-Biased Gain. (2017).

[26] Minh Thang Luong, Hieu Pham, and Christopher D Manning. 2015. Effective Approaches to Attention-based Neural Machine Translation. *Computer Science* (2015).

[27] Thomas Mandl. 2006. Implementation and evaluation of a quality-based search engine. In *Proceedings of the seventeenth conference on Hypertext and hypermedia*. ACM, 73–84.

[28] SchÃijtze Manning Raghavan. 2008. Introduction to Information Retrieval. *Journal of the American Society for Information Science & Technology* 43, 3 (2008), 824–825.

[29] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Distributed Representations of Words and Phrases and their Compositionality. *Advances in Neural Information Processing Systems* 26 (2013), 3111–3119.

[30] L Page. 1999. The PageRank Citation Ranking : Bringing Order to the Web. *Stanford Digital Libraries Working Paper* 9, 1 (1999), 1–14.

[31] Liang Pang, Yanyan Lan, Jiafeng Guo, Jun Xu, Shengxian Wan, and Xueqi Cheng. 2016. Text Matching as Image Recognition.. In *AAAI*. 2793–2799.

[32] Tao Qin, Tie Yan Liu, Jun Xu, and Hang Li. 2010. LETOR: A benchmark collection for research on learning to rank for information retrieval. *Information Retrieval* 13, 4 (2010), 346–374.

[33] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. 2017. Faster r-cnn: Towards real-time object detection with region proposal networks. *IEEE transactions on pattern analysis and machine intelligence* 39, 6 (2017), 1137–1149.

[34] Stephen Robertson and Hugo Zaragoza. 2009. The Probabilistic Relevance Framework: BM25 and Beyond. *Foundations & TrendsÂő in Information Retrieval* 3, 4 (2009), 333–389.

[35] Kalervo Rvelin, Kek, and Jaana Inen. 2002. Cumulated gain-based evaluation of IR techniques. *Acm Transactions on Information Systems* 20, 4 (2002), 422–446.

[36] Karen Simonyan and Andrew Zisserman. 2014. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556* (2014).

[37] Ruihua Song, Haifeng Liu, Ji-Rong Wen, and Wei-Ying Ma. 2004. Learning block importance models for web pages. In *Proceedings of the 13th international conference on World Wide Web*. ACM, 203–211.

[38] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. 2015. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 1–9.

[39] Shengxian Wan, Yanyan Lan, Jiafeng Guo, Jun Xu, Liang Pang, and Xueqi Cheng. 2015. A Deep Architecture for Semantic Matching with Multiple Positional Sentence Representations. (2015), 2835–2841.

[40] Chao Wang, Yiqun Liu, Meng Wang, Ke Zhou, Jian-yun Nie, and Shaoping Ma. 2015. Incorporating non-sequential behavior into click models. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, 283–292.

[41] Chao Wang, Yiqun Liu, Min Zhang, Shaoping Ma, Meihong Zheng, Jing Qian, and Kuo Zhang. 2013. Incorporating vertical results into search click models. In *Proceedings of the 36th international ACM SIGIR conference on Research and development in information retrieval*. ACM, 503–512.

[42] Qiang Wu, Christopher J. C. Burges, Krysta M. Svore, and Jianfeng Gao. 2010. Adapting boosting for information retrieval measures. *Information Retrieval* 13, 3 (2010), 254–270.

[43] Jun Xu and Hang Li. 2007. AdaRank:a boosting algorithm for information retrieval. 391–398.

[44] Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhutdinov, Richard Zemel, and Yoshua Bengio. 2015. Show, Attend and Tell: Neural Image Caption Generation with Visual Attention. *Computer Science* (2015), 2048–2057.

[45] Dawei Yin, Yuening Hu, Jiliang Tang, Tim Daly, Mianwei Zhou, Hua Ouyang, Jianhui Chen, Changsung Kang, Hongbo Deng, and Chikashi Nobata. 2016. Ranking Relevance in Yahoo Search. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 323–332.

[46] Masrour Zoghi, Tomáš Tunys, Lihong Li, Damien Jose, Junyan Chen, Chun Ming Chin, and Maarten de Rijke. 2016. Click-based hot fixes for underperforming torso queries. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*. ACM, 195–204.