data

MDPI

Review

Neural Networks in Big Data and Web Search †

Will Serrano

Intelligent Systems and Networks Group, Imperial College London, SW7 2AZ London, UK; G.Serrano11@imperial.ac.uk

† This article is an extended version of the paper "Will Serrano, The Random Neural Network and Web Search: Survey Paper" presented in Intelligent Systems Conference (2018), 6–7 September 2018, London, UK.

Received: 4 November 2018; Accepted: 24 December 2018; Published: 30 December 2018

Abstract: As digitalization is gradually transforming reality into Big Data, Web search engines and recommender systems are fundamental user experience interfaces to make the generated Big Data within the Web as visible or invisible information to Web users. In addition to the challenge of crawling and indexing information within the enormous size and scale of the Internet, e-commerce customers and general Web users should not stay confident that the products suggested or results displayed are either complete or relevant to their search aspirations due to the commercial background of the search service. The economic priority of Web-related businesses requires a higher rank on Web snippets or product suggestions in order to receive additional customers. On the other hand, web search engine and recommender system revenue is obtained from advertisements and pay-per-click. The essential user experience is the self-assurance that the results provided are relevant and exhaustive. This survey paper presents a review of neural networks in Big Data and web search that covers web search engines, ranking algorithms, citation analysis and recommender systems. The use of artificial intelligence (AI) based on neural networks and deep learning in learning relevance and ranking is also analyzed, including its utilization in Big Data analysis and semantic applications. Finally, the random neural network is presented with its practical applications to reasoning approaches for knowledge extraction.

Keywords: big data; neural networks; ranking algorithms; web search; deep learning; recommender systems

1. Introduction

The Internet has enabled the direct connection between users and information. This has fundamentally changed how businesses operate, including the travel industry and e-commerce. The Internet provides real-time information and the direct purchase of services and products. Web users can directly buy flight tickets, hotel rooms, and holiday packs. Travel industry supply charges have been eliminated or decreased because the Internet has provided a shorter value chain; however, services or products not displayed within the higher order of web search engines or recommender systems lose tentative customers. A similar scenario also applies in academic and publication searches where the Internet has permitted the open publication and accessibility of academic research. Authors are able to avoid the conventional method of human evaluation of the journal and upload their work onto their personal websites. With the intention to expand the research contribution to a wider number of readers and be cited more, authors have the personal interest to show publications in high academic search rank orders.

Web search engines and recommender systems were developed as interfaces between users and the Internet to address the need for searching precise data and items. Although they provide a straight link between web users with the pursued products or wanted information, any web search result list or suggestion will be biased due profitable economic or personal interests along with by the users' own

inaccuracy when typing their queries or requests. Sponsored search enables the economic revenue that is needed by web search engines [1]; it is also vital for the survival of numerous web businesses and the main source of income for free to use online services. Multiple payment options adapt to different advertiser targets while allowing a balanced risk share among the advertiser and the web search engine for which the pay-per-click method is the widest used model.

Ranking algorithms are critical in the presented examples as they decide on the result relevance and order, therefore, marking data as transparent or nontransparent to e-commerce customers and general web users. Considering the web search commercial model, businesses or authors are interested in distorting ranking algorithms by falsely enhancing the appearance of their publications or items, whereas web search engines or recommender systems are biased to pretend relevance exists with respect to the rank in which they order results from explicit businesses or web sites in exchange for a commission or payment. The main consequence for a web user is that relevant products or results may be "hidden" or displayed at the very low order of the search list and unrelated products or results on a higher order.

Searching for information or meaning needs three elements: a universe formed of entities or ideas to be searched, a high-level query that specifies the properties or concepts requested by a user, and a method that searches and selects entities from the universe according to an algorithm or rule. The examples used by animals when they search in a large unknown space were investigated and the role of teamwork was discussed [2]; these ideas were then applied to the search for information when N concurrent "search agents" with similar characteristics are being used [3] establishing that the total average search time can be reduced by aborting and re-starting a search process after a pre-determined time-out if it has not been successful. The use of more concurrent agents can actually reduce the total energy costs, despite the increase in the number of agents, as well as the search times [4]. These results were confirmed in terms of large datasets distributed over large networks [5].

Artificial neural networks are models based on the brain within the central nervous system; they are usually presented as artificial nodes or "neurons" in different layers connected together via synapses. The learning properties of artificial neural networks have been applied to resolve extensive and diverse tasks that would have been difficult to solve by ordinary rules-based programming. Neural networks and artificial intelligence have also been applied to web searching in result ranking and relevance as a method to learn and adapt to variable user interests.

This paper presents a survey of web searches in Section 2, including internet assistants, web search engines, meta-search engines, web result clustering, travel services, and citation analysis. Ranking is described in Section 3; with ranking algorithms, relevance metrics and learning to rank. We define recommender systems in Section 4 and neural networks in web searches, learning to rank, recommender systems, and deep learning are analyzed in Section 5, including the random neural network and its application to web searches and ranking algorithms. Finally, conclusions are explained on Section 6, followed by the survey bibliography.

2. Web Search

With the development of the Internet, several user semantic interfaces, applications, and services have been proposed or developed to manage the increasingly greater volume of information and data accessible in the World Wide Web.

2.1. Internet Assistants

Internet assistants learn and adapt to variable user's interests in order to filter and recommend information. These agents normally define a user as a set of weighted terms which are either explicitly introduced or implicitly extracted from the web browsing behavior. Relevance algorithms are determined by a vector space model that models both query and answer as an ordered vector of weighed terms. Web results are the parsed fragments obtained from either Web pages, documents or

web results retrieved from different sources. The user provides explicit or implicit feedback on the results considered relevant or interesting. This is then used to adapt the weights of the term set profile.

Intelligent agents [6] are defined as a self-contained independent software module or computer program that performs simple and structurally repetitive automated actions or tasks in the representation of web users while cooperating with other intelligent agents or humans. Their attributes are autonomy, cooperation with other agents, and learning from interaction with the environment and the interface with users' preferences and behavior. Intelligent agents [7] behave in a manner analogous to a human agent with autonomy, adaptability and mobility as desirable qualities. They have two ways to make the web invisible to the user: by abstraction where the used technology and the resources accessed by the agent are user-transparent, and by distraction, where the agent runs in parallel to the web user and performs tedious and complex tasks faster than would be possible for a human alone.

Spider Agent [8] is a metagenetic assistant to whom the user provides a set of relevant documents where the N highest frequency keywords form a dictionary which is represented as a $N \times 3$ matrix. The first column of the dictionary contains the keywords, whereas the second column measures the total amount of documents that contain the keywords. Finally, the third column contains the sum frequency of the specific word over the overall documents. The metagenetic algorithm first creates a population of keyword sets from the dictionary based on three genetic operators: crossover, mutation, and inversion. Then it creates a population of logic operators sets (AND, OR, NOT) for each of the first populations. Spider Agent forms different queries by the combination of both populations and searches for relevant documents for each combination. The main concept is that different combinations of words in different queries may result in search engines providing additional different relevant results.

Syskill and Webert [9] helps users to select relevant web pages on specific topics where each user has a set of profiles, one for each topic, and web pages are rated as relevant or irrelevant. Syskill and Webert transforms the source code of the web page, based on the Hyper Text Markup Language (HTML), into a binary feature vector which designates the presence of words using a learning algorithm based on a naive Bayesian classifier. Letizia [10] is a web user interface agent that helps web browsing. Letizia learns user behavior and provides additional interesting web pages by exploring the current web page links where the user interest assigned to a web document is calculated as the reading time, the addition to favorites, or the click of a shown link.

WebWatcher [11] is a web tour guide agent that provides relevant web links to the user while browsing the web; it acts as a learning apprentice observing and learning interest from its user actions when selecting relevant links. WebWatcher uses reinforcement learning where the reward is the frequency of each the searched terms within the web page recommending web pages that maximize the reward path to users with similar queries.

Lifestyle Finder [12] is an agent that generates user profiles with a large-scale database of demographic data. Users and their interests are grouped by their input data according to their demographic information. Lifestyle Finder generalizes user profiles along with common patterns within the population; if the user data corresponds to more than one cluster, the demographic variables whose estimates are close with the entire corresponding groups generate a limited user profile. The demographic feature that best distinguishes the corresponding groups is utilized to ask the Web user for additional details where the final group of corresponding clusters is obtained after several user iterations.

2.2. Web Search Engines

Internet assistants have not been practically adopted by Internet users as an interface to reach relevant information. Instead, web search engines are the preferred option as the portal between users and the Internet. Web search engines are software applications that search for information in the World Wide Web while retrieving data from web sites and online databases or web directories. Web search engines have already crawled the web, fetched its information, and indexed it into databases so when the user types a query, relevant results are retrieved and presented promptly (Figure 1). The main

issues of web search engines are result overlap, rank relevance, and adequate coverage [13] for both sponsored and non-sponsored results.

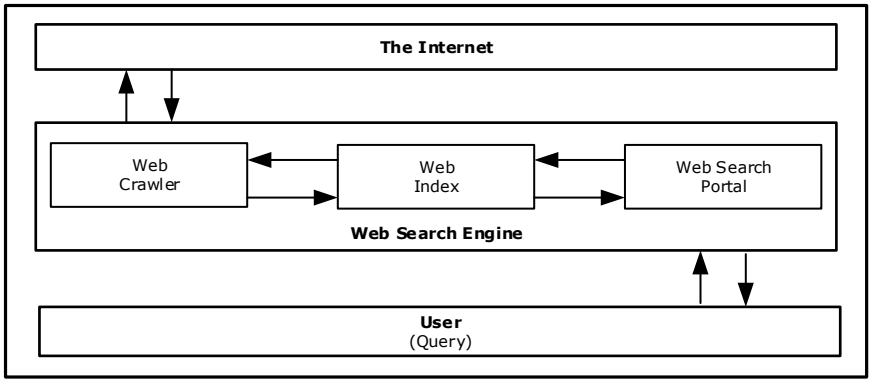

Figure 1. Web search engine architecture.

Web personalization builds the user's interest profile by using their browsing behavior and the content of the visited web pages to increase result extraction and rank efficiency in web searches. A model [14] that represents a user's needs and its search context is based on content and collaborative personalization, implicit and explicit feedback, and contextual search. A user is modeled by a set of terms and weights [15] related to a register of clicked URLs with the amount of visits to each, and a set of previous searches and web results visited; this model is applied to re-order the web search results provided by a non-personalized web search engine. Web queries are associated to one or more related web page features [16] and a group of documents is associated with each feature; a document is both associated with the feature and relevant to the query. These double profiles are merged to attribute a web query with a group of features that define the user's search relevance; the algorithm then expands the query during the web search by using the group of categories. There are different methods to improve the personalized web search based on the increment of the diversity of the top results [17] where personalization comprises the re-ranking of the first N search results to the probable order desired by the user and query reformulations are implemented to include variety; the three diversity methods proposed are: *the most frequent* selects the queries that most frequently precede the user query; *the maximum result variety* choses queries that have been frequently reformulated but distinct from the already chosen queries; and finally, *the most satisfied* selects queries that usually are not additionally reformulated, but they have a minimum frequency.

2.2.1. Spatial Search Variation

Spatial variation based on information from web search engine query records and geolocation methods can be included in web search queries [18] to provide results focused on marketing and advertising, geographic information, or local news; accurate locations are assigned to the IP addresses that issue the queries. The center of a topic is calculated by the physical areas of the users searching for it where a probabilistic model calculates the greatest probability figure for a geographic center and the geographical dispersion of the query importance. Aspects for a web query are calculated [19] as an effective tool to explore a general topic in the web; each aspect is considered as a set of search terms which symbolizes different information requests relevant to the original search query. Aspects are independent from each other while having a high combined coverage, two sources of information are combined to expand the user search terms: query logs and mass collaboration knowledge databases, such as Wikipedia. Spatial search variation can also be combined with time search variation [20] where a user can specify both the temporal and spatial attributes in addition to spatial attributes of the geographic entities in a time-specific regional web search system

2.2.2. Time Search Variation

Time is a key consideration role in Web search because web queries may be time-related: user relevance varies with time and most web pages contain temporal information. The web is not a static environment, its information changes constantly; relevant pages in the past may not be relevant pages at the present or in the future; showing new and relevant results to users is key as users want the latest information. The inclusion of time in web searches presents three issues [21]: the extraction of implicit temporal expressions from web pages, the determination of the precise and explicit time from the extracted temporal information, and the integration of the precise time into a search engine. Current web ranking techniques favor older pages because these pages have many in-links accumulated over time whereas new pages, which may be highly relevant, have few or no in-links, therefore, showing a lower rank [22]. In order to address this issue a timed PageRank is proposed with an aging factor weight that penalizes older pages. The best results of temporally-dependent searches change over time, therefore, these results should be fresh and most current [23], and a web search engine must be able to detect that certain queries have an implicit temporal intent and use this information to improve the search quality. Factual information associated with some targeted entity, such as time [24], can be represented as snippets and considered as a document sentence fragment that encodes the full length of the web page. Offering the user a comprehensive temporal perspective of a topic by clustering relevant temporal web search results is intuitively more informative than retrieving a result that only contains topical information [25]. A concept can be annotated with temporal, precise, and structured information that reflects its explicit and complex meanings to be used in trend analysis, aspect exploration and query suggestion [26]; the temporal semantic context can help users to learn, understand, and search unfamiliar or newly emerged concepts on the web, such as LinkedIn or Wikipedia, without any need for any prior knowledge, such as ontology or a hierarchical knowledge base.

2.3. Metasearch Engines

Metasearch engines were developed based on the concept that single web search engines were not able to crawl and index the entire web. While web search engines are useful for finding specific information, like home pages, they may be less effective with a comprehensive search or wide queries due to their result overlap and limited coverage area. Metasearch engines try to compensate their disadvantages by sending user queries to different web search engines simultaneously, databases, web directories, and digital libraries and combining their results into a single ranked list (Figures 2 and 3). The main operational difference with web search engines is that metasearch engines do not index the web.

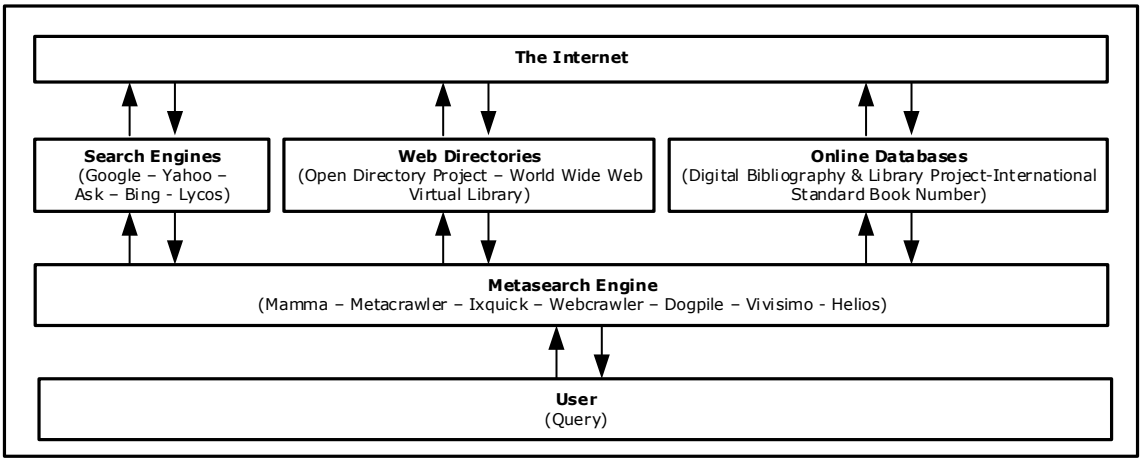

Figure 2. Metasearch service model.

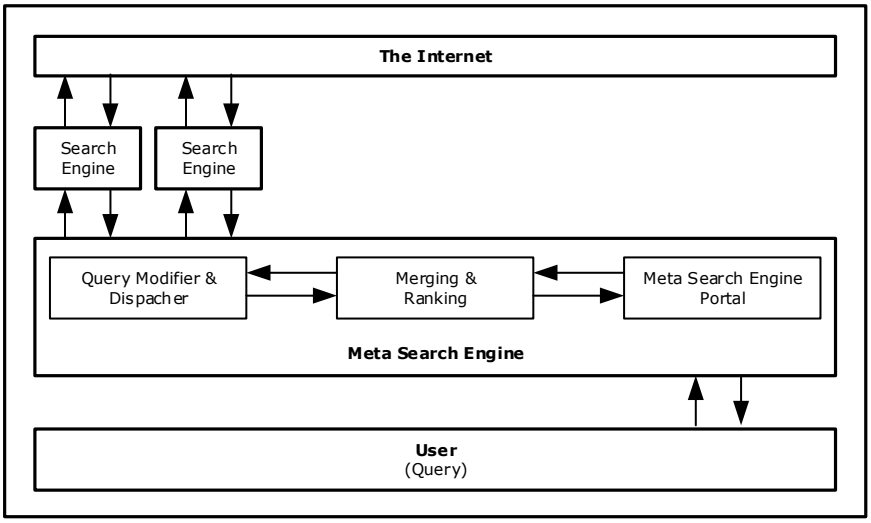

Figure 3. Metasearch engine architecture.

There are challenges for developing metasearch engines [27]: different web search engines are expected to provide relevant results which must be selected and combined. Different parameters shall be considered when developing a metasearch engine [28]: functionality, working principles—including querying, collection, and fusion of results—architecture, and underlying technology, growth, evolution, and popularity. Numerous metasearch architectures can be found [29] such as Helios, Tadpole, and Treemap, with different query dispatcher, result merger, and ranking configurations. Helios [30] is an open-source metasearch engine that runs above different web search engines where additional ones can be flexibly plugged into the architecture. Helios retrieves, parses, merges, and reorders results given by the independent web search engines.

An extensive modular metasearch engine with automatic search engine discovery [31] that incorporates several autonomous search engines is based on three components: automatic web search engine recognition, automatic web search engine interconnection, and automatic web search result retrieval. The solution crawls and fetches web pages choosing the web search engine ones, which, once discovered, are connected by parsing the HTML source code, extracting the form parameters and attributes, and sending the query to be searched. Finally, URLs or snippets provided by the different web search engines are extracted and displayed to the web user.

There are several rank aggregation metasearch models to combine results in a way that optimizes their relevance position based on the local result ranks, titles, snippets, and entire Web pages. The main aggregation models are defined as [32]: Borda-fuse is founded on the Borda Count, a voting mechanism on which each voter orders a set of web results; the web result with the most points wins the election. The Bayes-fuse is built on a Bayesian model of the probability of web result relevance. The difference between the probability of relevance and irrelevance, respectively, determines relevance of a web Page based on the Bayes optimal decision rule. Rank aggregation [33] can be designed against spam, search engine commercial interest, and coverage.

The use of the titles and associated snippets [34] can produce a higher success than parsing the entire web pages where the effective algorithm of a metasearch engine for web result merging outperforms the best individual web search engine. A formal approach to normalize scores for metasearch by balancing the distributions of scores of the first irrelevant documents [35] is achieved by using two different methods: the distribution of scores of all documents and the combination of an exponential and a Gaussian distribution to the ranks of the documents where the developed exponential distribution is used as an approximation of the irrelevant distribution.

Metasearch engines provide personalized results using different approaches [36]. The first method provides distinct results for users in separate geographical places where the area information is acquired for the user and web page server. The second method has a ranking method where the

user can establish the relevance of every web search engine and adjusts its weight in the result ranking in order to obtain tailored information. The third methods analyses web domain structures of web pages to enable the user the possibility to limit the amount of web pages with close subdomains.

Results from different databases present in the web can also be merged and ranked [37] with a representative database that is highly scalable to a large number of databases and representative for all local databases. The method assigns only a reduced but relevant number of databases for each query term where single-term queries are assigned the right databases and multi-term queries are examined to extract dependencies between them in order to generate phrases with adjacent terms. Metasearch performance against web search has been widely studied [38] where the search capabilities of two metasearch engines, Metacrawler and Dogpile, and two web search engines, Yahoo and Google, are compared.

2.4. Web Result Clustering

Web result clustering groups results into different topics or categories in addition to web search engines that present a plain list of result to the user where similar topic results are scattered. This feature is valuable in general topic queries because the user gets the theme bigger picture of the created result clusters (Figure 4). There are two different clustering methods; pre-clustering calculates first the proximity between web pages and then it assigns the labels to the defined groups [39] whereas post-clustering discovers first the cluster labels and then it assigns web pages to them [40]. Web clustering engines shall provide specific additional features [41] in order to be successful: fast subtopic retrieval, topic exploration and reduction of browsing for information. The main challenges of web clustering are overlapping clusters [42], precise labels [43], undefined cluster number [44], and computational efficiency.

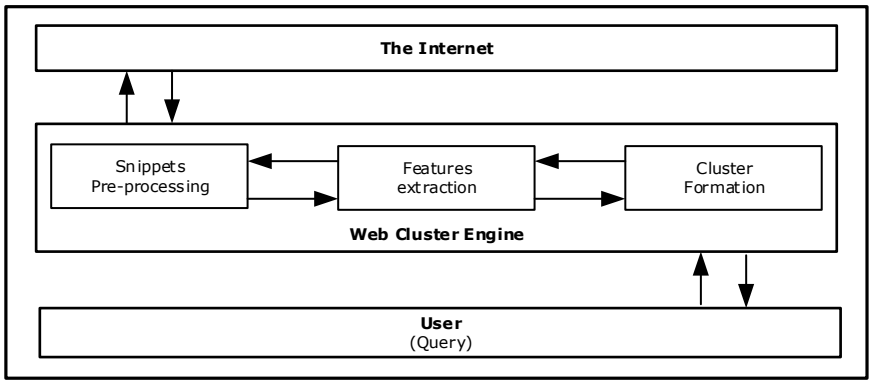

Figure 4. Web cluster engine architecture.

Suffix tree clustering (STC) [39] is an incremental time method which generates clusters using shared phrases between documents. STC considers a web page as an ordered sequence of words and uses the proximity between them to identify sets of documents with common phrases between them in order to generate clusters. STC has three stages: document formatting, base clusters identification, and its final combination into clusters.

The Lingo algorithm [40] finds first clusters utilizing the vector space model to create a "T×D term document matrix where T is the number of unique terms and D is the number of documents". A singular value decomposition method is used to discover the relevant matrix orthogonal basis where orthogonal vectors correspond to the cluster labels. Once clusters are defined documents are assigned to them.

A cluster scoring function and selection algorithm [42] overcomes the overlapping cluster issue; both methods are merged with suffix tree clustering to create a new clustering algorithm called extended suffix tree clustering (ESTC) which decreases the amount of the clusters and defines the most useful clusters. This cluster scoring function relies on the quantity of different documents in the cluster

where the selection algorithm is based on the most effective collection of clusters that have marginal overlay and maximum coverage; this makes the greatest amount of different clusters where most of the documents are covered in them.

A cluster labelling strategy [43] combines intra-cluster and inter-cluster term extraction where snippets are clustered by mapping them into a vector space based on the metric k-center assigned with a distance function metric. To achieve an optimum compromise between defining and distinctive labels, prospect terms are selected for each cluster applying an improved model of the information gain measurement. Cluster labels are defined as a supervised salient phrase ranking method [44] based on five properties: "sentence frequency/inverted document frequency, sentence length, intra-cluster document similarity, cluster overlap entropy, and sentence independency" where a supervised regression model is used to extract possible cluster labels and human validators are asked to rank them.

Cluster diversification is a strategy used in ambiguous or broad topic queries that takes into consideration its contained terms and other associated words with comparable significance. A method clusters the top ranked web pages [45] and those clusters are ordered following to their importance to the original query however diversification is limited to documents that are assigned to top ranked clusters because potentially they contain a greater number of relevant documents. Tolerance classes approximate concepts in web pages to enhance the snippets concept vector [46] where a set of web pages with similar enhanced terms are clustered together. A technique that learns relevant concepts of similar topics from previous search logs and generates cluster labels [47] clusters web search results based on these as the labels may be better than those calculated from the current terms of the search result.

Cluster hierarchy groups data over a variety of scales by creating a cluster tree. A hierarchy of snippets [48] is based on phrases where the snippets are assigned to them; the method extracts all salient phrases to build a document index assigning a cluster per salient phrase; then it merges similar clusters by selecting the phrase with highest number of indexing documents as the new cluster label. Finally, it assigns the snippets whose indexing phrases belong to the same cluster where the remaining snippets are assigned to neighbors based on their k-nearest distance.

Web search results are automatically grouped through semantic, hierarchical, online clustering (SHOC) [49]. A web page is considered as a string of characters where a cluster label is defined as a meaningful substring which is both specific and significant. The latent semantic of documents is calculated through the analysis of the associations between terms and web pages. Terms assigned with the same web page should be close in semantic space, the same as the web pages assigned with the same terms. Densely-allocated terms or web pages are close to each other in semantic space, therefore, they should be assigned to the same cluster. Snake [50] is a clustering algorithm based on two information databases. The anchor text and link database assigns each web document to the terms contained in the web document itself and the words included in the anchor text related to every link in the document, the semantic database ranks a group of prospect phrases and choses the most significant ones as labels. Finally, a hierarchical organizer combines numerous base clusters into a few super-clusters. A query clustering approach [51] calculates the relevance of web pages using previous choices from earlier users, the method applies a clustering process where collections of semantically-similar queries are identified and the similarity between a couple of queries is provided by the percentage of shared words in the clicked URL within the web results.

2.5. Web Search Business Case 1: Travel Services

Originally, travel service providers, such as airlines or hotels, used global distribution systems to combine their offered services to travel agencies. Global distribution systems required high investments due to their technical complexity and physical dimensions as they were mainframe-based infrastructure; this generated the monopoly of a few companies that charged a high rate to the travel services providers in order to offer their services to the travel agencies [52]. With this traditional model, customers could purchase travel provider services directly at their offices or through a travel agent.

This scenario has now been changed by two factors: the Internet has enabled e-commerce—the direct accessibility of customers to travel services providers' information in real-time with the availability of online purchase [53]—and higher computational servers and software applications can implement the same services as the global distribution systems did, at a lower cost.

As a consequence, new players have entered to this scenario [54]; software database applications, such as ITA Software or G2 SwitchWorks, use high computational server applications and search algorithms to process the data provided from travel services providers; they are the replacement of the global distribution systems. Online travel agents, such as Expedia or TripAdvisor, are traditional travel agents that have automatized the global distribution systems interface: customers can interact with them directly through the Internet and buy the final chosen product. Metasearch engines like Cheapflights or Skyscanner, among many others, use the Internet to search customers' travel preferences within travel services providers and online travel agents however customers cannot purchase products directly; they are mainly referred to the supplier web site (Figure 5).

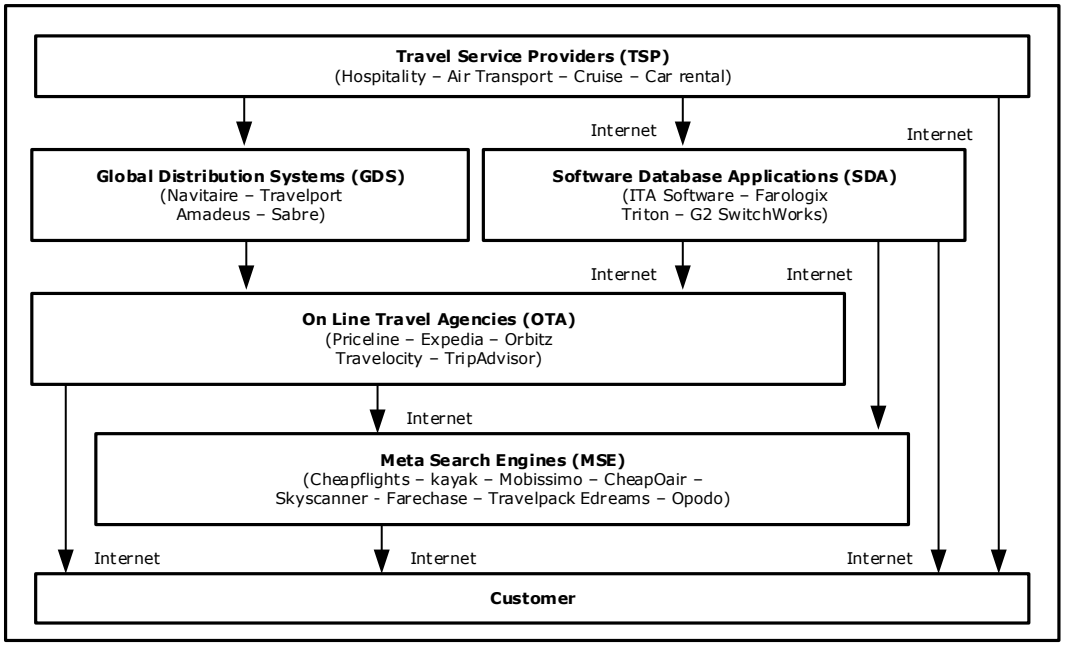

Figure 5. Metasearch engine architecture.

Other players that have an active part in this sector are Google and other web search engines [55]; they provide search facilities that connect directly travel service providers with consumers bypassing global distribution systems, software distribution systems, Metasearch engines and online travel agents. This allows the direct product purchase that can reduce distribution costs due to a shorter value chain. Travel ranking systems [56] also recommend products and provide the greatest value for the consumer's budget with a crucial concept where products that offer a higher benefit are ranked in higher positions.

With the integration of the customer and the travel service provider through the Internet, different software applications have been developed to provide extra information or to guide through the purchase process [57] based on user interactions. Travel related web interfaces [58] help users to make the process of planning their trip more entertaining and engaging while influencing users' perceptions and decisions. There are different information search patterns and strategies [59] within specific tourism domain web search queries based on search time, IP address, language, city, and key terms; the study shows that clicking and re-clicking on the search engines homepage is the most repetitive and iterative task of web users, the city is most likely to be included as a key term in the search queries,

followed by the hotel where, finally, web users normally select the results that show the city home web page.

2.6. Web Search Business Case 2: Citation Analisys

Citation analysis consists on the access and referral of research information including articles, papers and publications. Its importance is based on its measurement of the value or relevance of personal academic work; it is becoming more respected since its evaluation impacts on other decisions, such as promotion, scholarships, and research funds. The journal impact factor [60] is calculated as "the number of citations received in the current year to articles published in the two preceding years divided by the total number of articles published in the same two years". It is based on the concept that high-impact journals will only try to publish very relevant work, therefore, they will be used for outstanding academics. These traditional measurements are peer review based on human judgment, however, it is time consuming and assessors may be biased by personal judgments, such as the quality of journal or the status of the university, instead of competent and impartial reviews.

The web has provided the possibility of other alternative citation measures making the citation monopoly to change because of the accessibly it provides. New online platforms [61] such as Scopus, Web of Science, Google Scholar, Microsoft Academic, and Semantic Scholar provide other measures that may represent more accurately the relevance of an academic contribution. Bibliometric indicators [62] evaluate the quality of the publication based on empirical values; mostly number of citations or number of downloads. They are not only quicker to obtain but they also provide a wider coverage because they also retrieve results from relevant academic works published on personal or academic web pages and open journals based on conference papers, book chapters, or theses. In addition, academics are personally interested to artificially influence ranking methods by "optimizing" the appearance of their research publications to show them in higher ranks with the purpose to extend their reader audience [63] in order to get cited more.

The h-index [64] is applied to qualify the contribution of individual scientific research work; "An academic has an h-index of h if the x publications have individually received at least h citations and the other $(x - h)$ publications have up to h citations". The h-index has become popular among researchers because it solves the disadvantages of the journal impact factors and it has also a good correlation with the citation analysis database. It is simple and fast to calculate utilizing databases like Google Scholar, Microsoft Academic, Semantic Scholar, Scopus, or Web of Science, however, it suffers some disadvantages [60]: it does not differentiate negative citations from positive ones, so both are equally taken into account; it does not consider the entire number of citations an author has collected; and it disadvantages a reduced but highly-cited paper set very strongly.

Google Scholar, Semantic Scholar, Microsoft Academic, DBLP Computer Science Bibliography, CiteSeerX, and IEEE Xplore are free web academic search engines that register the metadata or the complete text of academic articles from different databases and journals. Google Scholar result relevance is calculated by a ranking algorithm that combines the term occurrence, author, publications, and citation weights [65]. Google Scholar has received both good and bad critics since it started in 2004 [66]: the lack of transparency in the relevance assessment of the information due to its automatic process of retrieving, indexing, and storing information has been criticized; however, the more transparent its evaluation algorithms, the easier it becomes to influence its metrics. Google Scholar has been widely analyzed and studied by the research community against other online databases [67,68].

A hybrid recommender system based on content-based and collaborative-based relevance [69] expands the method of the traditional keyword or key term search by adding citation examination, author examination, source examination, implicit ratings and explicit marks. Two ranking methods are applied: "in-text citation frequency examination" evaluates the occurrence which a publication is "cited within the citing document" and "in-text citation distance examination" determines the similarity between references within a publication by evaluating the word separation.

3. Ranking

Ranking is the core process on many information retrieval applications and Internet user semantic interfaces, including web search engines and recommender systems, for two main reasons: the user evaluation of quality or performance from a customer point of view and sponsored adverts based on ranking from a commercial perspective. There are several methods and techniques for ranking web pages and products that exploit different algorithms built on the analysis of the web page content, the anchor text of the links between pages or the network structure of the World Wide Web hyperlinked environment.

3.1. Ranking Algorithms

Web search engine evaluation comprises the measurement of the quality and services they provider to users; it is also used to compare their different search capabilities. Automatic evaluation analyses the different presented URLs whereas human evaluation is based on the measurement of result relevance. While the latest one is more precise, it takes more effort, and therefore is more expensive, as it is normally done by surveys. Other methods that combine both automatic and user evaluation are applications that monitor user activities such as click-troughs or time spent of each web site. "Dynamic ranking or query-dependent ranking" improves the order of the results returned to the user based on the query, while "static ranking or query-independent" calculates the relevance of the web pages to different topics.

Web spamming refers to any deliberate human action with the purpose of generating an unjustifiably favorable relevance or rank for some web pages, against the page's true value and purpose [70]. There are two main categories of web spamming: *boosting techniques* influence the ranking algorithms used by search engines based on term and link spamming that optimizes the keywords of the web Pages or the links pointing between them, respectively, and *hiding techniques* conceal the applied boosting methods from regular web users that visit spam pages. Hiding techniques are based on web Page content hiding, cloaking from web crawlers, and URL redirection.

There are different methods to assess web pages relevance.

3.1.1. HTML Properties

The first method focuses on the HyperText Markup Language (HTML) properties of the web pages. The internal web page structure consists of specified text segments defined by semantic tags or mark up. HTML defines a set of roles assigned to the text in a web document related to formatting (bold or italics) or richer semantic values such as headlines or hyperlinks to other web pages. Web spammers mostly base their strategy in the design of the HTML properties of the web page to get an artificial better rank from web search engines.

HTML ranking algorithms make use of the structure HTML imposes on web documents and the relative order between terms and semantic tags [71]; the parameters for weighting words in the HTML fields are based on type of text, section, size, position, anchor text and URL. The structure of a HTML page can be represented as a tag tree by a Document Object Model (DOM). Although DOM, in general, provides a useful structure for a web page, its defined tags are used not only for content organization, but also for layout presentation. DOM analysis tends to disclose the presentation structure rather than the content structure [72]; it is often not accurate enough to discriminate different semantic blocks in a web page, therefore, ranking algorithms based only on HTML properties will not accurately measure the meaning or relevance of a web page. In order to address the DOM structure issues, a vision-based page segmentation algorithm divides a web page into semantically-related units [73]. The vision segmentation algorithm is based on two key concepts: the visual presentation of the page that groups together semantically related content and the division of the web page into regions for different contents using explicit or implicit visual separators [74].

3.1.2. TF-IDF

The second method is founded on the inspection of the web page text itself and how the words in the query relate within it. TF-IDF or "Term frequency and inverse document frequency" applies the vector space model with a scoring function where the importance of a term increases if it is repeated often in a document, however, its weight decreases if the same term appears multiple times on numerous documents as it is considered less descriptive:

$$TF - IDF(t, d, D) = TF(t, d) \times IDF(t, D) = \left[1 + \log f_{t,d}\right] \times \left[\log\left(1 + \frac{N}{d_t}\right)\right] \quad (1)$$

where t is the keyword, d is the web page, $f_{t,d}$ the amount of occurrences that keyword t is contained within the web page d, N is the entire amount of web pages, and dt is the quantity of web pages that include the keyword t.

A probabilistic TF-IDF retrieval model [75] emulates the decision making of a human brain for two categories of relevance: "local relevance" is first applied to a particular web page location or section, whereas the "wide relevance" spans to cover the whole web page; the method then merges the "local relevance" decisions for each position as the web page "wide relevance" decision. Document transformation adapts the web page vector close to the query vector by adding or removing terms. An example of long-term incremental learning [76] is based on a search log that stores information about click-throughs web users have browsed among the web result pages from a web search engine and web pages selected by the user. Content-time-based ranking is a web search time focus ranking algorithm [77] that includes both "text relevance" and "time relevance" of web pages. "Time relevance" is the interval computed between the query and the content or update time where every term in a web page is assigned to an explicit content time interval and, finally, the time focus TF-IDF value of each keyword is calculated to obtain the overall rank.

3.1.3. Link Analysis.

The third method is the link analysis which it is founded on the relation between web pages and their web graph.

In-Degree [78] was the first link analysis ranking algorithm; it ranks pages according to their popularity which it is calculated by the sum of links that point to the page:

$$\text{Degree}(p_i) = \sum_{p_j \in M(p_i)} p_j \quad (2)$$

where $p_1, p_2, \ldots p_N$ correspond to the web pages to be ranked and $M(p_i)$ defines the collection of web documents pointing to p_i.

Page Rank [79] is founded on the concept that a web document has a top "page rank" if there are many web pages with reduced "page rank" or a few web pages with elevated "page rank" that link to it. The Page Rank algorithm provides a likelihood distribution that represents the probability that a web user arbitrarily following the links will reach to the specific web page:

$$\text{PageRank}(p_i) = \frac{1 - d}{N} + d \sum_{p_j \in M(p_i)} \frac{\text{PageRank}(p_j)}{L(p_j)} \quad (3)$$

where $p_1, p_2, \ldots p_N$ correspond to the web pages to be ranked, d the dumping factor, $M(p_i)$ defines the collection of web documents pointing to p_i, $L(p_j)$ is the amount of external connections from page p_j, and N is the entire quantity of web pages.

The HITS algorithm [80], or Hyperlink Induced Topic Search algorithm, assigns relevance founded on the authority concept and the relation among a group of relevant "authoritative web pages for a topic" and the group of "hub web pages that link to many related authorities" in the web link

arrangement. The hyperlink structure among web pages is analyzed to formulate the notion of authority; the developer of web document p when inserts a hyperlink to web document q, provides some relevance or authority on q. Hubs and authorities have a reciprocally strengthening association: "a relevant hub is a web page that links to many relevant authorities; a relevant authority is a web page which is linked by numerous relevant hubs".

The authority value of a web document is the addition of every hub value of web documents linking to it:

$$\mathrm{auth}(p) = \sum_{i=1}^{n} \mathrm{hub}(i) \tag{4}$$

where p the relevant authority web document, i is a web document linked to p and n is the entire quantity of web documents linked to p.

The hub value of a web document is the addition of total number of authorities' values of web documents linking to it:

$$\mathrm{hub}(p) = \sum_{i=1}^{n} \mathrm{auth}(i) \tag{5}$$

where p is the relevant authority web document, i is a web document that p links to and n is the entire quantity of web documents p connects to.

The SALSA algorithm [81], or Stochastic Approach for Link-Structure Analysis, combines Page Rank and HITS by taking a random walk alternating between hubs and authorities. Two random walks are assigned two scores per web page; it is based on the in link and out link that represent the proportion of authority and hub respectively.

The hub array is expressed in the below sum:

$$h_{i,j} = \sum_{\{k|(i_h,k_a),(j_h,k_a)\in G\}} \frac{1}{\deg(i_h)} \times \frac{1}{\deg(k_a)} \tag{6}$$

The authority array is expressed in the below sum:

$$a_{i,j} = \sum_{\{k|(k_h,i_a),(k_h,j_a)\in G\}} \frac{1}{\deg(i_a)} \times \frac{1}{\deg(k_h)} \tag{7}$$

where web document h links to mutual documents i and j and deg is the amount of hyperlinks connecting to a page.

The QS page-rank is a query-sensitive algorithm [82] that combines both global scope and local relevance. Global web page importance is calculated by using general algorithms, such as Page Rank. Local query sensitive relevance is calculated by a voting system that measures the relation between the top web pages retrieved.

The priority of a query sensitiveness rank (QS) is defined where document 1 is ordered in front of document 2:

$$(QS_1 > QS_2) \;||\; (QS_1 == QS_2 \;\&\&\; PR_1 > PR_2) \tag{8}$$

The priority of a global importance rank (PR) is defined where document 1 is ordered in front of document 2:

$$(PR_1 > PR_2) \;||\; (PR_1 == PR_2 \;\&\&\; QS_1 > QS_2) \tag{9}$$

where QS_1 and QS_2 are the query sensitive values and PR_1 and PR_2 are the global importance values for document 1 and document 2, respectively.

DistanceRank [83] is an iterative algorithm built on reinforcement learning; the distance between pages is counted as punishment where distance is established as the number of links among two web pages. The objective of the method is the reduction of the punishment or distance therefore a web

document with reduced "distance" has a greater rank. The main concept is that as correlated web pages are connected between them; the distanced built method finds relevant web pages quicker:

$$\text{distance}_n[j] = (1 - \alpha) \times \text{distance}_{n-1}[j] + \alpha \times \min_i(\log(O[i]) + \text{distance}_{n-1}[i]) \quad (10)$$

where i is the collection of web documents linking to j and O(i) is the amount of outbound hyperlinks in web document i and α is the learning rate of the user.

DING is a dataset ranking [84] algorithm that calculates dataset ranks; it uses the connections among them to combine the resulting scores with internal semantic based ranking strategies. DING ranks in three stages: first, a dataset rank is calculated by an external dataset hyperlink measurement on the higher layer, then for each dataset, local ranks are calculated with hyperlink measurements between the inner entities and, finally, the relevance of each dataset entity is the combination of both external and internal ranks.

The internal rank score of entity e is calculated as:

$$r(e) = w_{\sigma,i,j} = LF(L_{\sigma,i,j}) \times IDF(\sigma) = \frac{|L_{\sigma,i,j}|}{\sum_{L_{r,i,k}} |L_{r,i,k}|} \times \log \frac{N}{1 + \text{frequency}(\sigma)} \quad (11)$$

where $L_{\sigma,i,j}$ is a linkset (σ is term, i is the source and j is the target), N represents the amount of datasets and freq(σ) defines the rate of appearance of the term σ in the group of datasets.

The dataset rank score is calculated as:

$$r^k(D_j) = \alpha \sum_{L_{\sigma,i,j}} r^{k-1}(D_i) w_{\sigma,i,j} + (1 - \alpha) \frac{|E_{D_j}|}{\sum_{D \in G} |E_D|} \quad (12)$$

where D is a dataset and E_{Dj} is the set of nodes within D_j.

The final global score of entity e for dataset D is:

$$r_g(e) = r(e) \times r(D) \quad (13)$$

where r(e) and r(D) are the internal rank entities and dataset rank score, respectively

ExpertRank [85] is an online community and discussion group expert ranking algorithm that integrates a vector space method to calculate the expert subject importance and a method similar to Page Rank algorithm to evaluate topic authority. The expert subject relevance score is calculated as a likeness of the candidate description and the provided request where the candidate description is generated by combining all the comments in the topic conversation group and the authority rating is computed according to the user interaction graph:

$$\text{Expert}(c, q) = i(RE(c, q), AU(c)) \quad (14)$$

where c is the candidate, q is the query, i is the integration method, RE(c,q) represents the relevance rating among the query q and candidate c knowledge, and AU(c) represents the authority rating.

3.2. Relevance Metrics

Performance can be described as the display of relevant, authoritative, updated results on first positions. Retrieval performance is mostly evaluated on two parameters: precision is the percentage of applicable results within the provided web document list and recall is the percentage of the entire applicable results covered within the provided web document list. Due to the intrinsic extent of the web, where it is difficult to measure the quantity of all web pages and the fact users only browse within the first 20 results; precision is widely used as a performance measurement. In addition, another factor

to consider is that web search engines are different and, therefore, they perform differently [86]; some may perform better than others for different queries and metrics.

There are different parameters to assess the performance of recommender systems:

- Precision evaluates the applicability of the first N results of the rating result set in relation to a query:

$$\text{P@N} = \frac{\text{Number of relevant documents in top N results}}{\text{N}} \tag{15}$$

- Recall represents the applicability of the first N results of the rating result set in relation to the complete result set:

$$\text{R@N} = \frac{\text{Number of relevant documents in top N results}}{\text{S}} \tag{16}$$

where S is the result set.

- The mean absolute error (MAE) evaluates the error between user predictions and user ratings:

$$\text{MAE} = \frac{\sum_{d=1}^{N} |P_{ui} - R_{ui}|}{N} \tag{17}$$

- The root squared mean error (RSME) measures the differences between predictions and user ratings:

$$\text{RME} = \frac{\sqrt{\sum_{d=1}^{N} (P_{ui} - R_{ui})^2}}{N} \tag{18}$$

where P_{ui} is the forecasted ranking for u on product i, R_{ui} is the current rank for u on product I, and N is the entire amount of products ranked by the user u.

- F-score combines Precision and Recall in an evenly weighted metric:

$$F_{measure} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{19}$$

- The average precision calculates the mean figures of P@n over the number n of retrieved web pages:

$$\text{AP} = \frac{\sum_{n=1}^{N} \text{P@n} \times \text{rel(n)}}{\text{Number of relevant documents for this query}} \tag{20}$$

where N is the total amount of retrieved web pages in the result list, and rel(n) represents that the n-th web page is either applicable or not to the query with values 1 or 0.

- The mean average precision measures the balance of average precision within a collection of Q web searches:

$$\text{MAP} = \frac{\sum_{q=1}^{Q} \text{AP(q)}}{Q} \tag{21}$$

- The normalized discounted cumulative gain uses a rated relevance metric to measure the relevance of a result that uses its rank within the web page set:

$$\text{NDCG@N} = Z_N \sum_{j=1}^{N} \frac{2^{R(j)} - 1}{\log_2(1 + j)} \tag{22}$$

where R(j) is the rated applicability of the result at position j within the web page set, Z_N is the standardization parameter that assures the optimum NDDC@N matches to 1. For binary ranking R(j) is fixed to 1 when the result at position j is relevant to the web search, and is fixed to 0 when the web page is irrelevant.

- TREC average precision includes rank position on the performance evaluation; it is defined at a cutoff N:

$$\text{TSAP@N} = \frac{\sum_{i=1}^{N} r_i}{N} \left\{ \begin{array}{l} r_i = \frac{1}{i} \text{ if the } i_{th} \text{ ranked result is relevant} \\ r_i = 0 \text{ if the } i_{th} \text{ ranked result is not relevant or duplicate} \end{array} \right\} \quad (23)$$

- The reciprocal rank of a web search is defined as the multiplicative inverse applied to the rank of the first relevant result:

$$RR = \frac{1}{rank_i} \quad (24)$$

where $rank_i$ is the order of the first relevant result.

- The mean reciprocal rank corresponds to averaged value of the RR (reciprocal rank) over a collection of Q searches:

$$MRR = \frac{\sum_{q=1}^{Q} RR(q)}{Q} \quad (25)$$

where RR(q) is the reciprocal rank associated to the search q and Q is the total collection of searches.

- The expected reciprocal rank is based on a cascaded model which penalizes documents which are shown below very relevant documents:

$$ERR = \sum_{r=1}^{N} \frac{1}{r} \prod_{i=1}^{r-1} (1 - R_i) \times R_r \quad (26)$$

where N is the amount of web pages in the ranking, r is the position at which the web user stops its search and R measures the probability of a click which can be interpreted as the relevance of the snippet.

Different research analyses have studied the efficiency of web search engines [87] and the level of overlap among the web pages presented by the search engines. Two new measurements to assess web search engine performance are proposed [88]: the relevance of result rating by a web search engine is assessed by the association among web and human rating and the capability to retrieve applicable web pages as the percentage of first ranked within the retrieved web pages. Four relevance metrics are applied to calculate the quality of web search engines [89]; three of them are adapted from the information retrieval: vector space method, Okapi likeness value, and coverage concentration rating and a new method which is developed from the human interaction to take human expectations into account based on a raw score and a similarity score.

The Text REtrieval Conference (TREC) [78] is supported by the National Institute of Standards and Technology (NIST) and the U.S. Department of Defense to sponsor studies in information and data retrieval research. TREC provides the framework for extensive evaluation applied to different text retrieval methods. Effectiveness of TREC algorithms against web search engines has been compared [90] where tested algorithms were developed with Okapi or Smart based on the term weighting methods (TF-IDF). Web search engines are evaluated against a series of relevance metrics [91,92]: precision, TREC-based mean reciprocal rank (MRR), and the TREC style average precision (TSAP) obtained from double relevance human evaluations for the top 20 web results displayed.

Web search evaluation methods can be assigned into eight different categories [93]: assessment founded on relevance, assessment founded on ranking, assessment founded on user satisfaction, assessment founded on size and coverage of the web, assessment founded on dynamics of search results, assessment founded on few relevant and known items, assessment based on specific topic, or domain and automatic assessment.

Different automatic web search engine evaluation methods have been proposed [94,95] to reduce human intervention. The algorithm first submits a query to the different web search engines where

the top 200 web pages of every web search engine are retrieved; then it ranks the pages in relation to their likeness to the web user information requirements; the model is built on the vector space method for query-document matching and ranking using TF-IDF. After ranking; the method lists as relevant the web pages that are in the first 20 documents retrieved by each web search engine with the top 50 ranked results. An automatic C++ application [96] compares and analyzes web result sets of several web search engines; the research analyses the URL coverage and the URL rank where recall and accuracy are used as relevance metrics however there is not description regarding the method that decides URL relevance. A technique for evaluating web search engines robotically [97] is founded in the way they order already-rated web search results where a search query is transmitted to different web search engines and the rank of retrieved web results is measured against the document that has already been paired with that query.

Web search engine evaluation has been examined from a webometric perspective [98] using estimates of success values, amount of provided URL, amount of domains retrieved, and the number of sites returned where evaluations are made to assess the retrieval of the most precise and comprehensive web results from every single web engine.

3.3. Learning to Rank

Learning to rank is defined as the implementation of semi-supervised or supervised computer learning techniques to generate a ranking model by combining different documents' features obtained from training data in an information retrieval system. The training data contains sets of elements with a specified position represented by a numerical score, order, or a binary judgment for each element. The purpose of a ranking method is to order new result lists in order to produce rankings similar to the order in the training data. Query document sets are normally defined by quantitative vectors which components are denoted features, factors or ranking signals. The main disadvantages of the learning to rank approach are that the optimization of relevance is based on evaluation measurements without addressing the decision of which measures to use; in addition, learning to rank does not take into consideration that user relevance judgment changes over time.

The model of learning to rank consists of:

- a set Q of M queries, $Q = \{q_1, q_2, \ldots, q_M\}$
- a set D of N web pages per query q_M, $D = \{ d_{11}, d_{12}, \ldots, d_{NM}\}$
- a set X of M feature vectors per each query q_M and web page d_N pairs, $X = \{x_{11}, x_{12}, \ldots, x_{NM}\}$
- a set Y of M relevance decisions per query q_M and web page d_N pairs, $Y = \{y_{11}, y_{12}, \ldots, y_{NM}\}$

The set of query-web page pairs $\{q_M, d_{NM}\}$ has associated a set of feature vectors $\{x_{NM}\}$ that represents specific ranking parameters describing the match between them. The set of relevance judgments $\{y_{NM}\}$ can be a binary decision, order, or score. Each feature x_{NM} and the corresponding score y_{NM} form an instance. The input to the learning algorithm is the set feature vectors $\{x_{NM}\}$ that represents the pair $\{q_M, d_{NM}\}$ and the desired output corresponds to set of relevance judgments $\{y_{NM}\}$. The ranking function f(x) is acquired by the learning method during the training process on which relevance assessments are optimized with the performance measurement or cost function (Figure 6).

The RankSVM learning to rank algorithm learns retrieval functions using clickthrough data for training based on a support vector machine (SVM) approach [99]. It uses a mapping function to correlate a search query with the features of each of the possible results where each data pair is projected into a feature space combined with the corresponding click-through data that measures relevance. Distinct ranking methods are developed for several classes of contexts [100] based on RankSVM that integrates the ranking principles into a new ranking model that encodes the context details as features. It learns a support vector machine model for a binary categorization on the selection between a couple of web pages where the evaluation is based on human conclusions and indirect user clicks.

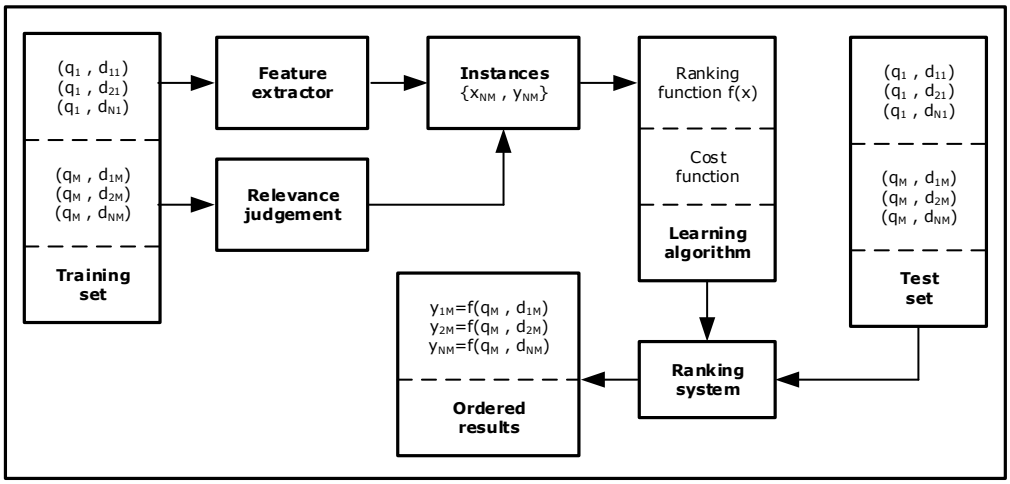

Figure 6. Learning to rank model.

A machine learned ranking model [101] routinely perceives and reacts to recent related queries where recent ranking takes into account freshness and relevance; the system is divided into two approaches: a high-accuracy recent related query detector and a specific recent related sorter trained to represent the characteristics and data distribution applicable for recent ordering.

Random forests is a low computational cost alternative to point-wise ranking approach algorithms [102] founded on the machine learning method "gradient boosted regression trees". The learning algorithm of random forests is applied multiple times to different subsets and the results are averaged whereas gradient boosted regression trees sequentially adds small trees at each iteration instead of training many full trees. The combination of both two algorithms first learns a ranking function with random forests and later uses it as initialization for Gradient boosted regression trees.

3.4. User Experience Feedback

User feedback can be explicit by directly asking the user to evaluate the results or implicit by analyzing the user behavior. Relevance feedback is normally applied for query expansion during the short term modelling of a user's instantaneous information need and for user profiling during the long term modelling of a user's persistent interests and preferences [103]. Web search user experience can be increased by the collection of implicit user behavior measures as signs of user interest and satisfaction rather than forcing the user to submit explicit feedback, which can be costly in time and resources while altering the pattern and web search user experience [104]. Explicit and implicit feedback present diverse properties of users' preferences with advantages and disadvantages; their combination in a user preference model provides a number of challenges however this can also overcome their associated issues by complementing each other, with similar performances despite their different characteristics. Implicit feedback can act as a substitute for explicit feedback as interchangeable sources of relevance information [105] when compared in terms of user perception and search effectiveness in some defined simulations rather than general web search scenarios. Three types of implicit feedback with two strategies for using implicit feedback to make recommendations based on rating estimation and predicted observations are proposed [106]: *examination* captures temporary interactions that begin and end during a single session, *retention* category groups user behaviors that suggest an intention for future use of the material, and *reference* includes user behaviors that create explicit or explicit links between information objects.

There is a connection between implicit measures of user activity and user's explicit satisfaction ratings [107] where the best model is the combination of clickthrough, time spent on the search result page, and how a user exited a result or ended a search session. In addition, behavioral patterns can also be used to predict user satisfaction for search sessions by analyzing genes. Clicks on web results shall not be taken as absolute implicit feedback rather than informative but biased as users' clicking

decisions are not only influenced by the relevance of the results, but also by the trust they have in the retrieval function, and by the overall quality of the result set [108]. Web search users frequently make a sequence of queries with a similar information need where the use of query chains obtained from clickthrough data enables preference judgments from search engine logs [109].

4. Recommender Systems

Recommender systems as Internet user semantic interfaces predict users' interest of different items or products providing a set of suggested items applying tailored relevance algorithms. Recommender systems consist on a database where user ratings and item description are stored and updated iteratively. A user interface interacts with the user where a profiler extracts user properties with explicit and implicit feedback; different suggestions and their order are computed by the recommender ranking algorithm (Figure 7). Due to their filtering properties they are widely used within e-commerce [110] as they also support e-commerce customers to find rare products they might not have discovered by themselves.

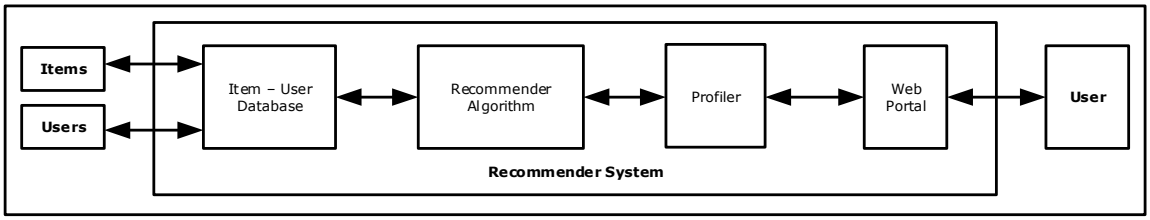

Figure 7. Recommender system architecture.

The model of a Recommender system (Figure 8) consists of:

- a set Q of N users, $Q = \{u_1, u_2, \ldots, u_N\}$
- a set I of M items, $I = \{i_1, i_2, \ldots, i_M\}$
- a rating matrix R, $R = [r_{ij}]$ where $i \in Q$ and $j \in I$
- a set X of N feature vectors, $X = \{x_1, x_2, \ldots, x_N\}$
- a weight matrix W, $W = [w_{ij}]$ where $i \in N$ and $j \in N + M$

The set of user-items $\{u_N, i_M\}$ has an associated set of feature vector $\{x_N\}$ that represents users and the different items assigned to them in the content model. The relevance judgment pred(u,i) can be a binary decision, order, or score based on the similarity between users or items in the collaborative model and weights in the content model.

4.1. Recommender System Classification

There are two main different categories of recommender systems [111]. Content-based recommender systems are built on a description of the product and a profile of the customer's wishes where different properties of the items and users are used to identify products with similar features without including other user's ratings. They suffer from some disadvantages [112], such as its inability to recommend completely different items that the user may also consider relevant and a customer needs to mark several products before getting relevant recommendations. Collaborative recommender systems are founded on the customer's previous marks to other products and the consideration of other decisions made by similar users; they made suggestions based on a high correlation between users or items. Although collaborative recommendation reduces the issues from the content-based solution; it has other drawbacks such as it needs a large number of rating data to calculate accurate correlations and predictions; it also ignores on its calculations new added users or items. Hybrid recommender systems take a combination of both approaches.

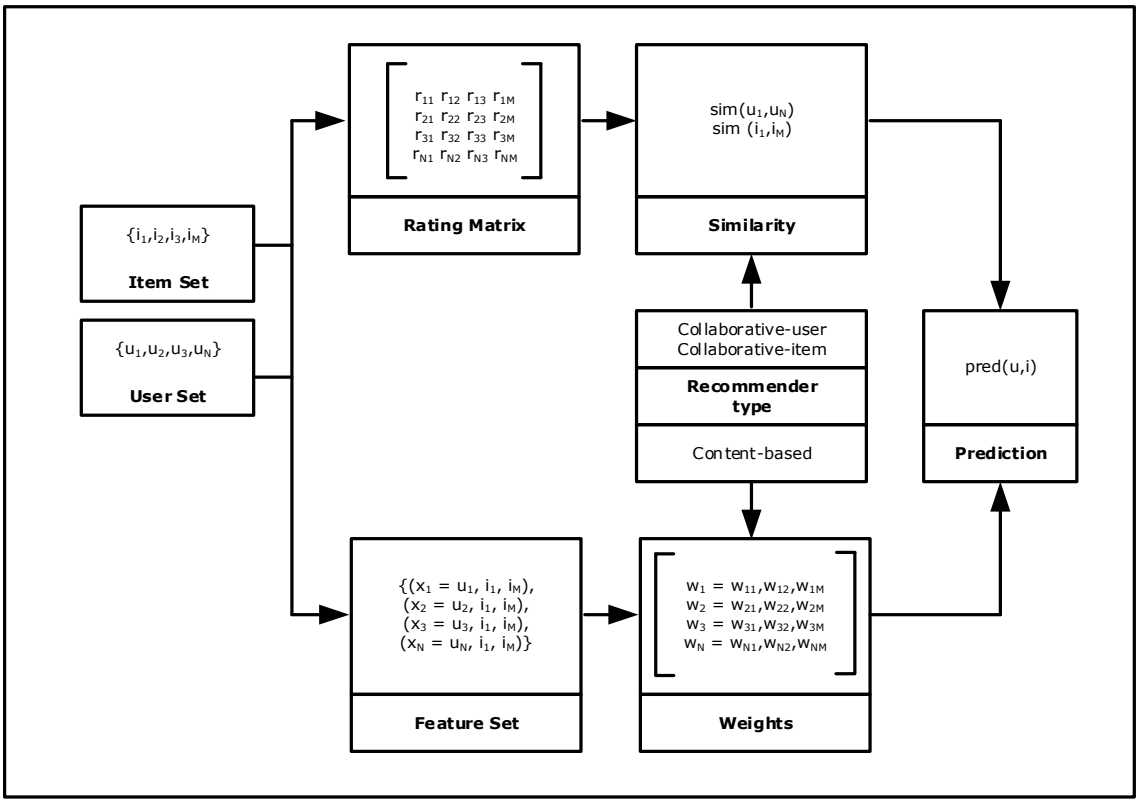

Figure 8. Recommender system model.

The user based recommender generally uses the Pearson's relationship similarity:

$$\text{similarity}(u,v) = \frac{\sum_{i \in M}(r_{u,i} - \overline{r_u}) \times (r_{v,i} - \overline{r_v})}{\sqrt{\sum_{i \in M}(r_{u,i} - \overline{r_u})^2} \times \sqrt{\sum_{i \in M}(r_{v,i} - \overline{r_v})^2}} \tag{27}$$

which the prediction on the user-based is defined below:

$$\text{prediction}\ (u,i) = \overline{r_u} + \frac{\sum_{v \in N(u)} \text{similarity}(u,v) \times (r_{v,i} - \overline{r_v})}{\sum_{v \in N(u)} |\text{similarity}(u,v)|} \tag{28}$$

where N(u) defines a set of neighbors of u, similarity(u,v) represents the correlation between user u and v, ru,i is the mark of customer u to product i, is the average rating of customer u and M is the total quantity of items.

The item based recommender systems generally use the cosine similarity:

$$\text{similarity}(u,v) = \frac{\sum_{d=1}^{N} i_d \times p_d}{\sqrt{\sum_{d=1}^{N} i_d^2} \times \sqrt{\sum_{d=1}^{N} p_d^2}} \tag{29}$$

although the Euclidean distance similarity can also be used:

$$\text{similarity}(i,p) = \sqrt{\sum_{d=1}^{N} (i_d - p_d)^2} \tag{30}$$

and the Manhattan distance similarity:

$$\text{similarity}(i,p) = \sum_{d=1}^{N} |i_d - p_d| \tag{31}$$

where the item-based prediction is defined below:

$$\text{prediction}\ (u,i) = \frac{\sum_{q \in N(i)} \text{similarity}(i,q) \times r_{u,q}}{\sum_{q \in N(i)} \text{similarity}(i,q)} \tag{32}$$

where N(i) is a collection of neighbors of i and similarity(i,p) is the correlation between item i and p is the relevance of item i to user d, and N is the overall quantity of users.

The linear prediction on the content-based is defined below:

$$\text{prediction}(u,i) = \langle w, x \rangle = \sum_{q=1}^{N+M} w_q x_q \tag{33}$$

where x are the instance vectors and w the matrix weights.

In multi-criteria ranking recommenders, users give ratings on several characteristics of an item as a vector of ratings [113] whereas cross domain recommender systems suggest items from multiple sources with item or user-based formulas founded on locality collaborative filtering [114]; they operate by first modelling the traditional likeness association as a directly connected graph and then exploring the entire potential paths that links users or items to discover new cross-domain associations.

There are several relevance metrics for assessment of recommender systems in various e-commerce business frameworks [115]. Accuracy evaluation metrics are allocated into three major categories: predictive, based on the error between estimated and true user ratings; classification, based on the successful decision making; and rank, based on the correct order.

Social network information is inserted to recommender systems [116] as an additional input to improve accuracy; collaborative filtering-based social recommender systems are classified into two categories: the matrix factorization approach, where user-to-user social information is integrated with user item feedback history, and neighborhood-based social approaches based on social network graphs.

4.2. Personalized Information and Recommender Systems

Personal recommender systems are one of the most efficient tools to solve the information overload problem, such as shopping in the Internet. The common problem faced by the different recommender systems is that the user and product information are dynamically changing [117]; in order to overcome this issue, an improved dynamic algorithm based on local information updating is proposed to avoid the computational expensive process to update static information when one new piece of information is produced by the user or products. A personalized recommendation methodology to obtain further effectiveness and quality when applied to e-commerce is based on a variety of data mining techniques, such as web usage mining learned from a clickstream, a decision tree induction that minimizes recommendation errors by making recommendation only for customers who are likely to buy recommended products, product taxonomy and, finally, association rule mining that chooses highly business-efficient products among the candidate recommendable products [118].

Theories related to the use of personalized content services and their effect on user satisfaction can be identified as [119]: *information overload theory*, which implies that user satisfaction increases when the recommended content fits user interests, *uses and gratifications theory* focuses on motivations for information access, and *user involvement theory* covers the user's preferred content recommended by a process in which the user has explicit involvement. The Internet of Things (IoT) Big Data sharing enable personal recommender systems to model their user relevance from external sources [120];

data from a variety of social web services, such as Facebook and LinkedIn, can improve personal recommender system quality as it contains large amounts of personal information about users.

Personal recommender systems can also be applied in traditional transport information systems [121]. In the current applications, users must explicitly provide information related to both their profiles and travels in order to receive a personalized response at additional extra effort from the user in terms of search time. The proposed model identifies implicit users' information and predicts their needs even if some data is missing by the use of an ontology that more accurately models semantic data. The contradiction between the expansion of tourism information and the difficulty of tourists obtaining tourism information enables the tourism information recommender system to have a business model [122]; the framework of a recommender system that combines key algorithms and existing tourism information recommendation websites allows customers to broaden their experience of the tourism information service and make tourism decisions more accurately and promptly.

A user model web recommender system for recommending, predicting and personalizing music playlists is based on a hybrid similarity matching method that combines collaborative filtering with ontology-based semantic distance measurements [123]; a personalized music playlist, from a selection of recommended playlists that comprises the most relevant tracks to the user is dynamically generated. A hybrid travel recommender system makes use of the individual demographic information to suggest what suits the user thereby making it more intelligent and user acceptable [124]. The model combines the features of content-based travel with collaborative and demographic filtering techniques.

5. Neural Networks

Artificial neural networks are representations of the brain, the principal component of the central nervous system. They are usually presented as artificial nodes or "neurons" in different layers connected together via synapses to create a network that emulates a biological neural network. The synapses have values called weights which are updated during network calculations.

Neurons are defined by:

- the activation, $a_j(t)$, that represents the neuron j state;
- the activation threshold Φ_j;
- the activation function f_a that computes the new activation; and
- the output function f_{out} that calculates the output from the activation.

The neuron model is represented mathematically as:

$$a_j(t+1) = f_a\left(a_j(t), p_j(t), \theta_j\right) \tag{34}$$

where $p_j(t)$ represents the combined input to neuron j:

$$o_j(t) = f_{out}\left(a_j(t)\right) \tag{35}$$

where $o_j(t)$ is the new output from the activation $a_j(t)$.

Artificial neural networks are normally described by three variables:

- the links among independent layers of neurons, w;
- the learning method for adapting the weights of the connections; and
- the propagation function f(t) that calculates the input $i_j(t)$ to the neuron j from the outputs $o_j(t)$ of the predecessor neurons:

$$f(t) = i_j(t) = \sum_{i=1}^{n} o_i(t) w_{ij} \tag{36}$$

where w_{ij} is the network weight matrix.

There are two main models to represent a neural network: the feed-forward model, where connections between neurons follow only the forward direction, and the recurrent model, where connections between neurons form a direct cycle (Figure 9).

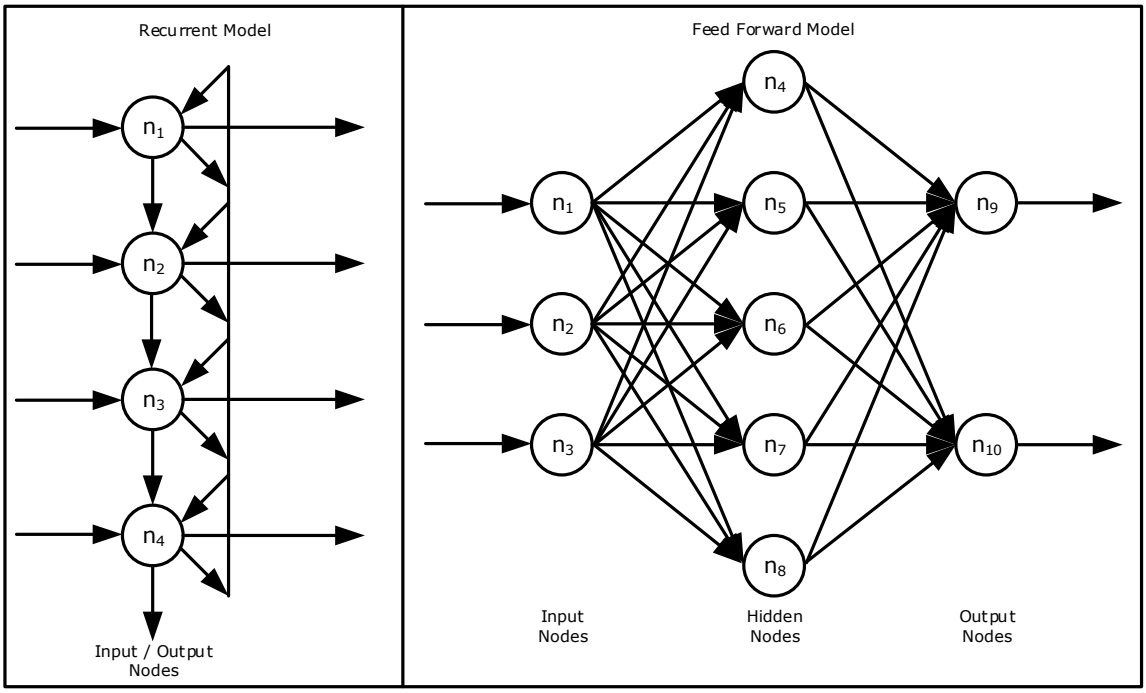

Figure 9. Neural networks.

Artificial intelligence is also applied to web search and recommender systems to increase search user experience and provide relevant results. AI is used for quality control to separate relevant results from spam and ranking algorithms in learning to rank. natural language processing (NLP), and image analysis are developed to understand user relevance and search queries. Future applications of AI in e-commerce include the understanding of human intent rather than keywords only, predicting the user's next search or need, understanding the buyer cycle, target and personalize customers or users with products or information, respectively, and, finally, real-time customer analytics of the increasing Big Data.

RankBrain is an artificial intelligence algorithm learning method used by Google that focuses on what the user means, rather than specific keywords or topics. RankBrain divides search queries into word vectors or distributed representations with linguistic or semantic similarity; it predicts words or phrases that may have a similar meaning and filter the results accordingly [125].

5.1. Neural Networks in Web Search

The capability of a neural network to sense and learn recursively from several input figures to obtain the preferred output values has also been applied in the World Wide Web as a user interest adjustment method to provide relevant answers. Neural networks have modelled web graphs to compute page ranking; a graph neural network [126] consists on nodes with labels to include features or properties of the web page and edges to represent their relationships; a vector called state is associated to each node, and is modelled using a feed-forward neural network that represents the reliance of a node with its neighborhood. Another graph model [127] assigns every node in the neural network to a web page where the synapses that connect neurons denotes the links that connect web pages. The web page material rank approximation uses the web page heading, text material, and hyperlinks. The link score approximation applies the similarity between the quantity of phrases in

the anchor text that are relevant pages. Both web page material rank approximation and web link rank approximation are included within the connection weights between the neurons.

An unsupervised neural network is used in a web search engine [128] where the k-means method is applied to cluster n web results retrieved from one or more web search engines into k groups where k is automatically estimated. Once the results have been retrieved and feature vectors are extracted, the k means grouping algorithm calculates the clusters by training the unsupervised neural network. In addition, a neural network method [129] classifies the relevance and reorders the web search results provided by a metasearch engine. The neural network is a three layer feed-forward model where the input vector represents a keyword table created by extracting the title and snippets of words from all the results retrieved and the output layer consists of a unique node with a value of 1 if the web page is relevant and 0 if irrelevant.

A neural network ranks pages using the HTML properties of the web documents [71] where words in the title have a stronger weight than in the body specific. Then, it propagates the reward back through the hypertext graph reducing it at each step. A back-propagation neural network [130] is applied to web search engine optimization and personalization where its input nodes are assigned to an explicit measured web user profile and a single output node represents the likelihood the web user may regard web page as relevant.

An agent learning method is applied to web information retrieval [131] where every agent uses different web search engines and learns their suitability based on user's relevance response; a back-propagation neural network is applied where the input and the output neurons are configured to characterize any training term vector set and relevance feedback for a given query.

5.2. Neural Networks in Learning to Rank Algorithms

Although there are numerous learning to rank methods we only analyze the ones based on neural networks. Learning to rank algorithms are categorized within three different methods according to their input and output representations:

The pointwise method considers every query web page couple within a training set is assigned to a quantitative or ordinal value. It assumes an individual document as its only learning input. Pointwise is represented as a regression model where, provided a unique query document couple, it predicts its rating.

The pairwise method evaluates only the relative order between a pair of web pages; it collects documents pairs from the training data on which a label that represents the respective order of the two documents is assigned to each document pair. Pairwise is approximated by a classification problem. Algorithms take document pairs as instances against a query where the optimization target is the identification of the best document pair preferences. RankNet [132] is a pairwise model based on a neural network structure and gradient descent as the method to optimize the probabilistic ranking cost function; a set of sample pairs together with target likelihoods that one result is to be ranked higher than the other is given to the learning algorithm. RankNet is used to combine different static web page attributes, such as web page content, domain or outlinks, anchor text or inlinks, and popularity [133]; it outperforms PageRank by selecting attributes that are detached from the link fabric of the web where accuracy can be increased by using the regularity with which web pages are visited. RankNet adjusts the attribute weights to best meet pairwise user choices [134] where the implicit feedback such as clickthrough and other user interactions is treated as vector of features which is later integrated directly into the ranking algorithm. SortNet is a pairwise learning method [135] with its associated priority function provided by a multi-layered neural network with a feed-forward configuration and is is trained in the learning phase with a dataset formed of pairs of documents where the associated score of the preference function is provided. SortNet is based on the minimization of the square error function between the network outputs and preferred targets on every unique couple of documents.

The listwise method takes ranked web page lists as instances to train ranking models by minimizing a cost function defined on a predicted list and a ground truth list; the objective of learning

is to provide the best ranked list. Listwise directly learns document lists by treating ranked lists as learning instances instead of reducing ranking to regression or classification. ListNet is a listwise cost function [136] that represents the dissimilarity between the rating list output generated by a ranking model and the rating list given as the master reference. ListNet maps the relevance tag of a query web page set to a factual value with the aim to define the distribution based on the master reference. ListNet is built on a neural network with an associated gradient descent learning algorithm with a probability model that calculates the cost function of the Listwise approach; it transforms the ratings of the allocated documents into probability distributions by applying a rating function and implicit or explicit human assessments of the documents.

5.3. Neural Networks in Recommender Systems

Neural networks have been also applied in recommender systems as a method to sense and predict user ratings to different items or to cluster users or items into different categories.

An adaptive resonance theory (ART) is an unsupervised learning method based on a neural network, comprised of a comparative and an identification layer both formed of neurons. In addition, a recognition threshold and a reset unit are included to the method. A recommender system based on a collaborative filtering application using the k-separability method [137] is built for every user on various stages: a collection of users is clustered into diverse categories based on their likeness applying adaptive resonance theory, then the singular value decomposition matrix is calculated using the k separability method based on a neural network with a feed-forward configuration where the n input layer corresponds to the user ratings' matrix and the single m output the user model with $k = 2m + 1$. An ART model is used to cluster users into diverse categories [138] where a vector that represents the user's attributes corresponds to the input neurons is and the applicable category to the output ones.

A self-organizing map (SOM) artificial neural network presents a reduced dimensional quantified model of the input space; the SOM is trained with unsupervised learning. A recommender application that joins the self-organizing map with collaborative sorting [139] applies the division of customers by demographic features in which customers that correspond to every division are grouped following their item selection; the collaborative filtering method is used on the group assigned to the user to recommend items. The SOM learns the item selection in every division where the input is the customer division and the output is the cluster type. A SOM calculates the ratings between users [140] to complete a sparse scoring matrix by forecasting the rates of the unrated items where the SOM is used to identify the rating cluster.

There are different frameworks that combine collaborative sorting with neural networks. Implicit patterns among user profiles and relevant items are identified by a neural network [141]; those patterns are used to improve collaborative filtering to personalize suggestions; the neural network algorithm is a multiplayer feed-forward model and it is trained on each user ratings vector. The neural network output corresponds to a pseudo user ratings vector that fills the unrated items to avoid the sparsity issue on recommender systems. A study of the probable similarities among the scholars' historic registers and final grades is based on an intelligent recommender system structure [142] where a multi layered neural network with a feed-forward configuration is applied with supervised learning.

Any machine learning algorithm that includes a neural network with a feed-forward configuration with n input nodes, two hidden nodes, and a single output node learning process [143] can be applied to represent collaborative filtering tasks; the presented algorithm is founded on the reduction of the dimensionality reduction applying the singular value decomposition (SVD) of a preliminary user ranking matrix that excludes the necessity for customers to rank shared items with the aim of becoming forecasters for another customer preferences. The neural network is trained with an n singular vector and the average user rating; the output neuron represents the predicted user rating.

Neural networks have been also implemented in film recommendation systems. A neural network in a feed-forward configuration with a single hidden layer is used as an organizer application that predicts if a certain program is relevant to a customer using its specification, contextual information,

and given evaluation [144]. A TV program is represented by a 24-dimensional attribute vector, however, the neural network has five input nodes: three for the transformed genre, one for the type of day, and the last one for the time of day, a single hidden neuron, and two output neurons: one for like and the other for dislike. A neural network identifies which household follower provided a precise ranking to a movie at an exact time [145]. The input layer is formed of 68 neurons which correspond to different user and time features and the output layer consists of 3 neurons which represent the different classifiers. An "interior desire system" approach [146] considers that if customers may have equivalent interest for specific products if they have close browsing patterns; the neural network classifies users with similar navigation patterns into groups with similar intention behavioral patterns based on a neural network with a back-propagation configuration and supervised learning.

5.4. Deep Learning

Deep learning applies neural and sensor networks with various computing layers that perform several linear and nonlinear transformations to model general concepts in data. Deep learning is under a branch of computer learning that models representations of information (Figure 10). Deep learning is characterized as using a cascade of l-layers of nonlinear computing modules for attribute identification and conversion; each input of every sequential layer is based on the output from the preceding layer. Deep learning learns several layers of models that correlate to several levels of conceptualization; those levels generate a scale of notions where the higher the level, the more abstract concepts are learned.

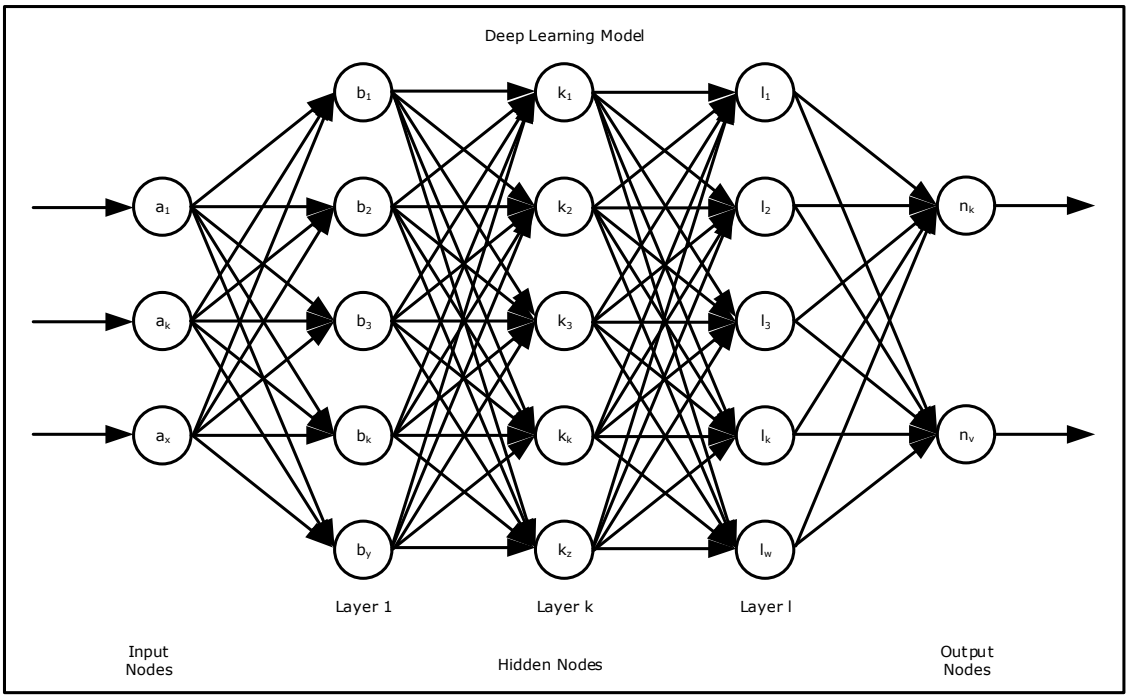

Figure 10. Deep learning.

5.4.1. Deep Learning in Ranking

Deep learning models have been used in learning to rank to rate brief text pairs which main components are phrases [147]. The method is built using a convolutional neural network structure where the best characterization of text pair sets and a similarity function is learned with a supervised algorithm. The input is a sentence matrix with a convolutional feature map layer to extract patterns, a pooling layer is then added to aggregate the different features and reduce the representation. An attention-based deep learning neural network [148] focuses on different aspects of the input data to include distinct features; the method incorporates different word order with variable weights changing

over the time for the queries and search results where a multilayered neural network ranks results and provides a listwise learning to rank using a decoder mechanism.

A deep learning architecture to relevance ranking in information retrieval (IR) simulates the human judgment process [149]; firstly, a detection strategy is designed to extract relevant contexts, then, a measure network is applied to determine local relevance by utilizing a convolutional neural network (CNN) and, finally, an aggregation network with sequential integration and a term gathering mechanism is used to produce a global relevance score. A recurrent neural network is based on the concept of pseudo relevance feedback where top ranked documents can provide important information about the query's characteristics [150]. The method uses the inherent feature distributions of the top results to learn a deep listwise context model that helps fine-tune the initial ranked list. Specifically, the recurrent neural network sequentially encodes the top results using their feature vectors, learns a local context model, and re-ranks the top results.

5.4.2. Deep Learning in Web Search and Information Retrieval

Deep stacking networks [151] are used for information retrieval with parallel and scalable learning; the design philosophy is based on basic modules of classifiers are first designed and then they are combined together to learn complex functions. The output of each deep stacking network is linear, whereas the hidden unit's output is sigmoidal nonlinear.

The key success of deep learning is its ability to accurately learn distributed representations; vector representations, or structured arrangements of them. Deep learning techniques, recurrent neural networks, and convolutional neural networks, applied in information retrieval and web search also covers natural language processing [152], such as word embedding, including matching, translation, classification, structured prediction, search, question answering, and image retrieval.

Latent semantic models based on a convolutional neural network to learn low-dimensional semantic vectors for search queries and web documents [153] apply convolution-max pooling operations where local contextual information at the word n-gram level is modelled first and salient local features in a word sequence are combined to form a global feature vector. Finally, the high-level semantic information of the word sequence is extracted to form a global vector representation. Deep semantic matching models for search and neural collaborative filtering models for recommendation are becoming the state-of-the-art technologies [154] where the key to the success of the deep learning approach is its strong ability in learning of representations and generalization of matching patterns from raw data, such as queries, documents, users, and items, particularly in their raw forms.

5.4.3. Deep Learning in Recommender Systems

Deep learning is also used in recommender systems. A deep feature representation [155] learns the content information and captures the likeness and implicit association among customers and items where collaborative filtering is used in a Bayesian probabilistic framework for the rating matrix. A deep learning approach [156] assigns items and customers to a vector space model in which the similarity between customers and their favored products is optimized; the model is extended to jointly learn features of items from different domains and user features according to their web browsing history and search queries. The deep learning neural network maps two different high dimensional sparse features into low dimensional dense features within a joint semantic space.

Deep neural networks can better generalize hidden feature combinations through low-dimensional dense embeddings learned from sparse features. However, this solution can over-generalize and recommend less relevant items when the user-item interactions are sparse [157]; a proposed framework combines the strengths of wide linear learning memorizing the sparse feature interactions using cross-product feature and deep learning that generalizes previously unseen feature interactions. A model that combines a collaborative filtering recommendation algorithm with deep learning [158] uses a feature representation method based on a quadric polynomial regression

model which obtains the latent features more accurately by improving upon the traditional matrix factorization algorithm and the deep neural network model to predict the rating scores.

Automatic music recommendation has become an increasingly relevant issue in recent years, since most music is now sold and consumed digitally. The use of a latent factor model for recommendation and the prediction of the latent factors from music audio when they cannot be obtained from usage data is proposed based on deep convolutional neural networks [159] despite the fact that there is a large semantic gap between the characteristics of a song that affect the user preference and the corresponding audio signal. Existing content-based music recommendation systems typically extract traditional audio content features, such as Mel-frequency cepstral coefficients, and then predict user preferences [160]. However, these traditional features were not originally created for music recommendation; a model based on a deep belief network and probabilistic graphical model automates the process that simultaneously learns features from audio content and makes personalized recommendations.

5.5. The Random Neural Network

The random neural network is founded on the G-networks or Gelenbe network [161,162]. G-networks are a model for neural networks as well as queueing methods with precise control operations, such as route management or traffic removal. The stationary distribution of G-networks has a product form solution to Jackson's theorem [163], however, the resolution of a set of non-linear equations for the traffic circulation [164] is required.

5.5.1. The Random Neural Network Model

The random neural network [165] is a spiked method with recurrent aleatory properties. The key analytical characteristics are the "product form" where a single network stable status solution exists. The random neural network model characterizes more accurately the transmission method of signals in numerous biological neural networks in which signals are transmitted as spikes instead of as fixed analogue impulses.

The random neural network consists on m-neurons. The status of the m neuron of the network at time p is characterized by a vector of positive integers k(p) = [k_1(p), ... k_m(p)] where k_m(p) is the potential value of node m at time p. Neurons interact with each other by interchanging impulses in the arrangement of a of single unit amplitude spike:

- A positive spike is interpreted as excitation signal because it increases by one unit the potential of the receiving neuron; and
- A negative spike is interpreted as inhibition signal decreasing by one unit the potential of the receiver node; when the potential is already zero, it does not produce any effect.

Every neuron amasses spikes and it will fire only if has a positive potential. Firing will occur at random and spikes will be transmitted at rate r(x) associated to individual, equally and exponentially dispersed spike time periods:

- Positive spikes will travel to neuron y with a likelihood p^+(x,y) as excitatory impulses; and
- Negative spikes will travel to neuron y with a likelihood p^-(x,y) as inhibitory impulses.

A node can transmit spikes outside of the network with a likelihood d(x). We have:

$$d(x) + \sum_{y=1}^{m} \left[p^+(x,y) + p^-(x,y) \right] = 1 for 1 \leq x \leq m \quad (37)$$

Neuron potential decreases by one unit when the neuron fires either an excitatory spike or an inhibitory spike. External positive or negative spikes to neuron x will arrive at rates Λ(x), λ(x), respectively, by stationary Poisson processes. The random neural network weight parameters w^+(y,x)

and $w^{-}(y,x)$ are the positive rate of positive and negative spike emission respectively from neuron x to neuron y:

- $w^{+}(y,x) = r(x)p + (x,y) \geq 0$;
- $w^{-}(y,x) = r(x)p - (x,y) \geq 0$.

This model communicates information by transmission of spikes at certain frequency or rate. Each neuron x, when it is excited, behaves as a frequency modulator emitting spikes at rate $w(x,y) = w^{+}(x,y) + w^{-}(x,y)$ to neuron y. Spikes are transmitted at exponentially dispersed aleatory time periods. Neurons act as demodulators of non-linear frequencies that transform the arriving positive and negative spikes into potential (Figure 11).

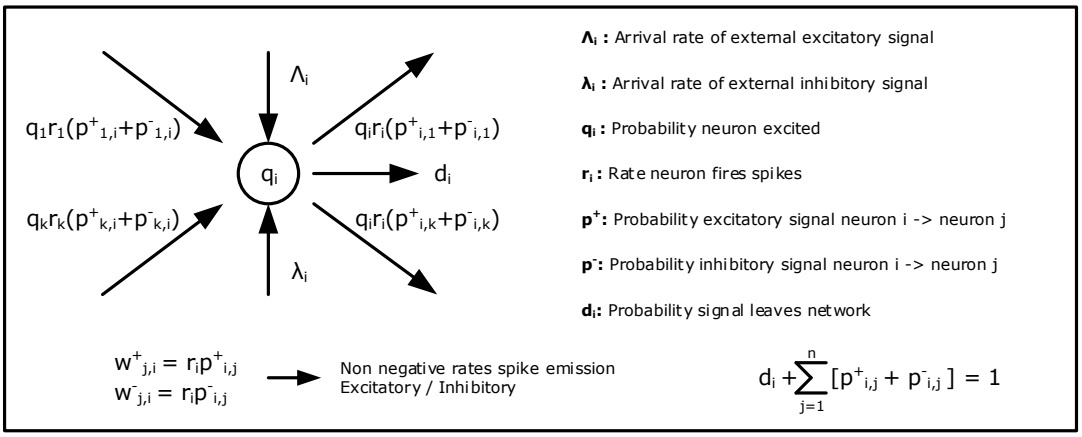

Figure 11. Random neural network model.

The random neural network method has a solution with a "product form"; the network's stable probability distribution may be represented as the multiplication of the partial probability of every node status [166].

The likelihood distribution of the random neural network state is designated as $p(k,p) = Prob[k(p) = k)]$ and the partial likelihood a neuron x is exited at time p as $q_x(p) = Prob[k_x(p) > 0]$. The stable likelihood distribution $p(k) = \lim_{p\to\infty} p(k,p)$ and $q_x = \lim_{p\to\infty} q_x(t)$ where k(p) is a continual time Markov model that solves Chapman-Kolmogorov equations.

q_x is defined:

$$q_x = \frac{\lambda^{+}(x)}{r(x) + \lambda^{-}(x)} r(x) = \sum_{y=1}^{m} \left[w^{+}(x,y) + w^{-}(x,y) \right] for\ 1 \leq x \leq m \tag{38}$$

where the $\lambda^{+}(x)$, $\lambda^{-}(x)$ for x=1, . . . ,m meets the set of non-linear concurrent equations:

$$\lambda^{+}(x) = \sum_{y=1}^{m} \left[q_y r(y) p^{+}(y,x) \right] + \Lambda(x) \tag{39}$$

$$\lambda^{-}(x) = \sum_{y=1}^{m} \left[q_y r(y) p^{-}(y,x) \right] + \lambda(x) \tag{40}$$

If $\{\lambda^{+}(x),\lambda^{-}(x)\}$ can be solved with positive values to the above equations and meets $q_x < 1$ then:

$$p(k) = \prod_{x=1}^{m} [1 - q_x] q_x^{kx} \tag{41}$$

The network will be stable if a value $q_x < 1$ can be found. Neuron x has an average potential of $q_x/[1 - q_x]$ and the frequency of transmission of impulses from node x in stable status is $q_x r(x)$. If we

have $\lambda^+(x) > [r(x) + \lambda^-(x)]$ for a neuron; this denotes that the node is not stable, which means that it is continuously excited in stable status with a rate of positive and negative spike transmission r(x) to another neuron y will be $r(x)p^+(x,y)$ and $r(x)p^-(x,y)$, respectively.

The random neural network learning algorithm [167] is built on a quadratic error cost function with gradient descent. The resolution of o linear and o nonlinear equations is required to solve the back-propagation model every time the m random neuron network learning algorithm is presented with an additional input and output set.

5.5.2. Random Neural Network Extensions

The random neural network was extended to multiple categories of signals [168] where each different flow is a category of signals in the configuration of spikes. A spike which represents a signal category when leaving a node could be codified at the receiving node as a positive or negative spike within the identical or different category. Transmission rates of spikes from a category are equivalent to the amount of excitation of the inner status in that specific category at the transmitting neuron. Its learning algorithm [169] applies to both recurrent and feedforward models founded on the optimization of a cost function using gradient descent that involves the solution of a set of mC nonlinear and mC nonlinear equations with an $O([mC]^2)$ complexity for the feedforward configuration and $O([mC]^3)$ complexity for the recurrent one.

The bipolar RNN [170] method has positive as well as negative nodes and balanced performance of positive and negative spikes flowing through the network. Positive neurons collect and transmit only positive spikes, whereas negative neurons accumulate and transmit only negative spikes. Positive spikes cancel negative spikes at each negative neuron, and vice versa. Connections between different neurons can be positive or negative where a spike departing a neuron may transfer to a different neuron as a spike of alike or opposite potential. The bipolar RNN presents auto-associative memory capabilities and universal function approximation properties [171]. The feed-forward bipolar model [172] with r hidden layers (r + 2 in total) can uniformly estimate continuous functions of r parameters.

The RNN with synchronized interactions [173] includes the possibility of neurons acting together on other neurons, synchronized transmission among neurons, where a neuron may activate transmitting to another neuron and cascades of this activate transmissions can follow. The model defines the activation of transmission among a pair of neurons and activated transmission by sequences of neurons, as well as feedback rings in the sequences to produce extensive transmission bursts. Synchronous interactions between two neurons that jointly excite a third neuron [174] is enough to generate synchronous transmission by a great concentration of neurons. An m-neuron recurrent neural network that has both conventional positive and negative exchanges and synchronous exchanges has a learning algorithm complexity of $O(m^3)$ based on gradient descent.

Deep learning with the random neural network [175] is based on soma-to-soma interactions between natural neuronal cells. Clusters are formed of compact concentration of neurons where the transmission configuration of a neuron instantaneously stimulates and incites transmission by adjacent neurons through dense soma-to-soma interactions founded on the mathematical characteristics of the G-networks and the random neural network. The characteristics of deep learning clusters head to a transmission function that can be applied for wide sets of neurons.

5.5.3. The Random Neural Network in Web Search

An intelligent internet search assistant (ISA) [176] built on the random neural network measures customers' relevance and selects web snippets from several web search engines (Google, Yahoo, Ask Lycos, and Bing) applying the learned choices in an iterative method (Figure 12). ISA is an interface between an individual user's query and the different web search engines that uses a learning approach to iteratively determine the best set of results that best match the learned characteristics of the end user's queries. ISA acquires a query from the user and retrieves results from one or various search engines assigning one neuron per each web result dimension.

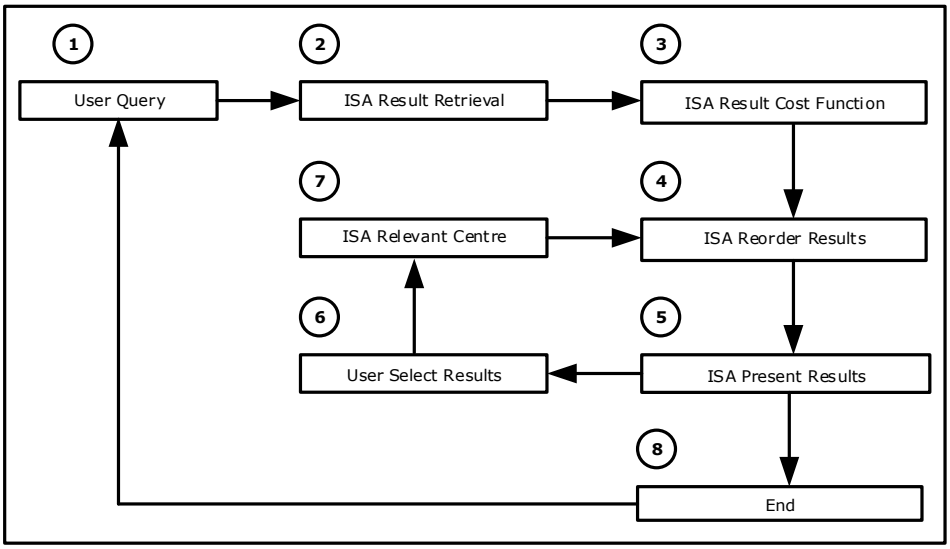

Figure 12. Intelligent search assistant user iteration.

The result relevance is calculated by applying an innovative cost function based on the division of a query into a multidimensional vector that weights its dimension terms with different relevance parameters [177]. User relevance is learned iteratively from user feedback, independently using either gradient descent to reorder the results to the minimum distance to the user Relevant center point or reinforcement learning that rewards the relevant dimensions and "punishes" the less relevant ones (Figure 13).

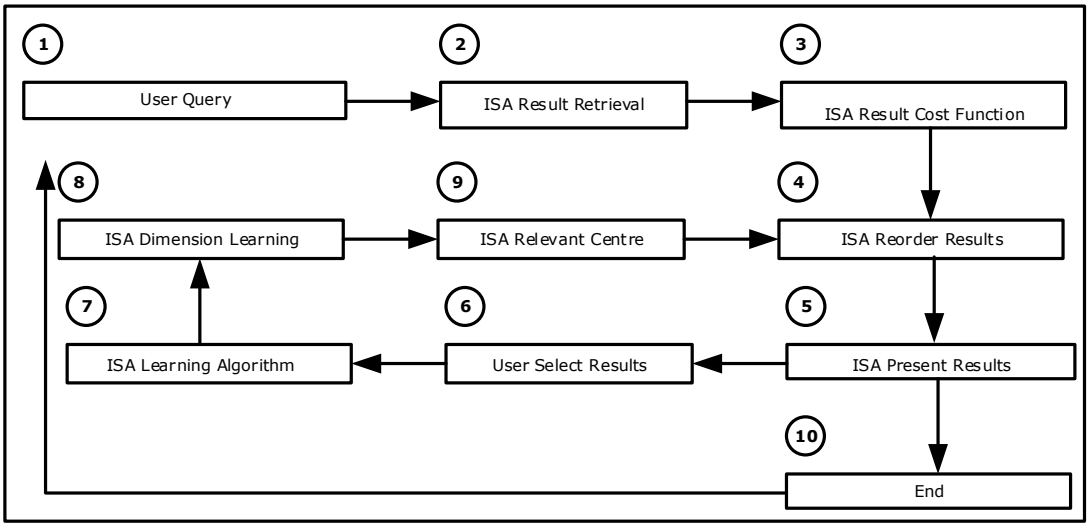

Figure 13. Intelligent search assistant learning algorithm model.

ISA behaves as a search assistant among users and Big Data [178]: recommender systems (GroupLens film, Trip Advisor, and Amazon), academic databases (Google Scholar, IEEE Xplore, CiteseerX, or Microsoft Academic), and metasearch engines (Ixquick and Metacrawler). ISA acquires a query from the user, retrieves results from one or more recommender systems, and reorders them according to an innovative cost function that measures results' relevance. ISA represents queries and search outcomes as vectors with logical or numerical entries, where the different terms are weighted with relevance parameters. ISA acts as a web search recommender system for general topic queries where the user explores the results provided rather than providing the better one of search results. ISA benefits the users by showing relevant set results on first positions rather than a best result surrounded of many other irrelevant results.

ISA emulates a brain learning structure [179] by using the random neural network with deep learning clusters where ISA measures and evaluates web result relevance connecting a dedicated deep learning cluster to a specific web search engine (Figure 14). ISA is designed with one neuron or cluster of neurons for each dimension of the data acquired from the web as a result of a query and it is used to determine the relevance from the end-user's perspective of the acquired result. The best performing cluster is assigned to the other web search engines in order to increase system speed, and improve result accuracy and relevance, while reducing learning time and network weight computation.

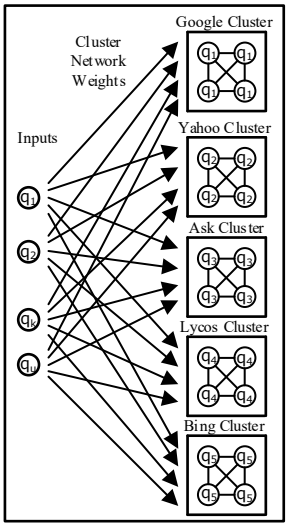

Figure 14. Intelligent search assistant learning deep learning clusters.

A deep learning management cluster [180] is incorporated into ISA to decide the final result relevance measured from the inputs of the different deep learning cluster assigned to each web search engine. The additional deep learning management cluster emulates the way the brain makes decisions by selecting results from the best performing deep learning clusters in an unsupervised learning algorithm (Figure 15).

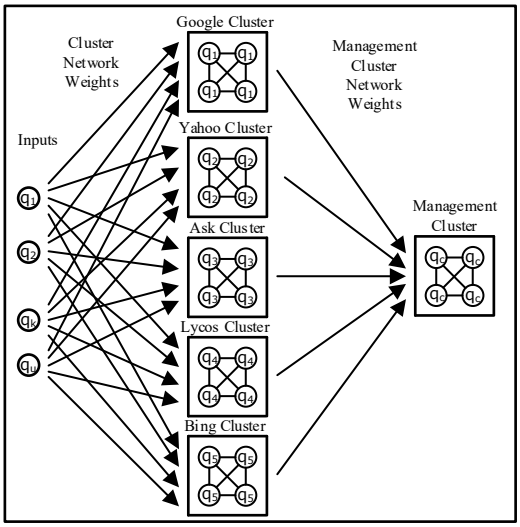

Figure 15. Intelligent search assistant learning deep learning clusters.

6. Conclusions

This paper has presented a detailed survey of web searching, including Internet assistants, web search engines, meta-search engines, web result clustering, travel services, and citation analysis

as Internet user semantic interfaces. In addition, Big Data ranking has been examined with ranking algorithms, relevance metrics, and learning to rank. Recommender systems as Internet user semantic applications have been defined and neural networks in web search, learning to rank, recommender systems, and deep learning have been analyzed. The random neural network has been presented with several applications to reasoning approaches for web search and knowledge extraction.

The digitalization of the real world generates Big Data with its associated challenges; in order for the additional data to be transformed into useful and relevant information, it requires indexation by search engines and filtering by recommender systems in order to discriminate value from noise. Future Big Data trends will apply additional deep learning techniques to find correlations between different variables or datasets that will be used by decision making management systems. As a result of the large volume of Big Data generated, only artificial intelligence will be able to understand, process, and create value for the additional produced information. Finally, Big Data platforms will be further integrated and standardized to enable a full, seamless integration between different Big Data generators and consumers.

Funding: This research received no external funding.

Conflicts of Interest: The author declares no conflict of interest.

References

1. Jansen, B.J.; Mullen, T. Sponsored search: An overview of the concept, history, and technology. *Int. J. Electron. Bus.* **2008**, *6*, 114–131. [CrossRef]
2. Gelenbe, E.; Schmajuk, N.; Staddon, J.; Reif, J. Autonomous search by robots and animals: A survey. *Robot. Auton. Syst.* **1997**, *22*, 23–34. [CrossRef]
3. Gelenbe, E. Search in unknown random environments. *Phys. Rev. E* **2010**, *82*, 061112. [CrossRef] [PubMed]
4. Abdelrahman, O.H.; Gelenbe, E. Time and energy in team-based search. *Phys. Rev. E* **2013**, *87*, 032125. [CrossRef]
5. Gelenbe, E.; Abdelrahman, O.H. Search in the universe of big networks and data. *IEEE Netw.* **2014**, *28*, 20–25. [CrossRef]
6. Murugesan, S. Intelligent Agents on The Internet and Web: Applications and Prospects. *Informatica* **1999**, *23*, 3.
7. Etzioni, O.; Weld, D.S. Intelligent Agents on the Internet: Fact, Fiction, and Forecast. *IEEE Expert.* **1995**, *10*, 44–49. [CrossRef]
8. Zacharis, N.Z.; Panayiotopoulos, T. A metagenetic algorithm for information filtering and collection from the World Wide Web. *Expert Syst.* **2001**, *18*, 99–108. [CrossRef]
9. Pazzani, M.J.; Billsus, D. Learning and Revising User Profiles: The Identification of Interesting Web Sites. *Mach. Learn.* **1997**, *27*, 313–331. [CrossRef]
10. Lieberman, H. Letizia: An Agent That Assists Web Browsing. In Proceedings of the International Joint Conference on Artificial Intelligence, Montreal, QC, Canada, 20–25 August 1995; pp. 924–929.
11. Joachims, T.; Freitag, D.; Mitchell, T.M. Web Watcher: A Tour Guide for the World Wide Web. In Proceedings of the International Joint Conference on Artificial Intelligence, Nagoya, Japan, 23–29 August 1997; pp. 770–777.
12. Bruce Krulwich: Lifestyle Finder: Intelligent User Profiling Using Large-Scale Demographic Data. *AI Mag.* **1997**, *18*, 37–45.
13. Spink, A.; Jansen, B.J.; Kathuria, V.; Koshman, S. Overlap among major web search engines. *Internet Res.* **2006**, *16*, 419–426. [CrossRef]
14. Micarelli, A.; Gasparetti, F.; Sciarrone, F.; Gauch, S. Personalized Search on the World Wide Web. In *The Adaptive Web*; Springer: Berlin/Heidelberg, Germany, 2007; pp. 195–230.
15. Matthijs, N.; Radlinski, F. Personalizing Web Search Using Long Term Browsing History. In Proceedings of the Web Search and Data Mining, Hong Kong, China, 9–12 February 2011; pp. 25–34.
16. Liu, F.; Yu, C.T.; Meng, W. Personalized Web Search for Improving Retrieval Effectiveness. *IEEE Trans. Knowl. Data Eng.* **2004**, *16*, 28–40.

17. Radlinski, F.; Dumais, S.T. Improving Personalized Web Search Using Result Diversification. In Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, New York, NY, USA, 6–10 August 2006; pp. 691–692.
18. Backstrom, L.; Kleinberg, J.M.; Kumar, R.; Novak, J. Spatial variation in search engine queries. In Proceedings of the 17th International Conference on World Wide Web, Beijing, China, 21–25 April 2008; pp. 357–366.
19. Wu, F.; Madhavan, J.; Halevy, A.Y. Identifying Aspects for Web-Search Queries. *J. Artif. Intell Res.* **2011**, *40*, 677–700. [CrossRef]
20. Tezuka, T.; Tanaka, K. Temporal and Spatial Attribute Extraction from Web Documents and Time-Specific Regional Web Search System. In *Web and Wireless Geographical Information Systems*; Springer: Berlin/Heidelberg, Germany, 2004; Volume 3428, pp. 14–25.
21. Lin, S.; Jin, P.; Zhao, X.; Yue, L. Exploiting temporal information in Web search. *Expert Syst. Appl.* **2014**, *41*, 331–341. [CrossRef]
22. Yu, P.S.; Li, X.; Liu, B. On the Temporal Dimension of Search. In Proceedings of the 13th International World Wide Web conference on Alternate Track Papers & Posters, New York, NY, USA, 17–20 May 2004; pp. 448–449.
23. Metzler, D.; Jones, R.; Peng, F.; Zhang, R. Improving Search Relevance for Implicitly Temporal Queries. In Proceedings of the 32nd International ACM SIGIR Conference on Research and Development in Information Retrieval, Boston, MA, USA, 19–23 July 2009; pp. 700–701.
24. Pasca, M. Towards temporal Web search. In Proceedings of the ACM Symposium on Applied Computing, Ceará, Brazil, 16–20 March 2008; pp. 1117–1121.
25. Campos, R.; Dias, G.; Jorge, A.M.; Nunes, C. GTE-Cluster: A Temporal Search Interface for Implicit Temporal Queries. European Conference on Information Retrieval. *Adv. Inf. Retr.* **2014**, *8416*, 775–779.
26. Xu, Z.; Liu, Y.; Mei, L.; Hu, C.; Chen, L. Generating temporal semantic context of concepts using web search engines. *J. Netw. Comput. Appl.* **2014**, *43*, 42–55. [CrossRef]
27. Meng, W.; Yu, C.T.; Liu, K. Building efficient and effective metasearch engines. *ACM Comput. Surv.* **2002**, *34*, 48–89. [CrossRef]
28. Manoj, M.; Elizabeth, J. Information retrieval on Internet using meta-search engines: A review. *J. Sci. Ind. Res.* **2008**, *67*, 739–746.
29. Jadidoleslamy, H. Search Result Merging and Ranking Strategies in Meta-Search Engines: A Survey. *Int. J. Comput. Sci.* **2012**, *9*, 239–251.
30. Gulli, A.; Signorini, A. Building an Open Source Meta-Search Engine. In Proceedings of the 14th International Conference on World Wide Web, WWW 2005, Chiba, Japan, 10–14 May 2005; pp. 1004–1005.
31. Wu, Z.; Raghavan, V.V.; Qian, H.; Vuyyuru, R.; Meng, W.; He, H.; Yu, C.T. Towards Automatic Incorporation of Search Engines into a Large-Scale Metasearch Engine. In Proceedings of the IEEE/WIC International Conference on Web Intelligence, Halifax, NS, Canada, 13–17 October 2003; pp. 658–661.
32. Aslam, J.A.; Montague, M.H. Models for Metasearch. In Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval Special Interest Group on Information Retrieval, New Orleans, LA, USA, 9–12 September 2001; pp. 275–284.
33. Dwork, C.; Kumar, R.; Naor, M.; Sivakumar, D. Rank aggregation methods for the Web. In Proceedings of the 10th International Conference on World Wide Web, Hong Kong, China, 1–5 May 2001; pp. 613–622.
34. Lu, Y.; Meng, W.; Shu, L.; Yu, C.T.; Liu, K. Evaluation of Result Merging Strategies for Metasearch Engines. In Proceedings of the Web Information Systems Engineering, New York, NY, USA, 20–22 November 2005; pp. 53–66.
35. Manmatha, R.; Sever, H. A formal approach to score normalization for metasearch. In Proceedings of the Second International Conference on Human Language Technology Research, San Diego, CA, USA, 24–27 March 2002; pp. 98–103.
36. Akritidis, L.; Katsaros, D.; Bozanis, P. Effective Ranking Fusion Methods for Personalized Metasearch Engines. In Proceedings of the Panhellenic Conference on Informatics, Samos Island, Greece, 28–30 August 2008; pp. 39–43.
37. Meng, W.; Wu, Z.; Yu, C.T.; Li, Z. A highly scalable and effective method for metasearch. *ACM Trans. Inf. Syst.* **2001**, *19*, 310–335. [CrossRef]
38. Kumar, B.T.S.; Pavithra, S.M. Evaluating the searching capabilities of search engines and metasearch engines: A comparative study. *Ann. Libr. Inf. Stud.* **2010**, *57*, 87–97.

39. Zamir, O.; Etzioni, O. Web Document Clustering: A Feasibility Demonstration. In Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval Special Interest Group on Information Retrieval, Melbourne, Australia, 24–28 August 1998; pp. 46–54.
40. Osinski, S.; Weiss, D. A Concept-Driven Algorithm for Clustering Search Results. *IEEE Intell. Syst.* **2005**, *20*, 48–54. [CrossRef]
41. Carpineto, C.; Osinski, S.; Romano, G.; Weiss, D. A survey of Web clustering engines. *ACM Comput Surv.* **2009**, *41*, 17. [CrossRef]
42. Crabtree, D.; Gao, X.; Andreae, P. Improving Web Clustering by Cluster Selection. In Proceedings of the 2005 IEEE/WIC/ACM International Conference on Web Intelligence, Compiegne, France, 19–22 September 2005; pp. 172–178.
43. Geraci, F.; Pellegrini, M.; Maggini, M.; Sebastiani, F. Cluster Generation and Cluster Labelling for Web Snippets: A Fast and Accurate Hierarchical Solution. In Proceedings of the 13th International Conference on String Processing and Information Retrieval, Glasgow, UK, 11–13 October 2006; pp. 25–36.
44. Zeng, H.; He, Q.; Chen, Z.; Ma, W.; Ma, J. Learning to Cluster Web Search Results. In Proceedings of the 27th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Sheffield, UK, 25–29 July 2004; pp. 210–217.
45. He, J.; Meij, E.; de Rijke, M. Result diversification based on query-specific cluster ranking. *J. Assoc. Inf. Sci. Technol.* **2011**, *62*, 550–571. [CrossRef]
46. Ngo, C.L.; Nguyen, H.S. A Method of Web Search Result Clustering Based on Rough Sets. In Proceedings of the 2005 IEEE/WIC/ACM International Conference on Web Intelligence, Compiegne, France, 19–22 September 2005; pp. 673–679.
47. Wang, X.; Zhai, C. Learn from Web Search Logs to Organize Search Results. In Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Amsterdam, The Netherlands, 23–27 July 2007; pp. 87–94.
48. Li, Z.; Wu, X. A Phrase-Based Method for Hierarchical Clustering of Web Snippets. In Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence, Atlanta, GA, USA, 11–15 July 2010; pp. 1947–1948.
49. Zhang, D.; Dong, Y. Semantic, Hierarchical, Online Clustering of Web Search Results. In Proceedings of the Advanced Web Technologies and Applications, Hangzhou, China, 14–17 April 2004; pp. 69–78.
50. Ferragina, P.; Gulli, A. The Anatomy of a Hierarchical Clustering Engine for Web-page, News and Book Snippets. In Proceedings of the Fourth IEEE International Conference on Data Mining, Brighton, UK, 1–4 November 2004; pp. 395–398.
51. Baeza-Yates, R.A.; Hurtado, C.A.; Mendoza, M. Query Clustering for Boosting Web Page Ranking. In Proceedings of the Advances in Web Intelligence, Second International Atlantic Web Intelligence Conference, AWIC 2004, Cancun, Mexico, 16–19 May 2004; pp. 164–175.
52. Sismanidou, A.; Palacios, M.; Tafur, J. Progress in airline distribution systems: The threat of new entrants to incumbent players. *J. Ind. Eng. Manag.* **2009**, *2*, 251–272. [CrossRef]
53. Granados, N.F.; Kauffman, R.J.; King, B. The Emerging Role of Vertical Search Engines in Travel Distribution: A Newly-Vulnerable Electronic Markets Perspective. In Proceedings of the 41st Annual Hawaii International Conference on System Sciences, Waikoloa, HI, USA, 7–10 January 2008; pp. 389–399.
54. Werthner, H. Intelligent Systems in Travel and Tourism. In Proceedings of the 18th International Joint Conference on Artificial Intelligence, Acapulco, Mexico, 9–15 August 2003; pp. 1620–1625.
55. Jansen, B.J.; Ciamacca, C.C.; Spink, A. An Analysis of Travel Information Searching on the Web. *J. Inf. Technol. Tour.* **2008**, *10*, 101–118. [CrossRef]
56. Ghose, A.; Ipeirotis, P.G.; Li, B. Designing Ranking Systems for Hotels on Travel Search Engines to Enhance User Experience. In Proceedings of the International Conference on Information Systems, Saint Louis, MO, USA, 12–15 December 2010; Volume 113, pp. 1–19.
57. Xiang, Z.; Fesenmaier, D.R. An Analysis of Two Search Engine Interface Metaphors for Trip Planning. *J. Inf. Technol. Tour.* **2004**, *7*, 103–117. [CrossRef]
58. Kruepl, B.; Holzinger, W.; Darmaputra, Y.; Baumgartner, R. A flight Metasearch engine with Metamorph. In Proceedings of the International Conference on World Wide Web, Madrid, Spain, 20–24 April 2009; pp. 1069–1070.
59. Mitsche, N. Understanding the Information Search Process within a Tourism Domain-specific Search Engine. In *Information and Communication Technologies in Tourism*; Springer: Vienna, Austria, 2005; pp. 183–193.

60. Meho, L.I.; Rogers, Y. Citation counting, citation ranking, and h-index of human-computer interaction researchers: A comparison of Scopus and Web of Science. *J. Am. Soc. Inf. Sci. Technol.* **2008**, *59*, 1711–1726. [CrossRef]
61. Bar-Ilan, J. Which h-index—A comparison of WoS, Scopus and Google Scholar. *Scientometrics* **2008**, *74*, 257–271. [CrossRef]
62. Bar-Ilan, J. Informetrics at the beginning of the 21st century—A review. *J. Informetr.* **2008**, *2*, 1–52. [CrossRef]
63. Beel, J.; Gipp, B.; Wilde, E. Academic Search Engine Optimization (ASEO): Optimizing Scholarly Literature for Google Scholar and Co. *J. Sch. Publ.* **2010**, *41*, 176–190. [CrossRef]
64. Hirsch, J.E. An index to quantify an individual's scientific research output. *Proc. Natl. Acad. Sci. USA* **2005**, *102*, 569–572. [CrossRef] [PubMed]
65. Beel, J.; Gipp, B. Google Scholar's Ranking Algorithm: An Introductory Overview. In Proceedings of the International Society for Scientometrics and Infometrics, Rio de Janeiro, Brazil, 14–17 July 2009; pp. 439–446.
66. Lewandowski, D. Google Scholar as a tool for discovering journal articles in library and information science. *Online Inf. Rev.* **2010**, *34*, 250–262. [CrossRef]
67. Walters, W.H. Comparative recall and precision of simple and expert searches in Google Scholar and eight other databases. *Libr. Acad.* **2011**, *11*, 971–1006. [CrossRef]
68. Harzing, A.; Alakangas, S. Google Scholar, Scopus and the Web of Science: A longitudinal and cross-disciplinary comparison. *Scientometrics* **2016**, *106*, 787–804. [CrossRef]
69. Gipp, B.; Beel, J.; Hentschel, C. Scienstein: A Research Paper Recommender System. In Proceedings of the International Conference on Emerging Trends in Computing, Virudhunagar, India, 8–10 January 2009; pp. 309–315.
70. Gyongyi, Z.; Garcia-Molina, H. Web Spam Taxonomy. In Proceedings of the First International Workshop on Adversarial Information Retrieval on the Web, Chiba, Japan, 10–14 May 2005; pp. 1–9.
71. Boyan, J.; Freitag, D.; Joachims, T. A Machine Learning Architecture for Optimizing Web Search Engines. In Proceedings of the Association for the Advancement of Artificial Intelligence Workshop on Internet-Based Information Systems, Portland, OR, USA, 10 May 1996; pp. 1–8.
72. Gu, X.; Chen, J.; Ma, W.; Chen, G. Visual Based Content Understanding towards Web Adaptation. In Proceedings of the Second International Conference on Adaptive Hypermedia and Adaptive Web-Based Systems, Malaga, Spain, 29–31 May 2002; pp. 164–173.
73. Cai, D.; Yu, S.; Wen, J.; Ma, W. Extracting content structure for web pages based on visual representation. In Proceedings of the 5th Asia-Pacific Web Conference on Web Technologies and Applications, Xi'an, China, 23–25 April 2003; pp. 406–417.
74. Yu, S.; Cai, D.; Wen, J.; Ma, W. Improving pseudo-relevance feedback in web information retrieval using web page segmentation. In Proceedings of the 12th International Conference on World Wide Web, Budapest, Hungary, 20–24 May 2003; pp. 11–18.
75. Wu, H.C.; Luk, R.W.P.; Wong, K.; Kwok, K. Interpreting TF-IDF term weights as making relevance decisions. *ACM Trans. Inf. Syst.* **2008**, *26*, 13. [CrossRef]
76. Kemp, C.; Ramamohanarao, K. Long-Term Learning for Web Search Engines. Principles and Practice of Knowledge Discovery in Databases. In *European Conference on Principles of Data Mining and Knowledge Discovery, Lecture Notes in Computer Science (Lecture Notes in Artificial Intelligence)*; Springer: Berlin/Heidelberg, Germany, 2002; Volume 2431, pp. 263–274.
77. Jin, P.; Li, X.; Chen, H.; Yue, L. CT-Rank: A Time-aware Ranking Algorithm for Web Search. *J. Converg. Inf. Technol.* **2010**, *5*, 99–111.
78. Haveliwala, T.H. Topic-Sensitive PageRank: A Context-Sensitive Ranking Algorithm for Web Search. *IEEE Trans. Knowl. Data Eng.* **2003**, *15*, 784–796. [CrossRef]
79. Brin, S.; Page, L. The Anatomy of a Large-Scale Hypertextual Web Search Engine. *Comput Netw.* **1998**, *30*, 107–117. [CrossRef]
80. Kleinberg, J.M. Authoritative Sources in a Hyperlinked Environment. In Proceedings of the Symposium on Discrete Algorithms, Association for Computing Machinery and Society for Industrial and Applied Mathematics, San Francisco, CA, USA, 25–27 January1998; pp. 668–677.
81. Lempel, R.; Moran, S. The stochastic approach for link-structure analysis (SALSA) and the TKC effect. *Comput. Netw.* **2000**, *33*, 387–401. [CrossRef]

82. Tao, W.; Zuo, W. Query-sensitive self-adaptable Web page ranking algorithm. In Proceedings of the 2003 International Conference on Machine Learning and Cybernetics, Xi'an, China, 5 November 2003; pp. 413–418.
83. Bidoki, A.M.Z.; Yazdani, N. DistanceRank: An intelligent ranking algorithm for web pages. *Inf. Process. Manag.* **2008**, *44*, 877–892. [CrossRef]
84. Delbru, R.; Toupikov, N.; Catasta, M.; Tummarello, G.; Decker, S. Hierarchical Link Analysis for Ranking Web Data. In Proceedings of the Extended Semantic Web Conference, Heraklion, Greece, 30 May–3 June 2010; Volume 2, pp. 225–239.
85. Jiao, J.; Yan, J.; Zhao, H.; Fan, W. ExpertRank: An Expert User Ranking Algorithm in Online Communities. In Proceedings of the 2009 International Conference on New Trends in Information and Service Science, Beijing, China, 30 June–2 July 2009; pp. 674–679.
86. Ljosland, M. Evaluation of Web search engines and the search for better ranking algorithms. In Proceedings of the Workshop on Evaluation of Web Retrieval, Special Interest Group on Information Retrieval Conference, Berkeley, CA, USA, 14 August 1999; pp. 1–6.
87. Gordon, M.D.; Pathak, P. Finding Information on the World Wide Web: The Retrieval Effectiveness of Search Engines. *Inf. Process. Manag.* **1999**, *35*, 141–180. [CrossRef]
88. Vaughan, L. New measurements for search engine evaluation proposed and tested. *Inf. Process. Manag.* **2004**, *40*, 677–691. [CrossRef]
89. Li, L.; Shang, Y.; Zhang, W. Relevance Evaluation of Search Engines' Query Results. In Proceedings of the World Wide Web Posters, International World Wide Web Conference, Hong Kong, China, 1–5 May 2001; pp. 1–2.
90. Singhal, A.; Kaszkiel, M. A case study in web search using TREC algorithms. In Proceedings of the World Wide Web, International World Wide Web Conference, Hong Kong, China, 1–5 May 2001; pp. 708–716.
91. Hawking, D.; Craswell, N.; Thistlewaite, P.B.; Harman, D. Results and Challenges in Web Search Evaluation. *Comput. Netw.* **1999**, *31*, 1321–1330. [CrossRef]
92. Hawking, D.; Craswell, N.; Bailey, P.; Griffiths, K. Measuring Search Engine Quality. *Inf. Retr.* **2001**, *4*, 33–59. [CrossRef]
93. Ali, R.; Beg, M.M.S. An overview of Web search evaluation methods. *Comput. Electr. Eng.* **2011**, *37*, 835–848. [CrossRef]
94. Can, F.; Nuray, R.; Sevdik, A.B. Automatic performance evaluation of Web search engines. *Inf. Process. Manag.* **2004**, *40*, 495–514. [CrossRef]
95. Ali, R.; Mohd, M.; Beg, S. Automatic Performance Evaluation of Web Search Systems using Rough Set based Rank Aggregation. In Proceedings of the First International Conference on Intelligent Human Computer Interaction, Allahabad, India, 20–23 January 2009; pp. 344–358.
96. Hou, J. Research on Design of an Automatic Evaluation System of Search Engine. In Proceedings of the IEEE International Conference on Future Computer and Communication, Wuhan, China, 6–7 June 2009; pp. 16–18.
97. Chowdhury, A.; Soboroff, I. Automatic Evaluation of World Wide Web Search Services. In Proceedings of the 25th annual international Association for Computer Machinery Special Interest Group on Information Retrieval Conference on Research and Development in Information Retrieval, Tampere, Finland, 11–15 August 2002; pp. 421–422.
98. Thelwall, M. Quantitative comparisons of search engine results. *J. Am. Soc. Inf. Sci. Technol.* **2008**, *59*, 1702–1710. [CrossRef]
99. Joachims, T. Optimizing search engines using clickthrough data. In Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Edmonton, AB, Canada, 23–26 July 2002; pp. 133–142.
100. Xiang, B.; Jiang, D.; Pei, J.; Sun, X.; Chen, E.; Li, H. Context-Aware Ranking in Web Search. In Proceedings of the Association for Computer Machinery Special Interest Group on Information Retrieval Conference on Research and Development in Information Retrieval, Geneva, Switzerland, 19–23 July 2010; pp. 451–458.
101. Dong, A.; Chang, Y.; Zheng, Z.; Mishne, G.; Bai, J.; Zhang, R.; Buchner, K.; Liao, C.; Diaz, F. Towards recency ranking in web search. In Proceedings of the Third ACM International Conference on Web Search and Data Mining, New York, NY, USA, 3–6 February 2010; pp. 11–20.
102. Mohan, A.; Chen, Z.; Weinberger, K.Q. Web-Search Ranking with Initialized Gradient Boosted Regression Trees. In Proceedings of the Learning to Rank Challenge, Haifa, Israel, 25 June 2010; Volume 14, pp. 77–89.

103. Kelly, D.; Teevan, J. Implicit Feedback for Inferring User Preference: A Bibliography. In *Newsletter ACM Special Interest Group of Information Retrieval Forum*; ACM: New York, NY, USA, 2003; Volume 37, pp. 18–28.
104. Jawaheer, G.; Szomszor, M.; Kostkova, P. Comparison of implicit and explicit feedback from an online music recommendation service. In Proceedings of the International Workshop on Information Heterogeneity and Fusion in Recommender Systems, Barcelona, Spain, 26–30 September 2010; pp. 47–51.
105. White, R.W.; Ruthven, I.; Jose, J.M. The Use of Implicit Evidence for Relevance Feedback in Web Retrieval. In Proceedings of the European Conference on Information Retrieval, Advances in Information Retrieval, Glasgow, UK, 25–27 March 2002; pp. 93–109.
106. Fox, S.; Karnawat, K.; Mydland, M.; Dumais, S.; White, T. Evaluating Implicit Measures to Improve Web Search. *J. ACM Trans. Inf. Syst.* **2005**, *23*, 147–168. [CrossRef]
107. Joachims, T.; Granka, L.; Pan, B.; Hembrooke, H.; Gay, G. Accurately Interpreting Clickthrough Data as Implicit Feedback. In Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Salvador, Brazil, 15–19 August 2005; ACM: New York, NY, USA; pp. 154–161.
108. Radlinski, F.; Joachims, T. Query Chains, Learning to Rank from Implicit Feedback. In Proceedings of the eleventh ACM SIGKDD International Conference on Knowledge Discovery in Data Mining, Chicago, IL, USA, 21–24 August 2005; pp. 239–248.
109. Oard, D.W.; Kim, J. Implicit Feedback for Recommender Systems. In Proceedings of the Association for the Advancement of Artificial Intelligence Workshop on Recommender Systems, Madison, WI, USA, 26–27 July 1998; pp. 81–83.
110. Lee, W.; Liu, C.; Lu, C. Intelligent agent-based systems for personalized recommendations in Internet commerce. *Expert Syst. Appl.* **2002**, *22*, 275–284. [CrossRef]
111. Almazro, D.; Shahatah, G.; Albdulkarim, L.; Kherees, M.; Martinez, R.; Nzoukou, W. A Survey Paper on Recommender Systems. *arXiv* **2010**, arXiv:1006.527.
112. Adomavicius, G.; Tuzhilin, A. Toward the Next Generation of Recommender Systems: A Survey of the State-of-the-Art and Possible Extensions. *IEEE Trans. Knowl. Data Eng.* **2005**, *17*, 734–749. [CrossRef]
113. Adomavicius, G.; Manouselis, N.; Kwon, Y. Multi-Criteria Recommender Systems. In *Recommender Systems Handbook*; Springer: Berlin, Germany, 2011; pp. 769–803.
114. Cremonesi, P.; Tripodi, A.; Turrin, R. Cross-Domain Recommender Systems. In Proceedings of the 2011 IEEE 11th International Conference on Data Mining Workshops, Vancouver, BC, Canada, 1–11 December 2011; pp. 496–503.
115. Schröder, G.; Thiele, M.; Lehner, W. Setting Goals and Choosing Metrics for Recommender System Evaluations. In Proceedings of the Conference on Recommender Systems, Chicago, IL, USA, 23–27 October 2011; pp. 78–85.
116. Yang, X.; Guo, Y.; Liu, Y.; Steck, H. A survey of collaborative filtering based social recommender systems. *Comput. Commun.* **2014**, *41*, 1–10. [CrossRef]
117. Liu, J.; Chen, M.Z.Q.; Chen, J.; Deng, F.; Zhang, H.; Zhang, Z.; Zhou, T. Recent Advances in Personal Recommender Systems. *Int. J. Inf. Syst. Sci.* **2009**, *5*, 230–247.
118. Cho, Y.H.; Kim, J.K.; Kim, S.H. A personalized recommender system based on web usage mining and decision tree induction. *Expert Syst. Appl.* **2002**, *23*, 329–342. [CrossRef]
119. Liang, T.; Lai, H.; Ku, Y. Personalized Content Recommendation and User Satisfaction: Theoretical Synthesis and Empirical Findings. *J. Manag. Inf. Syst. Arch.* **2007**, *23*, 45–70. [CrossRef]
120. Tiroshi, A.; Kuflik, T.; Kay, J.; Kummerfeld, B. Recommender Systems and the Social Web. In Proceedings of the 19th International Conference on Advances in User Modeling, Girona, Spain, 11–15 July 2011; pp. 60–70.
121. Essayeh, A.; Abed, M. A Recommendation System for Enhancing the Personalized Search Itineraries in the Public Transportation Domain. In Proceedings of the 19th International Conference on Enterprise Information Systems, Porto, Portugal, 26–29 April 2017; Volume 1, pp. 415–423.
122. Yu, H.; Dan, Y.; Jing, L.; Mu, Z. Research on Personalized Recommender System for Tourism Information Service. *Comput. Eng. Intell. Syst.* **2013**, *4*, 32–45.
123. Chedrawy, Z.; Abidi, S.S.R. A Web Recommender System for Recommending, Predicting and Personalizing Music Playlists. In Proceedings of the International Conference on Web Information Systems Engineering, Poznan, Poland, 5–7 October 2009; pp. 335–342.

124. Renjith, S.; Anjali, C. A personalized mobile travel recommender system using hybrid algorithm. In Proceedings of the First International Conference on Computational Systems and Communications, Trivandrum, India, 17–18 December 2014; pp. 12–17.
125. Capala, M. Machine learning just got more human with Google's RankBrain. 2016. Available online: https://thenextweb.com/artificial-intelligence/2016/09/02/machine-learning-just-got-more-human-with-googles-rankbrain/ (accessed on 30 December 2018).
126. Scarselli, F.; Yong, S.L.; Gori, M.; Hagenbuchner, M.; Tsoi, A.C.; Maggini, M. Graph Neural Networks for Ranking Web Pages. In Proceedings of the 2005 IEEE/WIC/ACM International Conference on Web Intelligence, Compiegne, France, 19–22 September 2005; pp. 666–672.
127. Chau, M.; Chen, H. Incorporating Web Analysis into Neural Networks: An Example in Hopfield Net Searching. *IEEE Trans. Syst. Man Cybern. C* **2007**, *37*, 352–358. [CrossRef]
128. Bermejo, S.; Dalmau, J. Web Meta-search Using Unsupervised Neural Networks. *Artif. Neural Nets Probl. Solving Methods* **2003**, *2*, 711–718.
129. Shu, B.; Kak, S.C. A Neural Network-based Intelligent Metasearch Engine. *Inf. Sci.* **1999**, *120*, 1–11. [CrossRef]
130. Wang, S.; Xu, K.; Zhang, Y.; Li, F. Search Engine Optimization Based on Algorithm of BP Neural Networks. In Proceedings of the 2011 Seventh International Conference on Computational Intelligence and Security, Hainan, China, 3–4 December 2011; pp. 390–394.
131. Choi, Y.S.; Yoo, S.I. Multi-agent Web Information Retrieval: Neural Network Based Approach. In Proceedings of the Third International Symposium on Advances in Intelligent Data Analysis, Amsterdam, The Netherlands, 9–11 August 1999; pp. 499–512.
132. Burges, C.J.C.; Shaked, T.; Renshaw, E.; Lazier, A.; Deeds, M.; Hamilton, N.; Hullender, G.N. Learning to rank using gradient descent. In Proceedings of the 22nd International Conference on Machine Learning, Bonn, Germany, 7–11 August 2005; pp. 89–96.
133. Richardson, M.; Prakash, A.; Brill, E. Beyond PageRank: MACHINE learning for static ranking. In Proceedings of the 15th International Conference on World Wide Web, Edinburgh, UK, 23–26 May 2006; pp. 707–715.
134. Agichtein, E.; Brill, E.; Dumais, S.T. Improving Web Search Ranking by Incorporating User Behavior Information. In Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Seattle, WA, USA, 6–11 August 2006; pp. 19–26.
135. Rigutini, L.; Papini, T.; Maggini, M.; Scarselli, F. SortNet: Learning to Rank by a Neural Preference Function. *IEEE Trans. Neural Netw.* **2011**, *22*, 1368–1380. [CrossRef]
136. Cao, Z.; Qin, T.; Liu, T.; Tsai, M.; Li, H. Learning to rank: FROM pairwise approach to listwise approach. In Proceedings of the International Conference on Machine Learning, Cincinnati, OH, USA, 13–15 December 2007; pp. 129–136.
137. Patil, S.K.; Mane, Y.D.; Dabre, K.R.; Dewan, P.R.; Kalbande, D.R. An Efficient Recommender System using Collaborative Filtering Methods with K-separability Approach. *Int. J. Eng. Res. Appl.* **2012**, *2012*, 30–35.
138. Chang, C.; Chen, P.; Chiu, F.; Chen, Y. Application of neural networks and Kano's method to content recommendation in web personalization. *Expert Syst. Appl.* **2009**, *36*, 5310–5316. [CrossRef]
139. Lee, M.; Choi, P.; Woo, Y. A Hybrid Recommender System Combining Collaborative Filtering with Neural Network. In Proceedings of the Second International Conference on Adaptive Hypermedia and Adaptive Web-Based Systems, Malaga, Spain, 29–31 May 2002; pp. 531–534.
140. Devi, M.K.K.; Samy, R.T.; Kumar, S.V.; Venkatesh, P. Probabilistic neural network approach to alleviate sparsity and cold start problems in collaborative recommender systems. In Proceedings of the 2010 IEEE International Conference on Computational Intelligence and Computing Research, Coimbatore, India, 28–29 December 2010; pp. 1–4.
141. Vassiliou, C.; Stamoulis, D.; Martakos, D.; Athanassopoulos, S. A recommender system framework combining neural networks & collaborative filtering. In Proceedings of the 5th WSEAS International Conference on Instrumentation, Measurement, Circuits and Systems, Hangzhou, China, 16–18 April 2006; pp. 285–290.
142. Kongsakun, K.; Fung, C.C. Neural Network Modeling for an Intelligent Recommendation System Supporting SRM for Universities in Thailand. *World Sci. Eng. Acad. Soc. Trans. Comput.* **2012**, *2*, 34–44.
143. Billsus, D.; Pazzani, M.J. Learning Collaborative Information Filters. In Proceedings of the Fifteenth International Conference on Machine Learning, Madison, WI, USA, 24–27 July 1998; pp. 46–54.

144. Krstic, M.; Bjelica, M. Context-aware personalized program guide based on neural network. *IEEE Trans. Consum. Electron.* **2012**, *58*, 1301–1306. [CrossRef]
145. Biancalana, C.; Gasparetti, F.; Micarelli, A.; Miola, A.; Sansonetti, G. Context-aware movie recommendation based on signal processing and machine learning. In Proceedings of the 2nd Challenge on Context-Aware Movie Recommendation, Chicago, IL, USA, 23–27 October 2011; pp. 5–10.
146. Chou, P.; Li, P.; Chen, K.; Wu, M. Integrating web mining and neural network for personalized e-commerce automatic service. *Expert Syst. Appl.* **2010**, *37*, 2898–2910. [CrossRef]
147. Severyn, A.; Moschitti, A. Learning to Rank Short Text Pairs with Convolutional Deep Neural Networks. In Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval, Santiago, Chile, 9–13 August 2015; pp. 373–382.
148. Wang, B.; Klabjan, D. An Attention-Based Deep Net for Learning to Rank. *arXiv* **2017**, arXiv:1702.06106.
149. Pang, L.; Lan, Y.; Guo, J.; Xu, J.; Xu, J.; Cheng, X. DeepRank: A New Deep Architecture for Relevance Ranking in Information Retrieval. In Proceedings of the 2017 ACM on Conference on Information and Knowledge Management, Singapore, 6–10 November 2017; pp. 257–266.
150. Ai, Q.; Bi, K.; Guo, J.; Croft, W.B. Learning a Deep Listwise Context Model for Ranking Refinement. In Proceedings of the 41st International ACM SIGIR Conference on Research & Development in Information Retrieval, Ann Arbor, MI, USA, 8–12 July 2018; pp. 135–144.
151. Deng, L.; He, X.; Gao, J. Deep stacking networks for information retrieval. In Proceedings of the 2013 IEEE International Conference on Acoustics, Speech, and Signal Processing, Vancouver, BC, Canada, 26–31 May 2013; pp. 3153–3157.
152. Li, H.; Lu, Z. Deep Learning for Information Retrieval. In Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval, Pisa, Italy, 17–21 July 2016; pp. 1203–1206.
153. Xu, J.; He, X.; Li, H. Deep Learning for Matching in Search and Recommendation. In Proceedings of the 41st International ACM SIGIR Conference on Research & Development in Information Retrieval, Ann Arbor, MI, USA, 8–12 July 2018; pp. 1365–1368.
154. Shen, Y.; He, X.; Gao, J.; Deng, L.; Mesnil, G. Learning semantic representations using convolutional neural networks for web search. In Proceedings of the 23rd International Conference on World Wide Web, Seoul, Korea, 7–11 April 2014; pp. 373–374.
155. Wang, H.; Wang, N.; Yeung, D. Collaborative Deep Learning for Recommender Systems. In Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Sydney, NSW, Australia, 10–13 August 2015; pp. 1235–1244.
156. Elkahky, A.M.; Song, Y.; He, X. A Multi-View Deep Learning Approach for Cross Domain User Modeling in Recommendation Systems. In Proceedings of the 24th International Conference on World Wide Web, Florence, Italy, 18–22 May 2015; pp. 278–288.
157. Cheng, H.; Koc, L.; Harmsen, J.; Shaked, T.; Chandra, T.; Aradhye, H.; Anderson, G.; Corrado, G.; Chai, W.; Ispir, M.; et al. Wide & Deep Learning for Recommender Systems. In Proceedings of the 1st Workshop on Deep Learning for Recommender Systems, Boston, MA, USA, 15 September 2016; pp. 7–10.
158. Zhang, L.; Luo, T.; Zhang, F.; Wu, Y. A Recommendation Model Based on Deep Neural Network. *IEEE Access* **2018**, *6*, 9454–9463. [CrossRef]
159. Van den Oord, A.; Dieleman, S.; Schrauwen, B. Deep content-based music recommendation. In Proceedings of the 26th International Conference on Neural Information Processing Systems, Lake Tahoe, NV, USA, 5–10 December 2013; Volume 2, pp. 2643–2651.
160. Wang, X.; Wang, Y. Improving Content-based and Hybrid Music Recommendation using Deep Learning. In Proceedings of the 22nd ACM International Conference on Multimedia, Orlando, FL, USA, 3–7 November 2014; pp. 627–636.
161. Gelenbe, E. *Réseaux Stochastiques Ouverts avec Clients Négatifs et Positifs, et Réseaux Neuronaux. Comptes-Rendus*; Académie des Sciences de Paris: Paris, France, 1989; Volume 309, pp. 979–982.
162. Gelenbe, E.; Tucci, S. *Performances D'un Systeme Informatique Dupliqué. Comptes-Rendus*; Académie des Sciences: Paris, France, 1991; Volume 312, pp. 27–30.
163. Gelenbe, E. Queueing networks with negative and positive customers. *J. Appl. Probab.* **1991**, *28*, 656–663. [CrossRef]
164. Gelenbe, E.; Schassberger, R. Stability of product form G-Networks. *Probab. Eng. Inf. Sci.* **1992**, *6*, 271–276. [CrossRef]

165. Gelenbe, E. Random Neural Networks with Negative and Positive Signals and Product Form Solution. *Neural Comput.* **1989**, *1*, 502–510. [CrossRef]
166. Gelenbe, E. Stability of the Random Neural Network Model. *Neural Comput.* **1990**, *2*, 239–247. [CrossRef]
167. Gelenbe, E. Learning in the Recurrent Random Neural Network. *Neural Comput.* **1993**, *5*, 154–164. [CrossRef]
168. Gelenbe, E.; Fourneau, J. Random Neural Networks with Multiple Classes of Signals. *Neural Comput.* **1999**, *11*, 953–963. [CrossRef] [PubMed]
169. Gelenbe, E.; Hussain, K. Learning in the multiple class random neural network. *IEEE Trans. Neural Netw.* **2002**, *13*, 1257–1267. [CrossRef] [PubMed]
170. Gelenbe, E.; Stafylopati, A.; Likas, A. Associative memory operation of the random network model. In Proceedings of the International Conference on Artificial Neural Networks, Espoo, Finland, 24–28 June 1991; pp. 307–312.
171. Gelenbe, E.; Mao, Z.; Li, Y. Function approximation with spiked random networks. *IEEE Trans. Neural Netw.* **1999**, *10*, 3–9. [CrossRef] [PubMed]
172. Gelenbe, E.; Mao, Z.; Li, Y. Function approximation by random neural networks with a bounded number of layers. *Differ. Equ. Dyn. Syst.* **2004**, *12*, 143–170.
173. Gelenbe, E.; Timotheou, S. Random Neural Networks with Synchronized Interactions. *Neural Comput.* **2008**, *20*, 2308–2324. [CrossRef]
174. Gelenbe, E.; Timotheou, S. Synchronized Interactions in Spiked Neuronal Networks. *Comput. J.* **2008**, *51*, 723–730. [CrossRef]
175. Gelenbe, E.; Yin, Y. Deep learning with random neural networks. In Proceedings of the International Joint Conference on Neural Networks, Vancouver, BC, Canada, 24–29 July 2016; pp. 1633–1638.
176. Serrano, W.; Gelenbe, E. An Intelligent Internet Search Assistant Based on the Random Neural Network. In Proceedings of the 12th IFIP International Conference on Artificial Intelligence Applications and Innovations, Thessaloniki, Greece, 16–18 September 2016; pp. 141–153.
177. Serrano, W.; Gelenbe, E. The Random Neural Network in a neurocomputing application for Web search. *Neurocomputing* **2018**, *280*, 123–134. [CrossRef]
178. Serrano, W. A Big Data Intelligent Search Assistant Based on the Random Neural Network. In Proceedings of the International Neural Network Society Conference on Big Data, Advances in Intelligent Systems and Computing, Hyderabad, India, 28–30 October 2016; Springer: Cham, Switzerland; Thessaloniki, Greece; Volume 529, pp. 254–261.
179. Serrano, W.; Gelenbe, E. Intelligent Search with Deep Learning Clusters. In Proceedings of the Intelligent Systems Conference, London, UK, 7–8 September 2017; pp. 254–261.
180. Serrano, W.; Gelenbe, E. The Deep Learning Random Neural Network with a Management Cluster. In Proceedings of the International Conference on Intelligent Decision Technologies, Vilamoura, Portugal, 21–23 June 2017; pp. 185–195.