Information Retrieval Methods: A Literature Survey

A project submitted in partial fulfilment of the requirements for the award of the degree of

Bachelor of Technology
In
COMPUTER SCIENCE AND ENGINEERING

Submitted by: Anupreet Singh
Roll Number: 11911063
Date: May 2022

Supervised by:
Dr. Mukesh Mann
Assistant Professor

Indian Institute of Information Technology, Sonepat-132101, Haryana, India

ACKNOWLEDGEMENT

The outcome of this project required ceaseless guidance and assistance; I was extremely privileged to have got this all along the project.

I would like to take this opportunity to acknowledge all the people who have helped us whole heartedly in every stage of this project.

I am indebtedly grateful to Dr. Mukesh Mann, Assistant professor, CSE, IIIT SONEPAT for providing this opportunity in the first place and giving us all the support at every step possible and thorough guidance, in spite of having a busy and packed schedule.

I am extremely grateful and thankful for his help which played very foremost part in the research project and for providing us all the indispensable information for developing project.

Anupreet Singh

SELF DECLARATION

I hereby state that work contained in the project titled “Information Retrieval Methods: A literature Review” is original. I have followed the standards of the project ethics to the best of my abilities. I have acknowledged all the sources of knowledge which I have used in the project.

Name: Anupreet Singh
Roll no: 11911063
Department of Computer Science and Engineering, Indian Institute of Information technology, Sonepat-131201, Haryana, India

CERTIFICATE

This is to certify that Mr. Anupreet Singh has worked on the project entitled “Information Retrieval Methods :A literature Review” under my supervision and guidance.

The contents of the project, being submitted to the Department of Computer Science and Engineering, IIIT SONEPAT, HARYANA, for the award of the degree of B. Tech in Computer Science and Engineering, are original and carried out by candidate himself. This project has not been submitted in full or part for award of any other degree or diploma to this or any other university.

Dr. Mukesh Mann
Supervisor

Department of Computer Science and Engineering,
Indian Institute of Information Technology,
Sonepat-131201, Haryana, India

ABSTRACT

Name of the student : Anupreet Singh
Roll No.:11911063
Degree for which submitted: B. Tech
Department of Computer Science and Engineering
Indian Institute of Information Technology,
Sonepat

Project Title: Information Retrieval Methods :A literature Review
Name of the Project Supervisor: Dr. Mukesh Mann

Information retrieval(IR) research revolves around ranking models . Over the years, several approaches for building ranking models have been created, ranging from old heuristic methods to probabilistic methods to more current machine learning methods. In this study, we'll examine at all of the key IR approaches that have been employed and are presently being researched. Since the most widespread application of IR is in Web Search Engines, a brief overview of their requisite literature is also provided. Because of the progress of deep learning technology, we have lately seen a growing body of work in applying shallow or deep neural networks to the ranking issue in IR, referred to as neural ranking models in this study. Given the wide range of neural ranking models that have been developed, we feel it is time to review the current state, learn from existing approaches, and acquire some ideas for future improvement. In Contrast to existing reviews, this survey is written as a stand-alone paper and no prerequisite knowledge of the field is required by beginners to understand this paper, it covers almost all aspects regarding the history and future of Information Retrieval.

LIST OF FIGURES

1.Introduction

Information retrieval is the process used to organise and extract relevant and required information from large data sets using keywords specified in a user's query. It may also be defined as a way of obtaining information from a document, such as a text document, as well as searching for metadata that defines the data and for text, picture, and photo databases.

2. Application of Information Retrieval

2.1 Digital Library

A digital library is collection of textual data like journals, books, magazines etc in digital format that can be acquired by posting a query by the user. It can be stored on a local machine or accessed remotely via computer networks[1]

2.2 Search Engines

Search engines are among the most practical application of Information retrieval system(IRS).Web search engines are the most commonly used and recognised example but other examples also include Mobile search, desktop search, Social Search, Enterprise search, etc[2].

2.3 Media Search

This information retrieval system is used on local computer systems to browse, search and retrieve image, audio and video content from large media databases[1].

3. Brief Overview of the structure of a web Search Engine

In order to learn how a web search engine is an information retrieval System the reader must have a basic understanding of how the web works and how Web search engines interact with it. The World Wide Web (WWWW), sometimes known as the Web, is the world's most popular software platform. It's an online resource centre where web surfers may access papers and other web resources. Any kind of downloadable media can be used as a web resource. Hypertext links structured in Hypertext Markup Language connect documents on the web (HTML). Web browsers use web search engines to fetch information on web pages relevant to users queries .

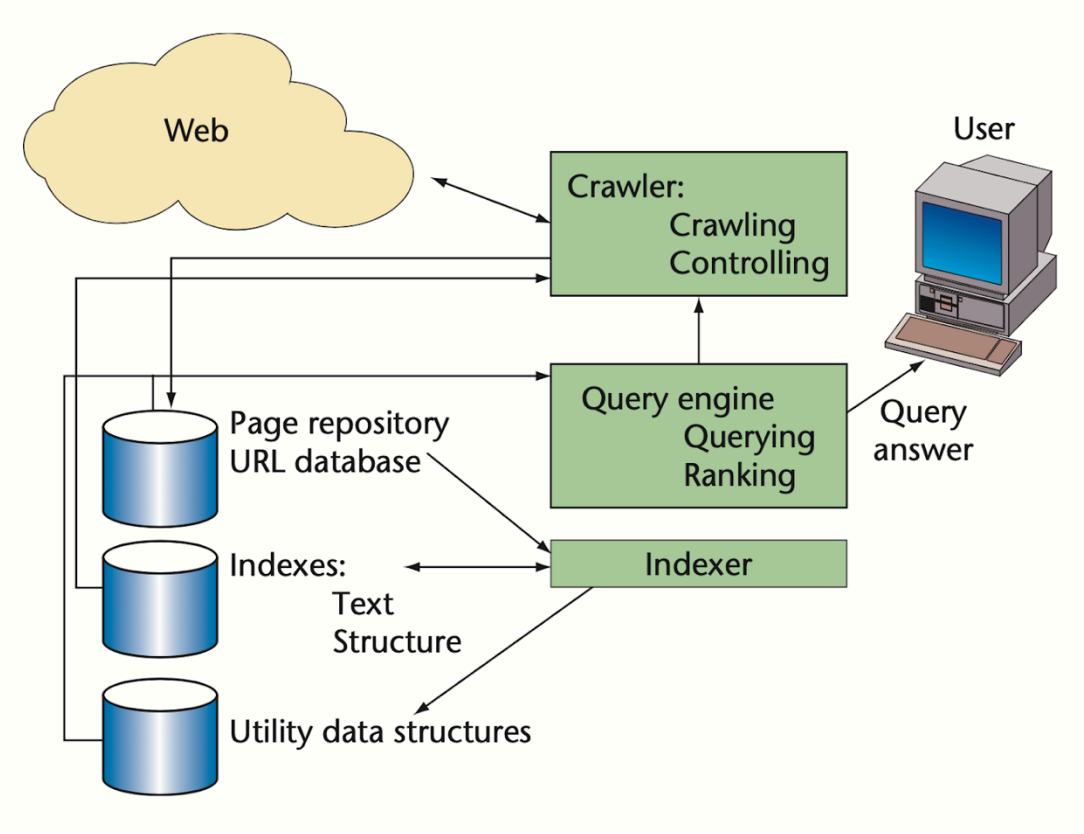


Fig 1. Architecture of a Search Engine[3]

As shown in Fig-1 the query process and the indexing process are the two fundamental components of the architecture of the search engine. The indexing process can also further be broken down into two parts: the crawler, and the indexer.

3.1 Indexing Process-

3.1.1 Crawler
This is a piece of software that crawls the Internet in a systematic and automatic manner, sending new or updated sites to a repository for processing. Robots, bots, spiders, and harvesters are all terms used to describe crawlers.

3.1.2 Indexer
The Indexer receives information from the Repository, which is passed to it by the crawler: The information consists of a collection of web pages (Corpus) crawled from the Internet, as well as metadata acquired by the Crawler and saved in the Repository. In some cases, there is no direct interface between the crawler and the indexer. The crawler receives URLs (for example, from the URL server) and converts them into local copies of available documents on the internet, complete with metadata. The indexer then begins the indexing process independently of the crawler. The indexer can work with/index one version of the Web, the crawler with another, and the query process/engine with an earlier version of the Web representation, older than the ones used by the Crawler and Indexer. Each month, one of the three copies is replaced, and one becomes obsolete (the current copy used by the query engine).From around 2008 to 2010, Google employed this notion and form. Things have altered dramatically since then, as will be detailed later.

When a query is posted a Web search engine searches the web's index, not the entire internet. When we do an ad-hoc Web search, we are not searching the Web directly, but rather the representation of the Web's documents as abstracted by the created index. In most situations, the index comprises a (sometimes inadequate) representation of the Web.

The indexer initially builds a forward index, which is an intermediate data structure. After that, the indexer uses a sorting operation to reverse the forward index.

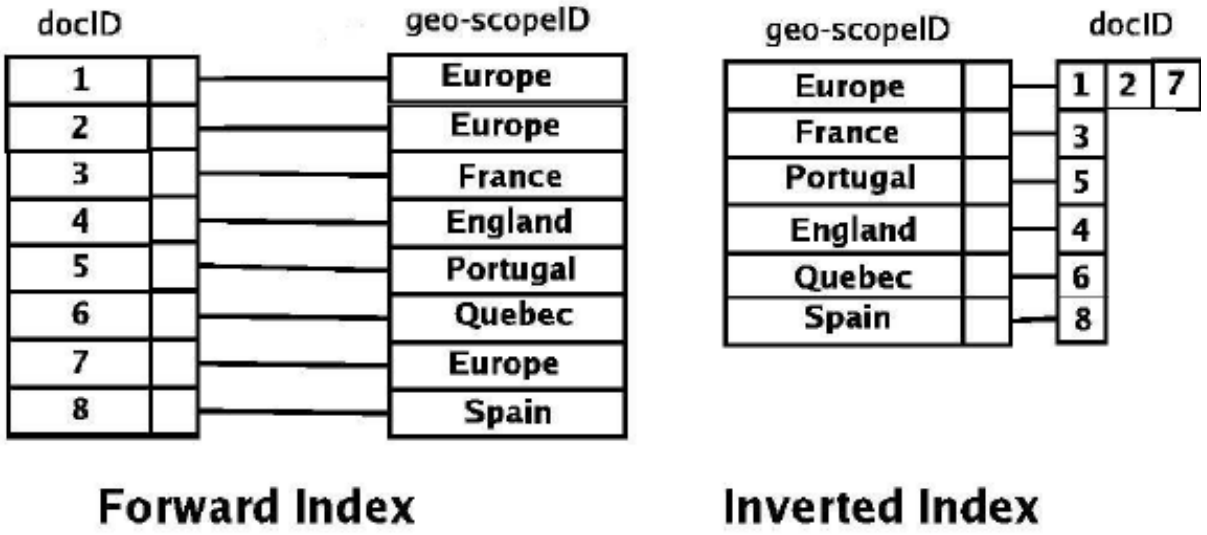


Fig-2.Example of Forward Index vs Inverted Index[4]

3.1.2.1 Forward Index

The indexer creates a series of tuples for each document in the collection that describe it. Each tuple has information about the document (docID), the token/keyword/index-term encountered (wordID), a token/keyword/index-term offset from the document's start (eg-word or character offset), and some context information (eg-does the word appear in the title of an HTML or other document, or in the anchor field of a link, in modified font such as in bold or emphasised font or in elevated font size).

3.1.2.2 Inverted Index

A sorting operation is performed on the forward index to generate an inverted index. The forward index tuples would then be sorted by wordID first, then by docID (for the same wordID tuples), and lastly by offset/context (for the same wordID, docID).The index for a given term is a list of documents (docIDs/URLs) that include that phrase in this format.

How is it Inverted?
Because the information it contains is the polar opposite of that present in a file. An inverted index stores the documents that include a specific index-term, whereas a file stores a list of index-terms (i.e. the words or tokens in the file).

Why use Inverted indexing?

If forward indexing is utilised as a data structure, indexing is quick since keywords are connected as they are discovered, but searching is sluggish. The indexing is slow in an inverted index since each word must be validated before producing the index, but the search results are rapid. This is why an inverted index is employed in a search engine (IRS) because the IRS's primary aim is to offer quick and correct responses to a query.

3.2 Query Process

The query engine's query user interface allows a web-search engine to communicate with the outside world. The user provides the query to the query engine using that interface, and the query engine then uses the index to satisfy the query. However, this "traditional method" is only employed around 10-15% of the time; more often than not, the inquiry has already been asked, and therefore the result has already been precomputed and is stored in the cache of the online search engine. As soon as the response is computed, a web server generates an HTML output containing the query's results.
The user writes a query in the query language of the search engine. The user interface parses user queries and translates search phrases into index terms that occur in the index vocabulary, which is suitable for input to the query engine.
Once the search engine has found the results to a query they are to be shown to user in an order of decreasing relevance, this order is decided by the search engines ranking algorithm .Different search engines use different ranking algorithm and they keep on updating, altering and imroving these algorithms on a daily basis.

3.2.1 Web Mining

“Web mining is a Data Mining approach that identifies or extracts information from web resources automatically”[5]. It's the process of gleaning fascinating and potentially beneficial patterns and information from World-related behaviour.

3.2.1.1 Web Mining Process

The following tasks make up the entire process of extracting information from online data[5]:
1. Finding resources: This entails retrieving desired online content.
2. Information pre-processing and selection: This entails the automated selection and pre-processing of certain information from retrieved online resources.
3. Generalization: It finds broad patterns on particular websites as well as across several sites.
4. Analysis: Validation and interpretation of the mined patterns are part of this step. Humans play a vital part in the knowledge discovery process on the internet.

3.2.1.2 Web Mining Categories-

Depending on the web data used input in Web data mining there are three categories of web mining[6]-

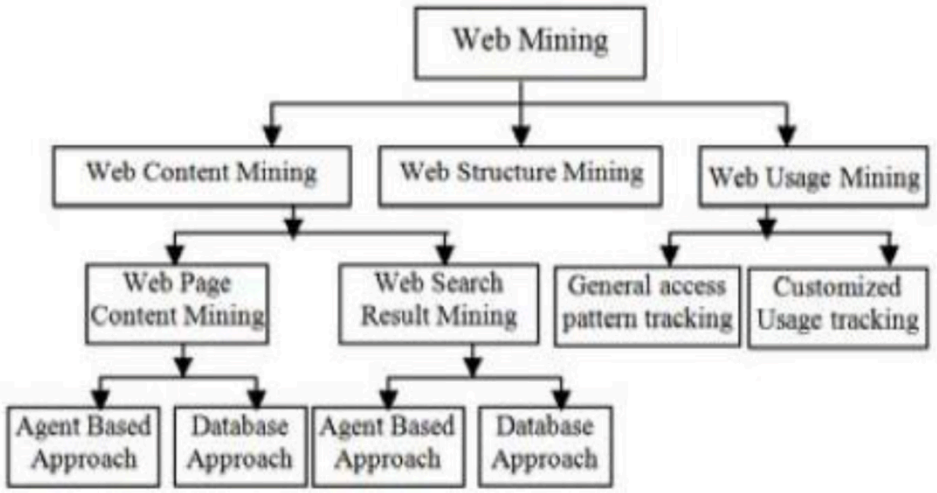


Fig-3.Classification of Web Mining[6]

A.Web Content Mining

It is the process of taking material from a Web document and organising it into more organised formats so that it can be discovered quickly. It is largely concerned with the internal structure of a document, or the level of the inner document. Web Content Mining is connected to Data Mining since many Data Mining techniques may be applied to it. Although it is similar to text mining in that much of the online material is text, it differs in that web data is largely semi-structured, whereas text mining focuses on unstructured text.

B.Web Structure Mining

It is the procedure for determining the link structure model of web pages. Using hyperlink topology, we catalogue the linkages and create information such as similarity and relationships between them. The term "online structure mining" refers to the process of extracting a structured summary of a website or web page from the internet. This area also includes Page Rank and hyperlink analysis. It tries to figure out the inter-document link structure of hyper links. Because online papers frequently contain connections and employ both actual and primary data on the internet, it is reasonable to deduce that Web Structure Mining and Web Content Mining are related.It analyses and describes HTML using a tree-like structure (Hyper Text Markup Language).

C.Web Usage Mining

It's a technique for detecting browsing patterns by analysing a user's navigational activities. It focuses on strategies for predicting user behaviour when they interact with the web. It makes advantage of web-based secondary data. This activity includes the automatic detection of user access patterns from one or more web servers. Using this mining method, we can figure out what people are looking for on the Internet. The approach is broken down into three steps: preprocessing, pattern finding, and pattern analysis. Data regarding Web usage may be easily captured by web servers, proxies, and client apps.

4. Link Based Analysis

Web mining gives more information by linking diverse publications together via hyperlinks.The web may be seen as a directed labelled graph, with nodes corresponding to articles or pages and edges corresponding to hyperlinks. Web graph is the name for this type of directed graph structure. A variety of methods based on link analysis have been proposed. Page Rank, Weighted PageRank, and HITS are three key algorithms detailed here.

4.1 Page Rank Algorithm

To date the most famous and widely used conventional ranking algorithm is google's Page Rank Algorithm and so it is crucial to briefly discuss its working in this paper.

The worth of a webpage or website is mostly subjective, as it is decided by the reader's interests, knowledge, and perspectives. However, there is still a lot to say objectively about the relative

relevance of Web sites. “A Web page’s “Page rank” is an objective measure of its citation importance that corresponds well with people’s subjective idea of importance . Page Rank is a great approach to rank the results of Web Key-word searches becaus of this correlation”[7].
The PageRank algorithm provides a significance rating to each page, which is a recursively defined measure that indicates how important a page is when other important pages link to it. This concept is recursive since the significance of a page is tied to the importance of other sites that link to it. Consider a random web surfer that follows links from page to page to understand PageRank. A page's page rank indicates the possibility of a random surfer landing on that page. The surfer is more likely to end up on the important pages since they have more links.

Calculation of Page Rank[7] **-**

- An assumption is made such that a page A has pages T1….Tn which point to it (i.e.are citations).
- ‘d’ is a parameter known as damping factor which can be valued between 0 and 1. The value of d is usually set to 0.85.
- C(A) is described as the number of outward links on page A.
- Page Rank of page A is given as follows:
- PR(A)=(1-d)+d{PR(T1)/C(T1)+...+PR(Tn)/C(T1n)}
- An important detail to be noted is that Page ranks form a probability distribution over web pages, so sum of page rank’s of all indexed web pages will be 1.

4.2 Weighted Page Rank

Instead of evenly dividing a page's rank value across its outlink sites, the Weighted Page Rank (Extended Page Rank algorithm) gives more significant pages a higher rank value. “Wenpu Xing and Ali Ghorbani presented this approach as an expansion of the PageRank algorithm”[6]. Rather than sharing the rank values evenly, this algorithm gives them to pages based on their relevance. Incoming and outgoing connections are given weight values to determine their relevance.
“Win (m, n) and Wout(m,n) are the two terms for this. The weight of link(m,n) as provided in is Win(m, n).The number of incoming links to page n and the number of incoming links to all reference pages on page m are used to compute it”[6].

$$W^{in}_{(m,n)} = \frac{I_n}{\sum_{p \in R(m)} I_p}$$

[6]

The number of incoming links on page n is In, the number of incoming links on page p is Ip, and the reference page list on page m is R. (m). Wout(m,n) provides the weight of link(m,n). It's based on the number of incoming links on all of page m's reference pages and the number of outgoing links on page n.[6].

$$W^{out}_{(m,n)} = \frac{O_n}{\sum_{p \in R(m)} O_p}$$

[6]

There are a specific number of outgoing connections on page n. op refers to the number of outbound links on page p. The below mentioned formula is then used to calculate the weighted PageRank.

$$WPR(n) = (1-d) + d \sum_{m \in B(n)} WPR(m) W^{in}_{(m,n)} W^{out}_{(m,n)}$$

[6]

4.3 HITS

Hyperlink-Induced Topic Search (HITS) (also known as Hubs and Authorities) is a link analysis tool developed by Jon Kleinberg that ranks Web sites based on their links. It was PageRank's precursor. Hubs and Authorities was inspired by a unique insight into the early days of the Internet's web page

creation: certain web pages, known as hubs, served as large directories that were not actually authoritative in the information they contained, but rather served as compilations of a broad catalogue of information that directed users to other authoritative pages. To put it another way, a good hub was a page that linked to many other pages, and a good authority was a page that was linked by many distinct hubs[8].

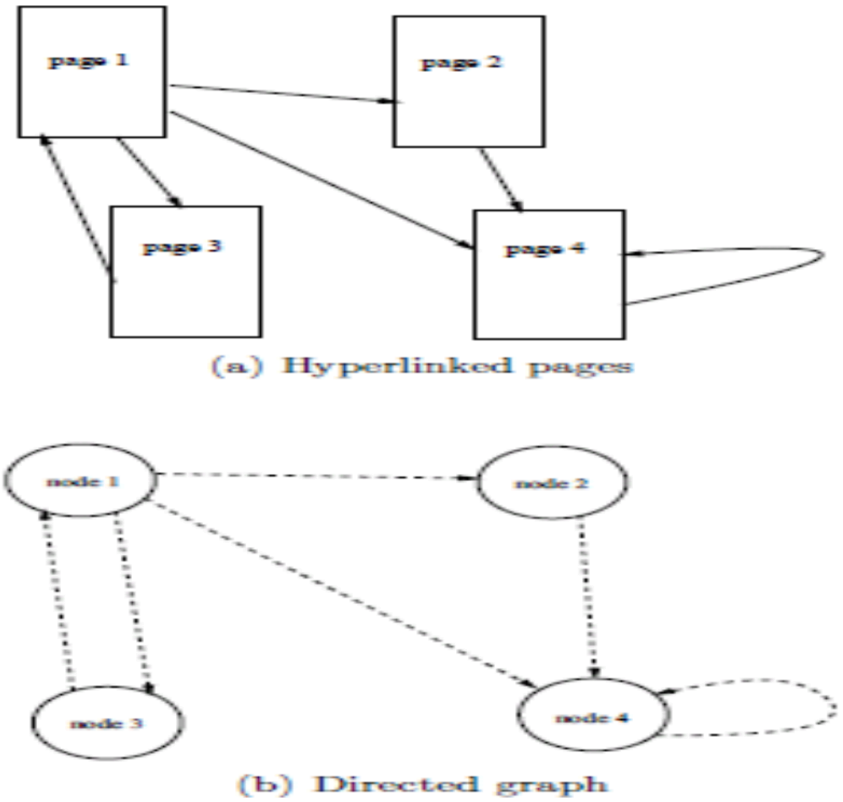


Fig-4. Hyperlinked Pages Modelled as Directed Graph[6]

As a consequence, each page is given two scores: authority, which determines the importance of the page's content, and hub value, which determines the importance of the page's links to other pages. A page can serve as a hub and authority at the same time. The WWW is interpreted by the HITS algorithm as a directed graph G(V,E), where V represents pages and E represents connections. Through examination of a relevant subgraph of the web, attempts are made to computationally discover hubs and authority on a certain issue.

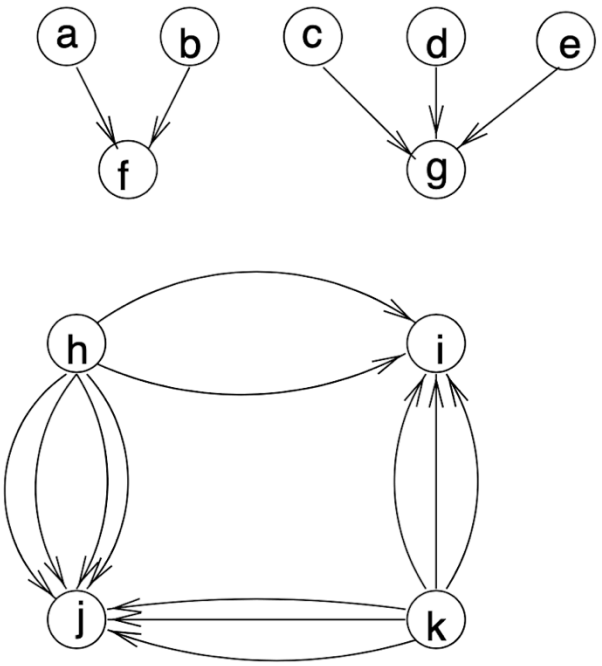


Fig-5. A Pictorial Representation of Hubs and authorities[9]

Hubs point to a variety of authority based on mutually recursive facts. Authorities are referenced by a vast number of hubs. A well-designed and entertaining website with a large number of hyperlinks pointing to it.. Authorities and hubs support each other. A hub is a site that serves as a gateway to numerous authoritative pages. Many good hubs link to a reliable authority. Many good authority are linked from a good hub.

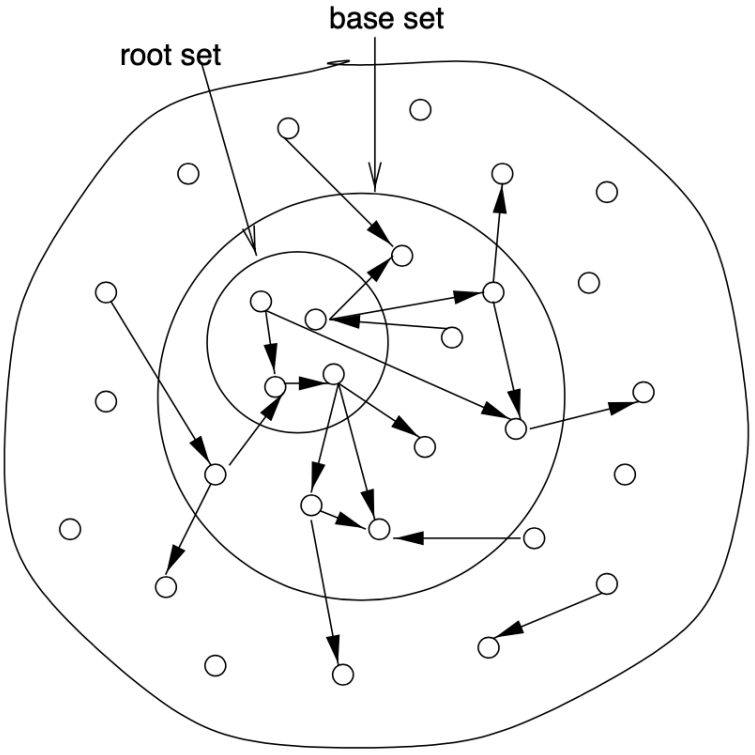


Fig-6.A pictorial Representation of HITS[9]

5. IR Models

An IR model specifies the complexities of document representation, query representation, and retrieval capabilities. The most fundamental IR models include Boolean, vector, probabilistic, and inference network models. The rest of this section provides a quick overview of various models.

5.1 Boolean Model

Although the Boolean model was the first information retrieval paradigm, it is also the most divisive. A query word may be thought of as a clear description of a set of documents. For example, the query term economic defines the collection of all texts indexed with the term economic. Query phrases and their related sets of documents can be concatenated using George Boole's mathematical logic operators to produce new sets of documents.The Boolean model allows you to utilise Boolean algebra operators like AND, OR, and NOT in your query, but it has one major flaw: it can't rank the items returned[10]. In the Boolean model, each document is linked to a set of keywords. To separate words in queries, AND, OR, and NOT/BUT are also utilised. The retrieval function of this approach sorts texts into relevant and irrelevant categories[11].

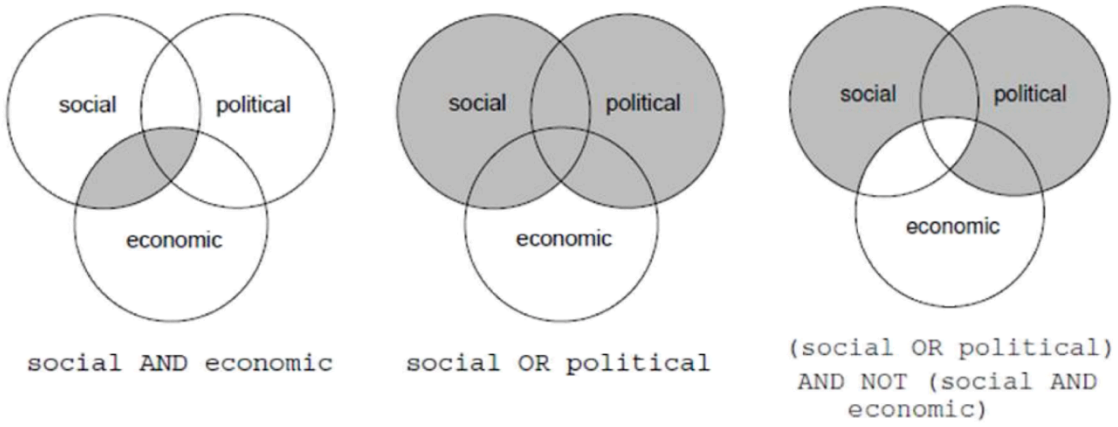


Fig 7.Venn diagrams depicting Boolean Set Combinations[1]

5.2 Vector Space Model

Based on Luhn's similarity criteria, Gerard Salton and colleagues suggested a more theoretically grounded approach. The query and index representations were seen as vectors embedded in a high-dimensional Euclidean space, each term having its own dimension. The vector space approach is most known for its effort to rank pages based on how similar the query and each document are[12]. The angle between documents and queries is determined using the similarity cosine function in the Vector Space Model (VSM). The cosine function of similarity is defined as follows:

$$sim(d_j, q) = \frac{d_j \cdot q}{\|d_j\| \cdot \|q\|} = \frac{\sum_{i=1}^{N} w_{i,j} w_{i,q}}{\sqrt{\sum_{i=1}^{N} w^2{}_{i,j}} \sqrt{\sum_{i=1}^{N} w^2{}_{i,q}}}$$

where:

$$d_j = (w_{1,j}, w_{2,j}, \ldots, w_{t,j})$$

$$q = (w_{1,q}, w_{2,q}, \ldots, w_{t,q})$$

The greater the angle between two vectors the smaller their cosine similarity will be and the lesser will be the chances of a match.The tf-idf weighting system has been introduced into the Vector Space Model.It has a term frequency factor (tf) that measures the frequency of occurrence of a word in a document or query text, as well as an inverse document frequency factor (idf) that measures the inverse of the number of documents that comprise a query or document phrase[10].

5.3 Probabilistic Model

The notion of ranking based on the probability of relevance was created by Maron and Kuhns, but it was Stephen Robertson who made it a principle.William Cooper is credited with inventing the probability ranking idea, according to him. The most important characteristic of the probabilistic model is its attempt to rank objects based on their likelihood of relevance in response to a query. Documents and queries are represented by binary vectors d and q, with each vector element signifying whether or not a document attribute or word exists in the document or query. The probabilistic approach uses odds O(R) instead of probabilities, where O(R) = P(R)/1-P(R), R signifies a relevant document and R denotes a non-relevant document[10].

5.4 Inference Network Model

In this paradigm, document retrieval is portrayed as a process of inference in an inference network. This method can be used to implement the majority of IR techniques. In the most basic version of this method, a document defines a word with a defined strength, and the credit from several terms is gathered using a query to calculate the content's equivalent of a numeric score[1]. From an operational viewpoint, the strength of instantiation of a word for a document may be regarded the weight of the term in the document, and document ranking in this model's simplest version resembles document ranking in the vector space model and the probabilistic models discussed above. Any formulation can be used because the model does not specify the strength of instantiation of a word for a text.

6. Neural Networks for Information Retrieval

Information retrieval(IR) research revolves around ranking models. Various ways for creating ranking models have been developed throughout the years, ranging from old heuristic methods to probabilistic methods to recent machine learning methods. Because of the advent of deep learning technology, we have lately seen a growing body of work in applying shallow or deep neural networks to the ranking issue in IR, referred to as neural ranking models in this study. The ability of neural ranking models to learn from raw text inputs allows them to avoid many of the limitations of handcrafted features. Neural networks have the potential to represent complex tasks, which is necessary for dealing with the complexities of ranking relevance estimation. Here we will look into the fundamental assumptions, main design concepts, and learning methodologies of neural ranking models from many aspects in this survey.

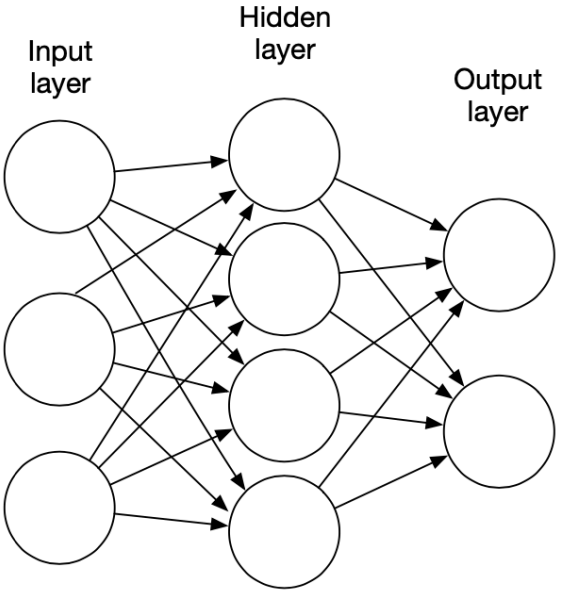


Fig-8.Feed-forward fully connected neural network[13]

7. Deep Learning Techniques in Neural Networks-

This section introduces the most often used deep learning terms and techniques in ad-hoc retrieval.When addressing the various neural ranking models later on in the literature, we will refer to these neural components and approaches.

7.1 Convolutional Neural Network(CNN)

By establishing a series of filters or kernels that spatially link small areas, a CNN retrieves features from data.Unlike dense networks, each neuron is connected to a small number of neurons rather than all of the neurons from the previous layer. With this architecture, the model's number of parameters is drastically decreased. Furthermore, the input weights of CNN filters are shared across several local regions, lowering the number of parameters even more[14]. Feature maps are the outputs of the CNN. To save just the most significant signals and reduce dimensionality, the feature map is commonly subjected to pooling processes such as average and max pooling. Padding is used to prolong the input in order to handle information near the edge because a CNN kernel has a set size. Image-related tasks, such as image classification, were the first tasks for which CNNs were utilised[15].

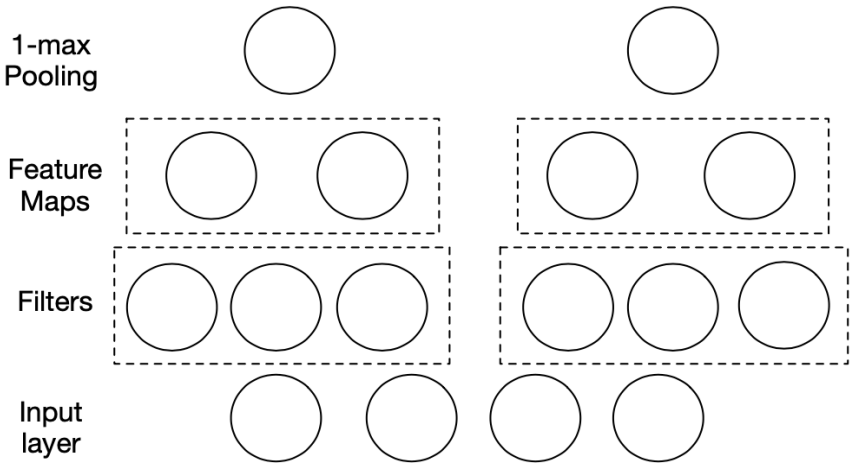


Fig-9.Only one convolutional layer in a one-dimensional CNN, followed by a one-max pooling layer[13]

7.2 Recurrent Neural Networks(RNN)

An RNN learns features and long-term dependencies from sequential and time-series data. RNN creates a hidden state in each timestamp by reading the input sequence sequentially. These concealed states can be compared to memory cells that store sequence information. The current hidden state is determined by the previous hidden state and the current input. As a result, for each timestamp, a hidden state is generated, with the hidden state corresponding to the sequence's last timestamp encapsulating the sequence's context-aware representation. The vanilla RNN has two major flaws: vanishing and growing gradients[16] during back-propagation during the training phase. When the gradient flows from later to earlier timestamps in the input sequence, for example, the gradient's signal may become very feeble or even vanish for long periods of time. LSTM and GRU are two RNN variants that have been proposed to better capture long-term dependencies than RNN and hence reduce gradient vanishing and explosion. The network may now capture long-range interactions using the new LSTM and GRU components[17].

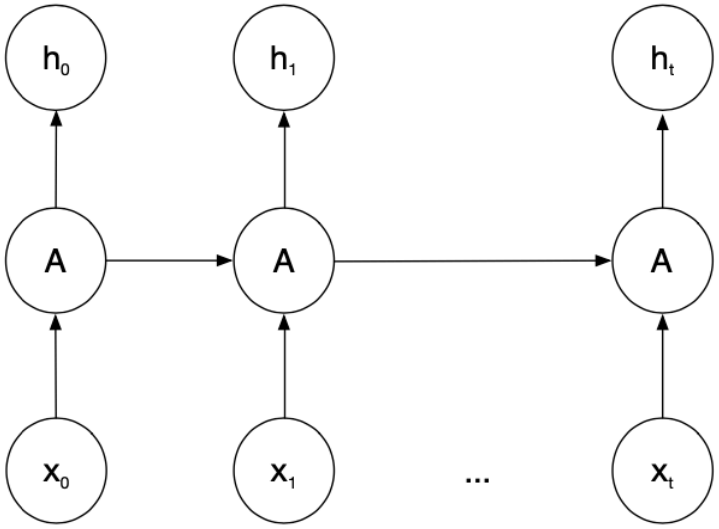


Fig-10.RNN representation, with x representing the input, A representing the shared processing unit across time steps, and h representing the hidden state vector[13].

7.3 Long Short-Term Memory(LSTM)

During the training phase, the network fails to learn long-term relationships in the input data due to exploding and disappearing gradients. To counteract the impacts of bursting and disappearing gradients, LSTM[18] was developed. The LSTM's memory cell structure differs from that of a traditional RNN in that three gates are used to manage the data in the memory cell. The input gate, for starters, controls how the memory cell responds to the current input. Secondly the forget gate determines which prior data should be erased from the present timestamp. At last the output gate regulates the memory cell's effect on the current timestamp's concealed state. In comparison to RNN, LSTM has resulted in considerable advances in a variety of disciplines using sequential data, including text, video, and audio. Language modelling, text classification, machine translation , video analysis , picture captioning and speech recognition have all been solved using LSTM [19].

7.4 Gated Recurrent Units(GRU)

GRU [20] is a sequence-based task that captures long-term dependencies, similar to LSTM. GRU, on the other hand, does not contain discrete memory cells like LSTM. The memory content used by other units is regulated by the output gate in an LSTM network. In contrast, the GRU model lacks an output gate and hence uses its content without any gating control. Furthermore, unlike the LSTM, which computes the value of the new added memory independently of the forget gate, the GRU does not control the new added activation independently of the reset gate, instead depending on the reset gate

to handle the previously hidden state. This model has proved to perform well in a variety of tasks, including machine translation[20] and sentiment categorization .

7.5 Attention Mechanism

The attention mechanism for neural machine translation was introduced for the first time in 2015 by D. Bahdanau[21]. In the original Seq2Seq model, Sutskever(2014) [22] used one LSTM to encode a sentence from its source language and another LSTM to decode the phrase into a target language. Long-term dependencies, on the other hand, were missed by this strategy. Bahdanau et al., (2015) advocated learning to align and translate the text at the same time to tackle this challenge. They learn attention weights while predicting a target word, which may be used to generate context vectors that focus on a set of locations in a source phrase. The attention vector is computed using a weighted sum of all the hidden states of an input sequence, with each attention weight representing the significance of a token from the source sequence in the attention vector of a token from the output sequence. Even though the attention mechanism was first applied in machine translation, it has had range of other applications including document retrieval[23], document classification, sentiment classification, recommender systems, speech recognition[24], and visual question answering[17].

7.6 Word Embedding

The method "word embedding" refers to a method of encoding a word's meaning using other words.. The identification of words which are employed in similar settings to a given phrase is possible thanks to the embedding of word vectors. Although word embedding has captivated the attention of natural language processing (NLP) scholars in the past few years, the methodologies' potential for use in information retrieval(IR) has received little attention.

Word2vec, a word embedding technique, has attracted the attentions of natural language processing (NLP) experts in recent years. The embedding of the word vectors assists in the discovery of a series of terms that have been used in similar settings to a given phrase. This section is about using word embedding to increase retrieval efficacy.

Word embedding approaches attempt to embed word representations. Two vectors v and v′, corresponding to the words t and t′, are near one other in a N - dimensional space of if their contexts are comparable, and vice versa (i.e. Similar expressions can be found in both settings)[25]. A cosine similarity metric may be used to this generic vector space of integrated words to identify a list of words that are used in similar scenarios to a given phrase. These semantically similar terms can be utilised for a variety of NLP applications. The fundamental concept is to train moving windows using word vector embeddings (instead of the more common word count vectors) and then categorise the individual windows. This might be used for things like POS tagging, named-entity identification, semantic role labelling and more. In cutting-edge word embedding systems, negative sampling is utilised to train deep neural networks. This negative sampling method (sometimes referred to as word2vec1) is believed to create reliable word embeddings in a highly efficient manner [26].

Why use Word Embedding in IR?

Now we'll look at how word embeddings can help with retrieval quality. In the context of IR, vocabulary mismatch is described as the employment of separate but semantically equivalent terms across papers on the same subject. The vector space model (VSM) assumes that texts are stored in a two - dimensional term space perpendicular to each other, whereas probabilistic models like the BM25 or the language model (LM) presume that words are sampled independently.

Standard IR approaches evaluate word association in two ways: one examines locally present information of words in the top rated documents acquired in response to a query and the other considers a global analysis of the entire set of documents (i.e. irrespective of the queries). Current global analysis techniques, like latent Dirichlet allocation (LDA) or latent semantic indexing (LSI), only look at term co-occurrences at the document level, rather understanding the context of a phrase. We believe that because the word embedding techniques we introduced at the beginning of this section use information about each word's local context to derive embeddings (If and only if two words are used in comparable situations, they have similar vector representations), Such a method has the potential to improve IR's global analysis strategy, resulting in more effective retrieval[27].

7.7 Deep Contextualized Language Models

Peters et al. proposed ELMo, a deep contextualised language model made up of backward and forward LSTMs (2018)[28]. To compute the embedding of a single token, ELMo uses job determined learnable weights to linearly combine representations from several layers of both backward and forward LSTMs. To produce deeper embeddings, internal states are integrated. Although ELMo improves the results of numerous NLP tasks, it does so by decoupling the left-to-right and right-to-left contexts via a shallow combination of internal states from individually trained forward and backward LSTMs. Devlin et al. (2018) proposed a language model called Bidirectional Encoder Representations from Transformers (BERT) that combines left and right contexts[29].

BERT is a multilayer Transformative language model with a deep contextualised language model. Each block has a multi-head self-attention structure followed by a feed-forward network that generates contextualised embeddings for each character from the input. BERT is trained utilising two pre-training tasks: next sentence prediction and masked language model on large volumes of unlabelled data. After the pre-training phase, BERT may be utilised for regression tasks on single texts or text pairs using special tokens ([SEP] and [CLS]) added to the input. Document retrieval, passage re-ranking, frequently asked question retrieval, table retrieval, and semantic labelling are all tasks that employ the sentence pair classification setting. Text categorization is done with the single sentence setting. The last hidden state hθ of the initial token [CLS] is used by BERT to represent the entire input sequence, where θ denotes the BERT parameters. Then, on top of BERT, a basic softmax layer with W parameters is built to forecast the likelihood of a given label l:
$p(l|h\theta)=softmax(Wh\theta)$[17],The BERT parameters, given by θ, and the softmax layer parameters W are fine-tuned by maximising the log-probability of the true label.

7.8 Knowledge Graphs

Large-scale broad domain knowledge bases (KBs) like Freebase and DBpedia contain rich semantics that may be used to increase the performance of a range of natural language processing and information retrieval tasks. Human knowledge about things, classes, relations, and descriptions is stored in knowledge bases. The notion of knowledge graphs arises from the fact that knowledge may be represented as graphs (KG). With the purpose of mapping entities and relations into a latent space, many algorithms for representation learning of knowledge graphs have been proposed.
The most representative translation-based model, TransE[30], is influenced by Word2Vec[25], and examines the translation operation between head and tail entities for relations. TransE variants like TransR and TransH use a similar method, but they learn the embeddings using different scoring systems. RDF2Vec extends the Word2Vec approach to RDF graphs learning embeddings for entities.

To perform many jobs, researchers have recently investigated a novel route termed graph neural networks . Graph neural networks employ graph embeddings to describe the geometric features of a graph in short feature vectors and message passing to capture complicated relationship patterns between nodes in a graph.By collecting high order neighbourhood information, the Graph Convolutional Network (GCN) can create interpretations of nodes in a graph. The success of GNNs has sparked research on a wide range of areas.

8. A Unified Model Formulation

The LTR (learning to rank) framework is used to study neural ranking algorithms. In this part, we provide a coherent definition of neural ranking models based on a more comprehensive understanding of LTR problems. Assume S is the generic query set, which may include natural language inquiries or search queries and T is the generalised document set, which may include documents, answers, or responses[29]. Consider the label set y={1,2,...l} where labels indicate grades. Between the grades l>l-1>....1, there is a total order, where > represents the order connection[29].
Let $si \in S$ be the ith query, $Ti=\{ti1,ti2,....,tini\} \in T$ be the set of documents associated with the query si, and yi={yi1, yi2,..., yini} be the set of labels associated with query si, where ni denotes the size of Ti and yi and yij denotes the relevance degree of tij with respect to si. Let F be the function class and $f(si,tij) \in F$ be a ranking function which associates a relevance score with a query-document pair[29]. Let L(f;si,tij,yij) be the loss function defined on prediction of f over the query document pair and their

corresponding label. So a generalized LTR problem is to find the optimal ranking function f* by minimizing the loss function over some labelled dataset
f*=arg minΣiΣjL(f;si,tij, yij)- Ⅰ
Without loss of generality, the ranking function f could be further abstracted by the following unified formulation [31]
f(s,t)=g(Ψ(s),ϕ(t),η(s,t))- Ⅱ
Where s and t are two input texts, Ψ, ϕ are representation functions which extract features from s and t respectively, η is the interaction function which extracts features from (s,t) pair and g is the evaluation function which computes the relevance score based on the feature representations[31].

Functions Ψ, ϕ and η are typically set to be static functions in classic LTR techniques (i.e., manually defined feature functions). Any machine learning model that can be learned from the training data, such as logistic regression or gradient boosting decision tree, can be used as the evaluation function g. In most situations, all of the functions Ψ, ϕ, η and g are contained in the network architecture of neural ranking models so that they can all be learnt from training data. The inputs s and t in typical LTR techniques are usually raw texts. Raw texts or word embeddings can be used as inputs to neural ranking algorithms. To put it another way, embedding mapping is a simple input layer that is not included Ψ, ϕ and η[31].

9. Neural Ranking Model Architecture Types

We evaluate current neural ranking model architectures based on the aforementioned unified formulation to better grasp their core assumptions and design principles.

9.1 Symmetric vs Asymmetric Architectures

In neural ranking models, two main designs emerge: symmetric architecture and asymmetric architecture, which are based on different central assumptions about the input texts s and t, respectively..

9.1.1 Symmetric Architecture

The inputs s and t are expected to be homogenous in order to apply a symmetric network topology to them. In the symmetric structure the inputs s and t can switch position and still not affect the final output. In particular, symmetric interaction and siamese networks networks are two representative symmetric architectures.

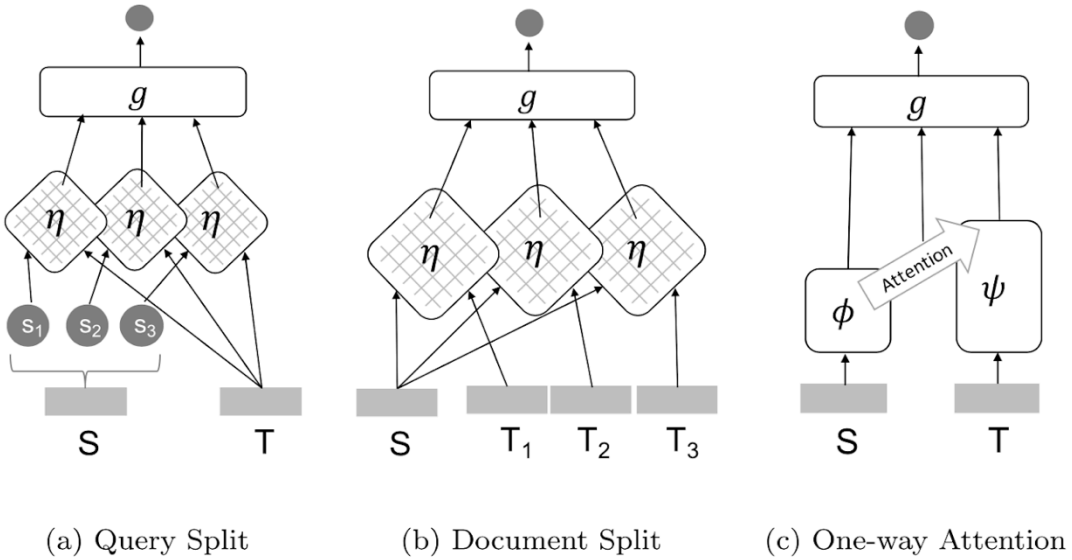
The term "Siamese network" literally means "symmetric network architecture." DSSM , CLSM, and LSTM-RNN are examples of representative models. For eg- DSSM shows 2 input texts though an integrated process that involves MLP transformation preceded by Letter-trigram mapping, i.e., function ϕ is the equal to Ψ.A cosine similarity function is then used to evaluate the similarity among the two representations, indicating that function g is symmetric. CLSM[32] captures local word order information by replacing the representation functions Ψ and ϕ with two alike convolutional neural networks (CNNs).For the purpose of capturing the long term reliance between words, LSTM-RNN replaces Ψ and ϕ with 2 alike long short-term memory (LSTM) networks.

A symmetric interaction function is used to describe inputs in case of symmetric networks. The final relevance score is then calculated using many convolutional and max-pooling layers, which is also symmetric across s and t. A symmetric interaction function is built by Match pyramid for every word pair in between s through t for capturing fine grained signals. The relevance score is then calculated using a symmetric evaluation function g, which consists of many 2D CNNs and a dynamic pooling layer[31].

9.1.2 Asymmetric Architecture

Asymmetric network architectures are used when input s and t are expected to be diverse. In this case we will totally unalike results if we put inputs s and t in the input layer. As seen in ad-hoc retrieval

the document and the query are innately heterogenous, asymmetric designs have been developed primarily in the ad-hoc retrieval job.



(a) Query Split (b) Document Split (c) One-way Attention

Fig-11.Three types of Asymmetric Architecture[31]

The asymmetric design is examined using the ad-hoc retrieval situation as an example. To address the heterogenous nature of the query and the document, the asymmetric architecture employs three key procedures: query split, document split, and joint split.

A. Query Split

The idea behind query split is that most ad-hoc retrieval requests are keyword-based, therefore we may break the query into words to match against the document, as shown in Fig. 11.(a). DRMM is a common model formed on this method[33]. “The interaction function is defined as the matching histogram mapping between each query term and the document by DRMM”[33], which breaks the query into terms. The evaluation function g is made up of a feed-forward network for term-level relevance calculation and a gating network for score accumulation. In terms of the query and the document, this procedure is obviously lopsided.

B. Document Split

The scope hypothesis states that a large text may only be partly relational to a query, therefore instead of dealing with the document as a whole we split it to get fine-grained interaction signals, as shown in Fig. 11.(b). HiNT is a good example of a model based on this method[34]. The document is initially divided into sections in HiNT using a sliding window. Interaction function is accurate matching between question and each segment of the documents. The evaluation function g includes the global decision layers and local matching layers[31].

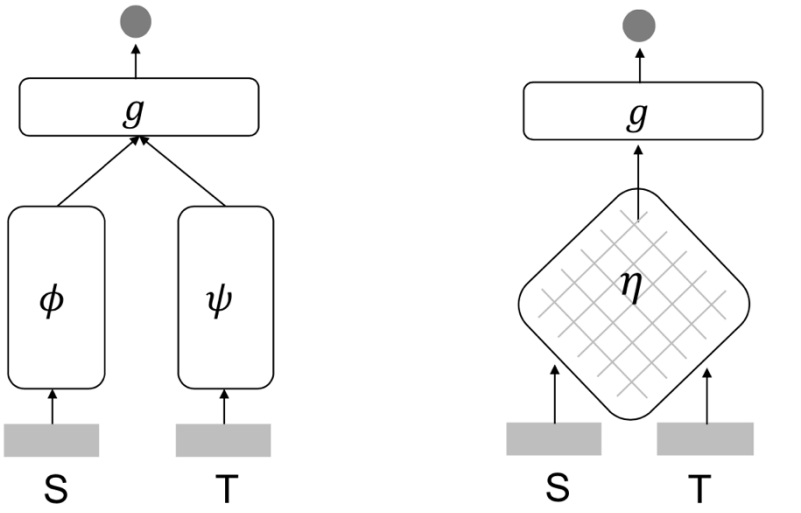
C. Joint Split

Joint split, as the name implies, employs both query and document split assumptions. DeepRank is a common model based on this method[35]. With regard to each query word, DeepRank divides the document into term-centric contexts. The query and term-centric contexts' interaction function is then described in a variety of ways.. Term-level computation, term-level aggregation, and global aggregation are the three aspects of the evaluation function g[31].

There is another another prominent method driving the asymmetric architecture in neural ranking models used for quality assurance. As shown in Fig. 11(c), “In order to improve the response representation, the one-way attention mechanism employs the question representation to acquire attention over potential answer terms”[31].

9.2 Representation focused vs interaction focused architectures

The present neural Ranking Models can also be categorised in two forms namely representation focused and interaction focused architecture, on the bases of variety of assumptions about the features (derived by the representation function Ψ, ϕ or the interaction function η) for the purpose of relevance evaluation, as shown in Fig. 12. Aside from the two above mentioned core categories, some neural ranking models use a hybrid approach to learn relevant characteristics, combining the benefits of both architectures.



(a) Representation-focused (b) Interaction-focused

Fig-12.Representaion Focused and Interaction focused architectures[31]

9.2.1 Representation focused architecture

The essential premise of this design is that relevance is determined by the input texts compositional meaning. Various deep network topologies, including fully-connected networks, convolutional networks, and recurrent networks, have been used for Ψ and ϕ[31].In this architecture relevance is evaluated on the basis of high level representation of each word[33]. This design is also better for activities that need brief input messages (because high-level representations of large texts are typically difficult to get by). Furthermore, once the model has learned Ψ and ϕ it is able to calculate representations of the text in advance which make it useful for applying in online computation.

9.2.2 Interaction-focused Architecture

The core idea of this design is that relevance is really about the relationship among the input text, therefore learning directly from interactions rather than isolated representations would be more successful. Here we describe the interaction function η instead of the representation functions Ψ and ϕ, and compute the relevance score using a sophisticated evaluation function g. Interaction functions are of two types parametric and non-parametric.

The interaction-focused design can match most IR tasks in general since it evaluates relevance solely on the basis of interactions. Because it avoids the complexity of encoding large texts, this design is also more suited to jobs involving diverse inputs, such as ad-hoc retrieval and quality assurance. Unlike the representation focused model in this architecture the interaction function η cannot be calculated before the input pair is obtained so this architecture is not suitable for online computation. As a result, a better method to employ these two types of models in practise is to use them in a "telescope" setup, where representation-focused models are used early in the search process and interaction-focused models are used afterwards[31].

9.2.3 Hybrid Architecture

In order to use both Representation and interaction focused architectures for feature learning the Hybrid architecture is developed. To merge the two architectures, we discovered two important hybrid strategies: combined strategy and linked strategy.

- **Combined Strategy-** It is a loose hybrid strategy uses both representation and interaction focused architectures as sub-models and then sums their result to estimate the ultimate relevance[36].
- **Coupled Strategy -** It is a compact Hybrid strategy which involves learning representations while paying attention to both inputs. As a result, both the representation and interaction functions are compactly integrated. IARNN and CompAgg are examples of models that use this method[37].

9.3 Single granularity vs Multi-granularity architecture

There are two types of neural ranking models now available, each based on different assumptions regarding the relevance estimation process: single-granularity models and multi-granularity models[31].

9.3.1 Single Granularity Architecture

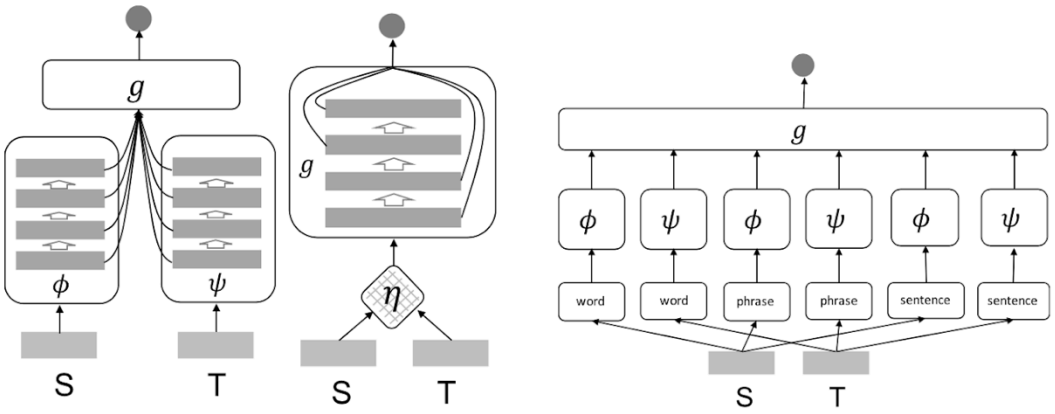
The single-granularity architecture is based on the notion that relevance may be assessed using the top-level characteristics derived by Ψ, ϕ and η from text inputs. Ψ, ϕ and η are assumed as black boxes in the evaluation process of g. Here g solely uses their final outputs to compute significance. DSSM and DRMM are examples of neural ranking models that fall into this category.

9.3.2 Multi-Granularity Architecture

The multi-granularity architecture is founded on the idea that relevance calculation requires a lot of aggregation of features, either from various levels of feature abstraction or from distinct types of input language units. Ψ, ϕ and η are no longer black-boxes for g under this assumption[31], and we analyse the linguistic structures in s and t. As shown in Fig. 13, there are two main forms of multi-granularity: vertical multi-granularity and horizontal multi-granularity.

- **Vertical Multi-Granularity**
  This uses deep networks hierarchical nature to allow the evaluation function g to use several levels of feature abstraction for relevance estimation. In order to encode input text the representation functions Ψ and ϕ are defined as two CNN networks respectively[38], and relevance estimation is done by g using each layers output.

- **Horizontal Multi-granularity**
  Horizontal multi-granularity assumes that language contains inherent structures (e.g., words or sentences), & we will use multiple sorts of language units as inputs for improved relevance estimate rather than simple words. For instance in huang et al(2017)[39] the character-level, word-level, and sentence-level representations of the inputs are obtained using a CNN and an LSTM, and each level representation is then worked on and accumulated to produce the ultimate relevance score by the evaluation function g[31].



(a) Vertical Multi-granularity (b) Horizontal Multi-granularity

Fig-13.Multi Granularity Architectures[31]

As can be seen in Fig-13, the multi-granularity design is a simple progression from the single-granularity architecture, which uses intrinsic language and network structures to improve relevance estimates.

10. Major Applications of Neural Ranking Models

We detail various significant IR applications in this part, including ad-hoc retrieval, question answering, community question answering, and automatic Conversation. Product search, sponsored search and so on are examples of applications where neural ranking models have been or might be used. However, owing to time constraints, these applications will not be included in this survey.

10.1 Ad-Hoc Retrieval

This is a traditional retrieval activity in which the user mentions his or her data needs in form of a query, which triggers a search for documents containing answers/data that is likely to be relevant to the user's query. The phrase "ad-hoc" is used to describe a situation in which the collection of documents remains relatively unchanged but fresh queries are made to the IR system on a regular basis. A number of recovered documents(or any form of data) is usually returned as a ranking list, with the papers at the top of the list being most relevant to the users query.

Ad-hoc retrieval has a lengthy study history, with various well-known characteristics and obstacles related with the job. The heterogenous nature of the query and the documents is a key feature of ad-hoc retrieval. The question is generally quite brief, ranging from a few words to a few phrases, and it originates from a search user with potentially confusing purpose[40]. The documents are usually written by several writers and have a greater text length, ranging from a few phrases to many paragraphs. Diverse relevance patterns result from such heterogeneity. The concept of relevance in ad-hoc retrieval is innately ambiguous and largely dependent on user's query which makes relevance evaluation a difficult task.

10.2 Question Answering

The Process of Question Answering is an attempt to autonomously respond to queries presented by a user in natural languages by an information retrieval system using its information sources. The inquiries might come from a closed or open domain, and the information resources could be organised (e.g., a knowledge base) or unstructured (e.g., documents or Web pages)[41] .However, some task layouts are not often viewed as IR issues. Multiple-choice selection, for example, is commonly presented as a classification issue. As a result, we will concentrate on response passage/sentence retrieval in this survey since it can be expressed as a standard IR issue that can be solved using neural ranking models. To keep things simple, we'll call this task QA from now on.

QA reveals less variation between the query and the response passage/sentence than ad-hoc retrieval. Nevertheless the inquiry in this case is frequently asked in natural language, that is lengthier than keyword inquiries and provides a clearer explanation of the goal. Response passages/sentences, on the other hand, are often considerably shorter text spans than documents, resulting in more focused topics/semantics. However, word mismatch remains a common QA issue. In QA, the concept of relevance is rather straightforward, i.e., if the target phrase/segment relates to the inquiry, but evaluation is difficult.

10.3 Community Question Answering

Websites such as Quora, Stack Overflow, Yahoo! Solutions, etc are examples of Community Question answering(CQA) tools that strive to discover results to user's queries on the basis of existing Question Answer resources in CQA websites. CQA may be classified into two groups as a retrieval task. The first is to get responses directly from the existing answer list, and this is identical to the QA tool mentioned above but includes some extra user behavioural data (e.g., upvotes/downvotes)[42]. The second method is to extract similar question-answer pair from the existing question-answer collection, with the notion that answers to homogenous questions may provide answers to new ones. Unless otherwise stated, the second task format will be referred to as CQA.

CQA is distinct from the previous two tasks in that it requires retrieval of comparable questions. This is owing to the homogenous nature of the current and already answered questions. Both the currently posted and already answered questions are brief natural language phrases that describe user's information demands[43]. A wide range of data sets have been made available for study in order to evaluate the CQA job. "The Quora Dataset, Yahoo! Answers Dataset, and SemEval-2017 Task3 are all well-known data sets"[31].

10.4 Automatic Conversation

The goal of automatic conversation (AC) is to build an automated man-machine dialogue process for answering queries, social chat, etc[44]. AC can be phrased as an IR problem aimed at ranking/selecting a correct answer from a dialogue library, or as a generation problem aimed at generating an adequate response.

Because question answering is already addressed in the aforementioned QA assignment, we limit AC to the social chat function in this study. AC might further classified into single-turn conversation[45] or multi-turn conversation. In case of social conversation AC exhibits large similarity to CQA. This is because, both the input speech and the answer are brief natural language phrases. In AC, relevance refers to a broad concept of semantic correspondence (or coherent structure), for example, given the input utterance "OMG I acquired myopia at such an 'old' age," the answer might range from general (e.g., "Really?") to specialised (e.g., "Yeah. a pair of spectacles as a present")[31].As shown in the example, a satisfactory answer need not have semantic matching , hence vocabulary mismatch is no longer the fundamental difficulty in AC.

11. Trending Topics For Future Discussion

This section covers a number of hot issues in the field of neural ranking models. Most of these subjects are significant but have received little attention in this discipline, while others are extremely promising study paths.

11.1 Learning with External Knowledge

The majority of present neural ranking algorithms pay attention to the patterns that match the input text with a document. Some academics have went far above textual object matching in recent years to improve ranking performance by integrating external knowledge. There are two types of research projects: (1) external organised knowledge, such as knowledge bases; (2) external unorganised knowledge, like retrieved top results, themes, or tags This work will now be briefly discussed[31].

The first area of study looked on using semantic information from knowledge bases to improve neural ranking algorithms. "EDRM, suggested by Liu et al. (2018b), includes entities in interaction-focused neural ranking models"[46]. EDRM initially gains an understanding of the distributed representations of entities from knowledge bases in descriptions and types, based on their semantics[46]. In

conclusion, learning with external knowledge is a fresh topic in the field of neural ranking models. To increase the performance of neural ranking models using refined external knowledge and to grasp the function of external knowledge in ranking tasks, more research is needed[31].

11.2 Learning with visualized technology

In this study, we addressed a variety of neural ranking models in the context of textual IR. A lot of research has also shown that it is also possible to solve IR Problems visually. The basic concept is that we may use deep neural networks to evaluate relevance based on visual cues by constructing two comparable inputs as an image. When compared to typical matching matrices, the matching picture has the advantage of preserving the layout data of the original input, allowing numerous valuable variables such as geographical closeness, font size, and colours to be modelled for relevance estimation. Solving IR problems visually is especially beneficial when dealing with ad-hoc retrieval jobs on the Web, since sites are frequently well-structured documents with complex layouts.

As explained in [47], Visual learning TO Rank (ViTOR) is a dataset created by Akker, Markov, and de Rijke (2019) for the LTR problem with visual characteristics . Snapshots, visual characteristics, and relevance assessments for ClueWeb12 websites and TREC Web Track searches make up the ViTOR dataset[47]. Their findings showed that aesthetic characteristics can boost LTR performance greatly. In conclusion, using graphical technology to solve the textual ranking problem is a fresh and exciting area. This method resembles human behaviour in that we determine significance through visual perception as well. Visual elements have only been shown to be beneficial in specific relevance evaluation tasks in previous research. More study is required however, to determine what may be learnt by such graphical technologies beyond text-based approaches, as well as which IR problems could gain from such models[31].

11.3 Learning with Context

Many queries are too brief to accurately describe the underlying information requirements. One popular technique for addressing this problem is to use query context to increase retrieval performance. The literature has looked into many sorts of query contexts[31]:

- Short Term History: In the current search session, the user's previous interactions with the system.
- Long Term History: the user's query history, which is frequently utilised for online search personalisation.
- Situational Context: the aspects of the current query such as location and time, that are independent of the query content.
- Relevance Feedback: To increase retrieval performance, implicit, explicit, or pseudo relevance signals for a particular query can be employed as the query context.

Although query context has gained a lot of heed in the literature, adding it into neural ranking models has not been that common. Zamani et al. (2017) in [48] suggested a deep and wide network architecture, in which the deep half of the model learns abstract representations for contextual characteristics while the wide part employs raw contextual values in binary format to prevent information loss due to high-level abstraction. Later, as shown in [49] Li et al. (2018b) used a neural pseudo-relevance feedback technique termed NPRF to expand current neural ranking models, such as DRMM(deep relevance matching model) and KNRM(Kernel based neural ranking model).

In conclusion, when intuitive search systems arise, context-aware ranking will become an essential tool in this settings. In terms of how to include query context information into neural ranking algorithms, there are various open research problems. In the near future, more study in this area is predicted.

11.4 Neural Ranking Model Understanding

Deep learning has been criticised as a "black box" that provides good outcomes but issues no insights or clarification. As a result, both the Machine Learning and IR groups have focused on how to comprehend and explain neural models. To our knowledge, the core introduction and establishment of neural ranking models has not been substantially examined. Instead, a few publications have been published that analyse and comprehend the empirical influence of various model components in IR tasks[31].

For example, Pang et al (2016a) in [50] examined different pooling sizes, kernals, and similarity functions in terms of retrieval performance while using the MatchPyramid model in ad-hoc retrieval. Cohen, O'Connor, and Croft (2018c) in [51] retrieved neural ranking models' internal representations and assessed their efficacy in four natural language processing tasks. They discovered that contemporary relevance information is generally stored in a neural model's high-level layers. In that they even noted that low-level network layers capture more specific text information, whereas high-level layers abstract more subject information. Overall, the field of analysis of working of neural ranking models has remained relatively unexplored.

12. Conclusion

The goal of this study is to review the present state of neural ranking model research, examine existing approaches, and obtain some insight into future progress. A Brief overview of all kinds of knowledge required to understand neural networks including history of information retrieval techniques and current Deep learning techniques are also Discussed. Under the topic of model architecture and model learning, we developed a unified model for neural IR tasks and reviewed current models based on this formulation from several aspects. Neural ranking model research has sped and expanded in terms of application, similar to the advancement of several deep learning-based systems. We believe that by looking at prior triumphs and mistakes, this paper may assist academics who are interested in this topic and inspire new ideas. Neural ranking models are a part of the wider field of neural IR, that is a combination of deep learning with IR technologies which thus opens up a slew of new research and application opportunities. We anticipate substantial advancements in this sector in the near future as a result of community initiatives, comparable to those seen in computer vision and natural language processing.

References

[1] A. Roshdi and A. Roohparvar, “Information retrieval techniques and applications,” *International Journal of Computer Networks and Communications Security*, vol. 3, no. 9, pp. 373–377, 2015.

[2] R. Baeza-Yates, B. Ribeiro-Neto, and others, *Modern information retrieval*, vol. 463. ACM press New York, 1999.

[3] J. Pokorny, “Web searching and information retrieval,” *Computing in Science & Engineering*, vol. 6, no. 4, pp. 43–48, 2004.

[4] L. Andrade and M. J. Silva, “Indexing Structures for Geographic Web Retrieval,” 2006.

[5] K. Raymond and B. Hendrik, “Web mining research: A survey,” *SIGKDD Exploration, SIGKDD ACM*, vol. 2, no. 1, pp. 1–15, 2000.

[6] R. Prajapati, “A survey paper on hyperlink-induced topic search (HITS) algorithms for web mining,” *International Journal of Engineering*, vol. 1, no. 2, 2012.

[7] S. Brin and L. Page, “The anatomy of a large-scale hypertextual web search engine,” *Computer networks and ISDN systems*, vol. 30, no. 1–7, pp. 107–117, 1998.
[8] J. M. Kleinberg, R. Kumar, P. Raghavan, S. Rajagopalan, and A. S. Tomkins, “The web as a graph: Measurements, models, and methods,” in *International Computing and Combinatorics Conference*, 1999, pp. 1–17.
[9] M. Agosti and M. Melucci, “Information retrieval on the web,” in *European Summer School on Information Retrieval*, 2000, pp. 242–285.
[10] D. Hiemstra and A. P. de Vries, “Relating the new language models of information retrieval to the traditional retrieval models,” 2000.
[11] A. A. Alhenshiri, “Web information retrieval and search engines techniques,” *Al-Satil J*, pp. 55–92, 2010.
[12] H. Turtle and W. B. Croft, “Inference networks for document retrieval. In ‘SIGIR’90: Proceedings of the 13th annual international ACM SIGIR conference on Research and development in information retrieval.’” ACM Press, 1990.
[13] Y. Zhang *et al.*, “Neural information retrieval: A literature review,” *arXiv preprint arXiv:1611.06792*, 2016.
[14] P. Haffner, L. Bottou, P. G. Howard, P. Simard, Y. Bengio, and Y. le Cun, “Browsing through high quality document images with DjVu,” in *Proceedings IEEE International Forum on Research and Technology Advances in Digital Libraries-ADL’98-*, 1998, pp. 309–318.
[15] X. Liu, Z. Tang, and B. Yang, “Predicting network attacks with CNN by constructing images from NetFlow data,” in *2019 IEEE 5th Intl Conference on Big Data Security on Cloud (BigDataSecurity), IEEE Intl Conference on High Performance and Smart Computing,(HPSC) and IEEE Intl Conference on Intelligent Data and Security (IDS)*, 2019, pp. 61–66.
[16] R. Pascanu, T. Mikolov, and Y. Bengio, “On the difficulty of training recurrent neural networks,” in *International conference on machine learning*, 2013, pp. 1310–1318.
[17] M. Trabelsi, Z. Chen, B. D. Davison, and J. Heflin, “Neural ranking models for document retrieval,” *Information Retrieval Journal*, vol. 24, no. 6, pp. 400–444, 2021.
[18] S. Hochreiter and J. Schmidhuber, “Long short-term memory,” *Neural Comput*, vol. 9, no. 8, pp. 1735–1780, 1997.
[19] A. Graves, A. Mohamed, and G. Hinton, “Speech recognition with deep recurrent neural networks,” in *2013 IEEE international conference on acoustics, speech and signal processing*, 2013, pp. 6645–6649.
[20] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, “Empirical evaluation of gated recurrent neural networks on sequence modeling,” *arXiv preprint arXiv:1412.3555*, 2014.
[21] J. K. Chorowski, D. Bahdanau, D. Serdyuk, K. Cho, and Y. Bengio, “Attention-based models for speech recognition,” *Adv Neural Inf Process Syst*, vol. 28, 2015.
[22] I. Sutskever, O. Vinyals, and Q. v Le, “Sequence to sequence learning with neural networks,” *Adv Neural Inf Process Syst*, vol. 27, 2014.
[23] R. McDonald, G.-I. Brokos, and I. Androutsopoulos, “Deep relevance ranking using enhanced document-query interactions,” *arXiv preprint arXiv:1809.01682*, 2018.
[24] W. Chan, N. Jaitly, Q. Le, and O. Vinyals, “Listen, attend and spell: A neural network for large vocabulary conversational speech recognition,” in *2016 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, 2016, pp. 4960–4964.
[25] Y. Goldberg and O. Levy, “word2vec Explained: deriving Mikolov et al.’s negative-sampling word-embedding method,” *arXiv preprint arXiv:1402.3722*, 2014.
[26] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, “Distributed representations of words and phrases and their compositionality,” *Adv Neural Inf Process Syst*, vol. 26, 2013.
[27] D. Ganguly, D. Roy, M. Mitra, and G. J. F. Jones, “Word embedding based generalized language model for information retrieval,” in *Proceedings of the 38th international ACM SIGIR conference on research and development in information retrieval*, 2015, pp. 795–798.
[28] M. Gardner *et al.*, “Allennlp: A deep semantic natural language processing platform,” *arXiv preprint arXiv:1803.07640*, 2018.
[29] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, “Bert: Pre-training of deep bidirectional transformers for language understanding,” *arXiv preprint arXiv:1810.04805*, 2018.
[30] A. Bordes, N. Usunier, A. Garcia-Duran, J. Weston, and O. Yakhnenko, “Translating embeddings for modeling multi-relational data,” *Adv Neural Inf Process Syst*, vol. 26, 2013.
[31] J. Guo *et al.*, “A deep look into neural ranking models for information retrieval,” *Information Processing & Management*, vol. 57, no. 6, p. 102067, 2020.
[32] H. Palangi *et al.*, “Semantic modelling with long-short-term memory for information retrieval,” *arXiv preprint arXiv:1412.6629*, 2014.

[33] J. Guo, Y. Fan, Q. Ai, and W. B. Croft, “A deep relevance matching model for ad-hoc retrieval,” in *Proceedings of the 25th ACM international on conference on information and knowledge management*, 2016, pp. 55–64.
[34] Y. Fan, J. Guo, Y. Lan, J. Xu, C. Zhai, and X. Cheng, “Modeling diverse relevance patterns in ad-hoc retrieval,” in *The 41st international ACM SIGIR conference on research & development in information retrieval*, 2018, pp. 375–384.
[35] L. Pang, Y. Lan, J. Guo, J. Xu, J. Xu, and X. Cheng, “Deeprank: A new deep architecture for relevance ranking in information retrieval,” in *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, 2017, pp. 257–266.
[36] B. Mitra, F. Diaz, and N. Craswell, “Learning to match using local and distributed representations of text for web search,” in *Proceedings of the 26th international conference on world wide web*, 2017, pp. 1291–1299.
[37] C. Wang, F. Jiang, and H. Yang, “A hybrid framework for text modeling with convolutional RNN,” in *Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining*, 2017, pp. 2061–2069.
[38] W. Yin and H. Schütze, “Convolutional neural network for paraphrase identification,” in *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2015, pp. 901–911.
[39] J. Huang, S. Yao, C. Lyu, and D. Ji, “Multi-granularity neural sentence model for measuring short text similarity,” in *International Conference on Database Systems for Advanced Applications*, 2017, pp. 439–455.
[40] B. Mitra and N. Craswell, “Neural models for information retrieval,” *arXiv preprint arXiv:1705.01509*, 2017.
[41] D. Mollá and J. L. Vicedo, “Question answering in restricted domains: An overview,” *Computational Linguistics*, vol. 33, no. 1, pp. 41–61, 2007.
[42] Y.-S. Hung, K.-L. B. Chen, C.-T. Yang, and G.-F. Deng, “Web usage mining for analysing elder self-care behavior patterns,” *Expert Systems with applications*, vol. 40, no. 2, pp. 775–783, 2013.
[43] A. Shtok, G. Dror, Y. Maarek, and I. Szpektor, “Learning from the past: answering new questions with past answers,” in *Proceedings of the 21st international conference on World Wide Web*, 2012, pp. 759–768.
[44] Y. Zhang *et al.*, “Generating informative and diverse conversational responses via adversarial information maximization,” *Advances in Neural Information Processing Systems*, vol. 31, 2018.
[45] H. Wang, Z. Lu, H. Li, and E. Chen, “A dataset for research on short-text conversations,” in *Proceedings of the 2013 conference on empirical methods in natural language processing*, 2013, pp. 935–945.
[46] Z. Liu, C. Xiong, M. Sun, and Z. Liu, “Entity-duet neural ranking: Understanding the role of knowledge graph semantics in neural information retrieval,” *arXiv preprint arXiv:1805.07591*, 2018.
[47] B. van den Akker, I. Markov, and M. de Rijke, “ViTOR: learning to rank webpages based on visual features,” in *The world wide web conference*, 2019, pp. 3279–3285.
[48] M. Dehghani, H. Zamani, A. Severyn, J. Kamps, and W. B. Croft, “Neural ranking models with weak supervision,” in *Proceedings of the 40th international ACM SIGIR conference on research and development in information retrieval*, 2017, pp. 65–74.
[49] C. Li *et al.*, “NPRF: A neural pseudo relevance feedback framework for ad-hoc information retrieval,” *arXiv preprint arXiv:1810.12936*, 2018.
[50] L. Pang, Y. Lan, J. Guo, J. Xu, and X. Cheng, “A study of matchpyramid models on ad-hoc retrieval,” *arXiv preprint arXiv:1606.04648*, 2016.
[51] D. Cohen, B. O’Connor, and W. B. Croft, “Understanding the representational power of neural retrieval models using NLP tasks,” in *Proceedings of the 2018 ACM SIGIR International Conference on Theory of Information Retrieval*, 2018, pp. 67–74.