International Journal of Electrical and Computer Engineering (IJECE)
Vol. 10, No. 4, August 2020, pp. 3869~3882
ISSN: 2088-8708, DOI: 10.11591/ijece.v10i4.pp3869-3882

Vertical intent prediction approach based on Doc2vec and convolutional neural networks for improving vertical selection in aggregated search

Sanae Achsas, El Habib Nfaoui
LIIAN Laboratory, Faculty of Sciences Dhar El Mahraz, Sidi Mohammed Ben Abdellah University, Morocco

Article Info

Article history:

Received Jul 18, 2019
Revised Jan 13, 2020
Accepted Feb 1, 2020

Keywords:

Aggregated search
Convolutional neural network
Deep learning
Doc2vec
Information retrieval
Vertical intent
Vertical selection

ABSTRACT

Vertical selection is the task of selecting the most relevant verticals to a given query in order to improve the diversity and quality of web search results. This task requires not only predicting relevant verticals but also these verticals must be those the user expects to be relevant for his particular information need. Most existing works focused on using traditional machine learning techniques to combine multiple types of features for selecting several relevant verticals. Although these techniques are very efficient, handling vertical selection with high accuracy is still a challenging research task. In this paper, we propose an approach for improving vertical selection in order to satisfy the user vertical intent and reduce user's browsing time and efforts. First, it generates query embeddings vectors using the doc2vec algorithm that preserves syntactic and semantic information within each query. Secondly, this vector will be used as input to a convolutional neural network model for increasing the representation of the query with multiple levels of abstraction including rich semantic information and then creating a global summarization of the query features. We demonstrate the effectiveness of our approach through comprehensive experimentation using various datasets. Our experimental findings show that our system achieves significant accuracy. Further, it realizes accurate predictions on new unseen data.

Corresponding Author:

Sanae Achsas,
Department of Computer Science, Faculty of Sciences Dhar El Mahraz,
Sidi Mohammed Ben Abdellah University,
Fez, Morocco.
Email: sanae.achsas@usmba.ac.ma

1. INTRODUCTION

One of the most significant developments online in the last few years is the rising popularity of aggregated search (AS) systems, they present a most popular web search presentation paradigm used by major search engines in recent years, this technique consists of integrating search results from a variety of diverse verticals such as news, images, videos, health, and Wikipedia into a single interface with general web search. As shown in Figure 1 [1], research in aggregated search has taken two main directions. The first direction studies different methods used for predicting which verticals to present known as vertical selection (VS), another direction involves techniques that analyses the way of presenting these verticals in the Web results known as vertical presentation (VP), the research problem investigated in this paper focuses on the first one.

Vertical selection task consists of selecting a subset of the most relevant verticals to a given user information need and improves the search effectiveness while reducing the load of querying a large set of multiple verticals. The main goal behind this task is to help the user to satisfy his information needs.

Identifying the intent behind the user query is a crucial step toward reaching this goal. In a broad sense, the automatic prediction of user intentions helps in enhancing the user experience by returning more relevant results to users and adapting these results to their specific needs. Thus, vertical selection is associated with two main challenges that are: the diversity of the verticals and the understanding of the user intent.

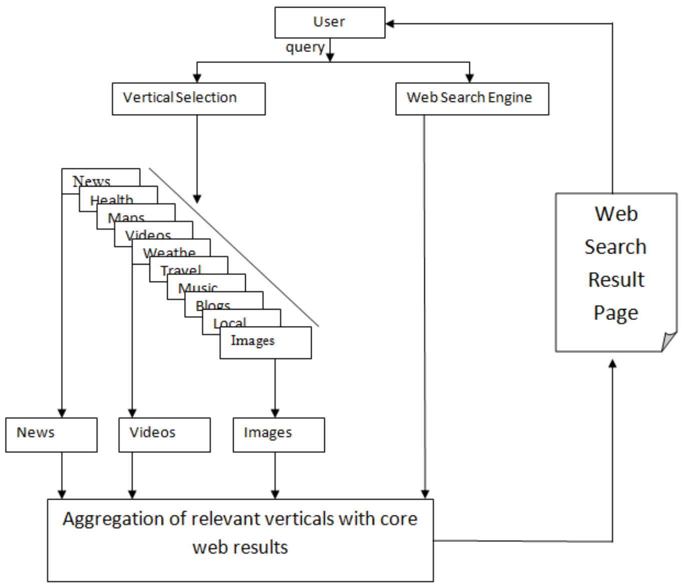

Figure 1. Aggregated search process [1]

Regarding the first challenge, a variety of heterogeneous vertical search engines exist in the web, which means that each vertical has its own features, for example, some verticals are not directly searchable by users like the weather vertical, thus features generated from the vertical query-log will not be available for this kind of verticals. Also, features generated from vertical corpus will not be available for verticals such as the calculator and language translation [2]. Therefore, researchers must deal with the fact that different verticals may require different feature representations when creating new vertical selection approaches.

The second challenge is related to the user intent; we know that a vertical selection system focuses on retrieving relevant verticals for showing its results to the user. For example, if the user searches images and news verticals, he specifically needs the results of these verticals, he isn't interested in the content of the other verticals such as shopping or weather, even if their content is relevant because the goal is to not only have relevant results but also satisfy the user intent.

Existing research papers to date have studied the problem of vertical selection from different ways [3], some prior work focused on constructing models that aim at detecting queries with content-specific vertical such as shopping [4], news [5], jobs [4] or Question Answering [6]. Other works for vertical selection [7-10] focused on using traditional machine learning techniques to combine multiple types of features for selecting several relevant verticals. Although these techniques are very efficient, handling vertical selection with high accuracy is still a challenging research task.

In this paper, we are interested in textual queries; image queries tend to have image results. Our proposed approach for predicting vertical intent consists of generating query embeddings vectors using doc2vec algorithm, that can accurately preserve syntactic and semantic information within each query, therefore we propose to use it as a primary query representation in our vertical selection model pipeline; then it will be used as input to a convolutional neural network (CNN) model that can increase this representation with multiple levels of abstraction comprising rich semantic information and creating a global summarization of the query features. To the best of our knowledge, this is the first time when the benefits of deep learning and paragraph vectors are exploited in the context of vertical selection, which can achieve an amazing progression and development in this area.

The remainder of this paper is structured as follows. In the next section, we review the related work concerning vertical selection. In Section 3, we provide a description of our proposed method. Section 4 is devoted to the experimental settings. We present and discuss the experimental results in Section 5. Finally, we conclude the study and discuss the future issue.

2. RELATED WORK

Aggregated search can be compared to federated search, which aims to provide an integrated search across multiple text collections, referred to as resources, into one single ranking list [11]. Similar to aggregated search, federated search is typically decomposed into two sub-tasks: resource selection and results merging. The main difference between federated search and aggregated search is the heterogeneity of the data and the presentation of the results.

Regarding resource selection, existing approaches can be categorized into two classes of algorithms: term-based and sample-based. For the first class, Term-based algorithms represent shards by collection statistics about the terms in the search engine's vocabulary. Callan et al. [12] propose the CORI algorithm, in which a shard is represented by the number of its documents that contain the terms of a vocabulary. The shards are ranked using the INDRI version of the $tf.idf$ score function using the mentioned number as the frequency of each query term. CORI selects a fixed number of shards from the top of this ranking. Another work of [13] proposes Taily, a novel shard selection algorithm that models a query's score distribution in each shard as a Gamma distribution and selects shards with highly scored documents in the tail of the distribution. Taily estimates the parameters of score distributions based on the mean and variance of the score function' s features in the collections and shards. Concerning the algorithms of the second category, they use a central sample index (CSI) of documents from each shard for shard selection. For example, in [14] authors proposed REDDE algorithm, where they rank shards according to the number of the highest ranked documents that belong to this shard, weighted by the ratio between the shard's size and the size of the shard's documents in the CSI. The SUSHI algorithm [15] choose the best fitting function from a list of possible functions between the estimated ranks of a shard's documents in the CSI and their observed scores from the initial search. Using this function, the algorithm estimates the scores of the top-ranked documents of each shard. SUSHI selects shards based on their estimated number of documents among a number of top-ranked documents in the global ranking.

In the other hand, identifying the query intent of a user is a well-known problem in Information retrieval and have been studied by a considerable number of researchers in this field, where its goal is to select relevant documents or web search results that can satisfy the user information need. For this reason, most of the approaches cited in the literature employ query logs, supervised, semi-supervised classifiers, different language models and word embeddings. We have for example some popular works that examined query intent detection, like the work of [16] where the authors have developed a new dataset with almost 2,000 queries tagged in informational, navigational and transactional categories. They calculated features for each of these queries using a real-world query log. Jiang et Yang in [17], tackle the problem of query intent inference by integrating multiple information sources in a seamless manner. They first propose a comprehensive data model called Search Query Log Structure (SQLS) that represents the relationship between search queries via the User dimension, the URL dimension, the Session dimension, and the Term dimension. Then they propose three new frameworks that are effective to infer query intents by mining the multidimensional structure constructed from the search query log. Another work in [18] explores the competence of lexical statistics and embedding method. First, a novel term expansion algorithm is designed to sketch all possible intent candidates. Moreover, an efficient query intent generation model is proposed, which learns latent representations for intent candidates via embedding-based methods, and then vectorized intent candidates are clustered and detected as query intents. Kim et al. [19] used enriched word embeddings to force semantically similar or dissimilar words to be closer or farther away in the embedding space to improve the performance of intent detection task for spoken language understanding, and thus by exploiting several semantic lexicons, such as WordNet, PPDB (Paraphrase Database), and Macmillan Dictionary, and using them later as initial representation of words for intent detection. Thus, they train an end-to-end model that jointly learns the slot and intent classes from the training data by building a bidirectional LSTM (Long Short-Term Memory). In [20], the proposed approach aims at identifying query intent as a multi-class classification task which extracted query vector representations using CNN instead of engineering query features. This method uses deep learning to find query vector representations, then use them as features to classify queries by intent.

As we can see in the reviewed approaches above, the problem of query intent detection is studied without considering the vertical intent issue. knowing that with the appearance of aggregated search, various web search engines do not return uniform lists of Web pages, but they also include results of a different type from different verticals such as images, news, videos, and so on, which means that the key component of this domain is the notion of verticals. Therefore, an aggregated search system must make predictions about which vertical(s) are more expected to answer the issued query, which allows reducing the load of querying a large set of multiple verticals, and then improves the search effectiveness.

The relevance of a vertical depends basically on two factors, the relevance of the documents within the vertical collection and on the user's intent to the vertical (Vertical Intent). For the first one, to make vertical selection decision, various approaches use traditional machine learning techniques to combine different sources of evidence that can be found in Query features (features depend only on the query) [6-9], in Vertical features (features depend only on the vertical) [2, 5, 21, 22] and in Vertical-Query features (features aim to measure relationships between the vertical and the query, and are therefore unique to the vertical-query pair) [2, 7, 22-24]. Indeed, among all these works, there are those that integrate the content from a single vertical [3]. In this respect, Li et al. [4] address the general problem of vertical selection using an approach that focuses on shopping and job verticals to extract implicit feedback using semi-supervised learning based on clickthrough data. Diaz [5] investigated also the vertical selection problem with respect to the news vertical, where he derived features from news collection, web and vertical query-logs and incorporated click-feedback into the model. More recent work in [6] has also targeted a variant of the vertical selection problem, where the author use Community Question Answering (CQA) Verticals for detecting queries with CQA intent.

Other approaches have been developed where several verticals are considered simultaneously [3]. Arguello et al. [7] propose a classification-based approach for vertical selection in which they exploit features from the vertical content, the query string, and the vertical's query log. The click-through data is used to construct a descriptive language model for each vertical's related queries. Diaz and Arguello [9] also present several algorithms for combining user feedback with offline classifier information, the focus of their work was to maximize user satisfaction by presenting the appropriate vertical display. Another work from Arguello et al [8] have been proposed where the goal was to use training data associated with a set of existing verticals in order to learn a model that can make vertical selection predictions for a target vertical. Recent work in the same context was proposed from [10], in which the desired vertical of the user is placed on the top of the web result page. This is achieved by predicting verticals based on the user' s past behavior.

Regarding the second factor, there are a few works addressing the vertical intent issue when studying query intent detection. From the user intent perspective, user vertical intent also plays an important role in the improvement of the aggregated search process. For example, Zhou et al. [25] propose a methodology to predict the vertical intent of a query using a search engine log by exploiting click-through data. Recent work of Tsur et al. [6] present a supervised classification scheme where they aim at detecting queries with question intent as a variant of the vertical selection problem. They introduced two classification schemes that consider query structure. In the first approach, they induce features from the query structure as an input to supervised linear classification. In the second approach, word clusters and their positions in the query are used as input to a random forest classifier to identify discriminative structural elements in the query.

Despite its interesting role, the research in this direction is still limited, and there is no huge literature regarding vertical selection based on user vertical intent, especially with the evolution shown in IR (Information Retrieval) during the last years. Therefore, we will focus in this work on the problem of vertical intent prediction, where we propose a new approach that combines the doc2vec algorithms and convolutional neural networks and exploits for the first time the benefits of both techniques in order to improve vertical selection task.

3. PROPOSED APPROACH

Through the vertical selection process, the query is processed and sent to multiple verticals as well as the Web search engine, in order to decide which of those should be selected for a given query, this depends on what are the verticals intended to be retrieved by the user, we refer to this as the user vertical intent (VI) and it can be defined as follows:

Given a user's query (Q) and a set of candidate verticals $V = \{v_1, v_2, \ldots, v_n\}$, the vertical intent I_Q is represented by the vector $I_Q = \{i_1, i_2, \ldots, i_n\}$, where each value i_k indicates the importance of the given vertical v_k to the query Q, and for each vertical v_k, given a threshold δ, above which the vertical is assumed to have a high intent for the query: if $i_k > \delta$ then we can say that the vertical v_k is intended by the query Q.

To address this problem, we propose an approach based on Doc2vec and Convolutional Neural Networks. First, it generates a query embeddings vector using the doc2vec algorithm that preserves syntactic and semantic information within each query. Secondly, this vector will be used as input to a Convolutional Neural Network model for increasing the representation of the query with multiple levels of abstraction including rich semantic information and then creating a global summarization of the query features. Figure 2 shows the overall architecture of our proposed vertical selection system. It contains two main parts, the semantic representation of the query and the query level feature extraction. Subsections bellow describe these two parts and all performed steps in depth.

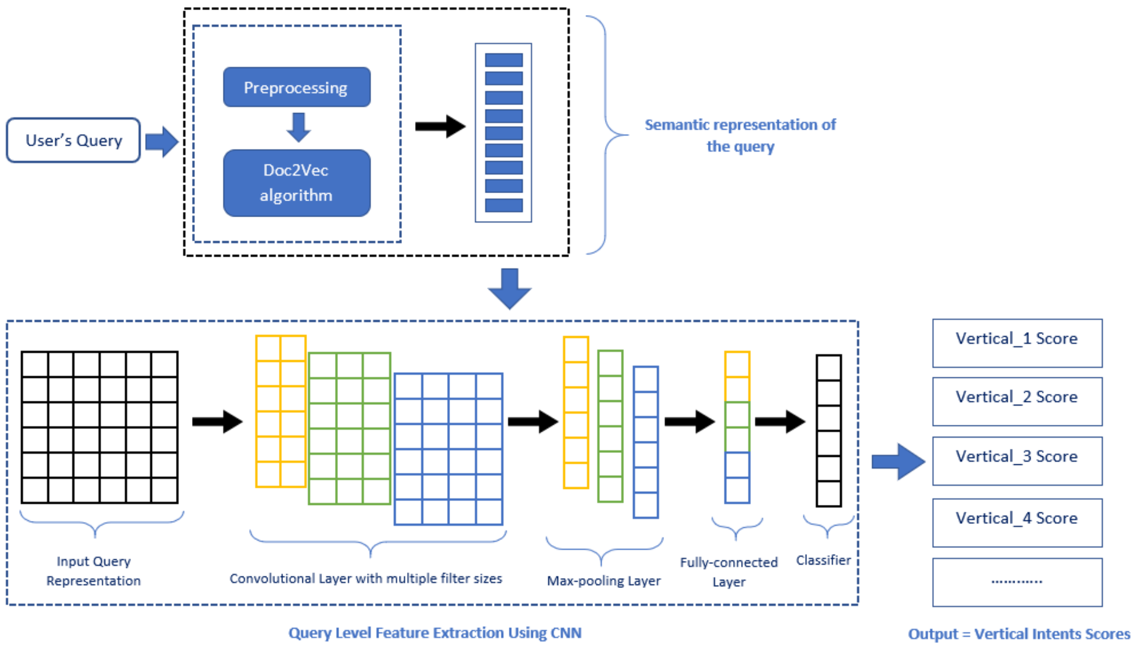
User's Query
Preprocessing
Doc2Vec algorithm
Semantic representation of the query
Input Query Representation
Convolutional Layer with multiple filter sizes
Max-pooling Layer
Fully-connected Layer
Classifier
Query Level Feature Extraction Using CNN
Vertical_1 Score
Vertical_2 Score
Vertical_3 Score
Vertical_4 Score
....
Output = Vertical Intents Scores

Figure 2. Architecture of the proposed vertical selection system

3.1. Semantic representation of the query

The core algorithm of this step is doc2vec which is an unsupervised model that is used most to construct distributed representations of arbitrarily long sentences. It is an extension of word2vec that learns fixed-length feature representations for variable-length pieces of texts such as sentences, paragraphs, and documents [26]. A Doc2vec or paragraph vectors has two different architectures: The Distributed Bag-of-Words model and the Distributed Memory model. The Distributed Bag-of-Words (DBOW) model trains faster and does not consider word order; it predicts a random group of words in a paragraph based on the provided paragraph vector. In the Distributed Memory (DM) model, the paragraph is treated as an extra word, which is then averaged with the local relevant word vectors for making predictions. This method, however, acquires additional calculation but can achieve better results than DBOW.

We chose Doc2Vec model because it overcomes the disadvantages of the other bag-of-words models by learning semantic relationships between words, this is why it has been widely used recently in various NLP tasks [27-29] and Information Retrieval works [30-35] where it has proven that it is able to capture the semantics of paragraphs which leads to excellent results. In this step, the goal is to learn a good semantic representation of the input query, following that idea; we tried to represent each query as vector and used this vector as features for our classification model as presented in Figure 2. Therefore, Doc2vec algorithm contributes effectively for improving the performance of our system.

3.2. Query level feature extraction

Unlike traditional approaches that require handcrafted features to predict relevant verticals, our approach consists of using a Convolutional Neural Network to extract the most important semantic features that represent each query and delete those that are unnecessary. Recently, CNNs have achieved promising performances in various NLP tasks, such as Information Extraction [36], Summarization [37], Machine Translation [38], Classification [39], Question Answering [40] and other traditional NLP tasks [41-43]. CNN architecture used in this paper is shown in Figure 3.

- *Input Layer:* First of all, once the query embeddings vectors are generated from the previous step, we use them to construct the input matrix needed in the embedding layer where each query is being represented as a 2-dimensional matrix.
- *Convolutional layer:* The primary purpose of this layer is to capture the syntactic and semantic features of the entire query and *compress* these valuable semantics into feature maps. Thus, we perform several convolutions over the embedded query vectors using multiple filters with different window size. As the filter moves on, various features are produced and combined into a feature map. Activation functions are added to incorporate element-wise non-linearity.

- *Pooling layer:* In this layer, the goal is to extract the most relevant features within each feature map, therefore, we use the max-pooling strategy for the pooling operation. Since there are multiple feature maps, we have a vector after each pooling *operation*. All vectors that are obtained from the max-pooling layer are concatenated into a fixed-length feature vector.
- *Fully connected layers:* these layers constitute the classification part of our model; their main purpose is to use high-level features obtained *from* the previous layers and passed them to the final softmax layer for classifying the input query into various classes based on its vertical intents scores.

In addition to these different layers, there are some optimizations that we have performed in order to reduce the overfitting and obtain better test accuracy. These optimizations include applying an l2 norm constraint of the weight vectors in the convolutions and fully connected layers as well as adding several Batch Normalization layers, that normalize the activations of the previous layer for each batch during training, which helps to train our model faster and consequently improve our model's performance.

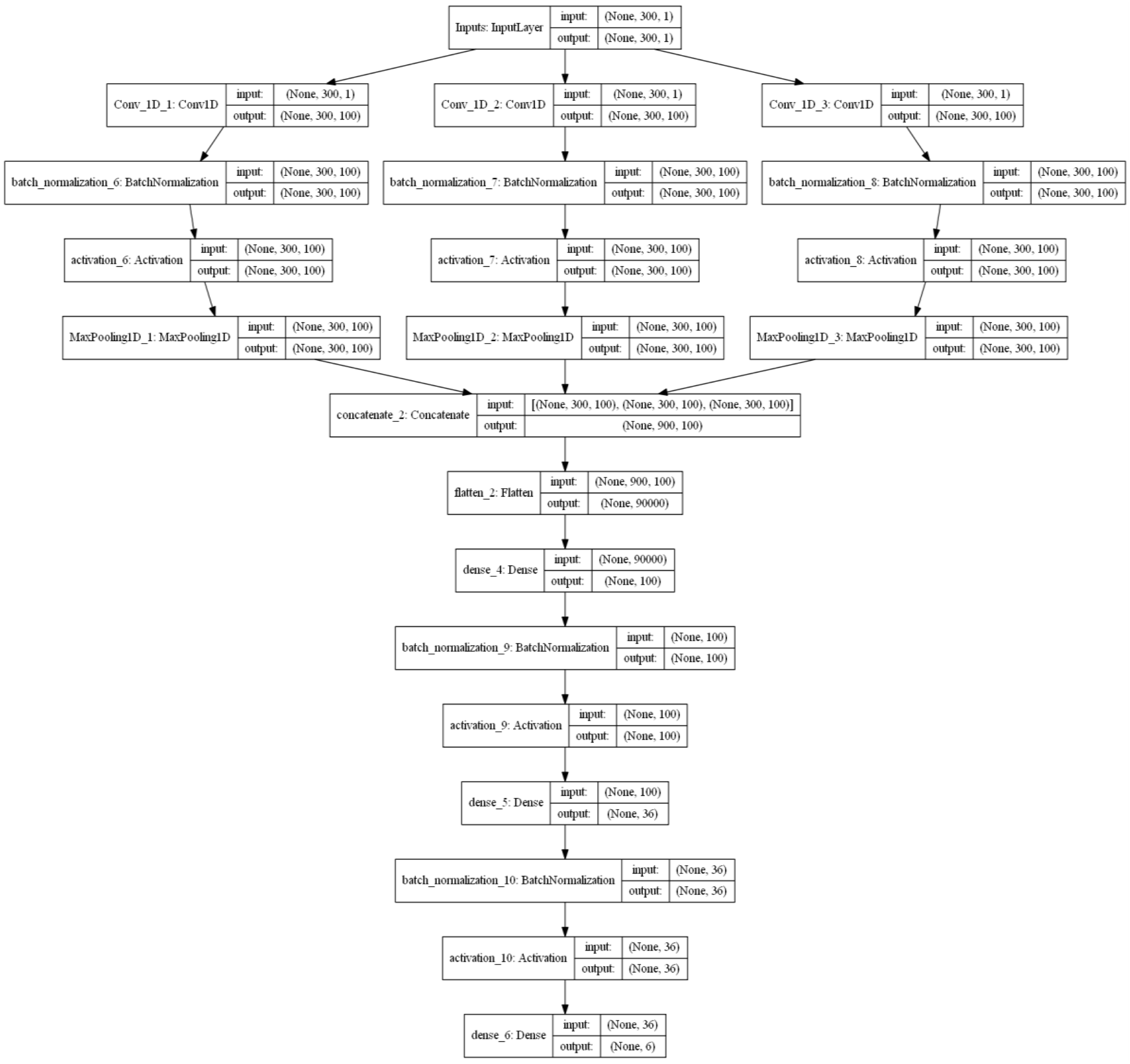
Inputs: InputLayer | input: (None, 300, 1) | output: (None, 300, 1)
Conv_1D_1: Conv1D | input: (None, 300, 1) | output: (None, 300, 100)
Conv_1D_2: Conv1D | input: (None, 300, 1) | output: (None, 300, 100)
Conv_1D_3: Conv1D | input: (None, 300, 1) | output: (None, 300, 100)
batch_normalization_6: BatchNormalization | input: (None, 300, 100) | output: (None, 300, 100)
batch_normalization_7: BatchNormalization | input: (None, 300, 100) | output: (None, 300, 100)
batch_normalization_8: BatchNormalization | input: (None, 300, 100) | output: (None, 300, 100)
activation_6: Activation | input: (None, 300, 100) | output: (None, 300, 100)
activation_7: Activation | input: (None, 300, 100) | output: (None, 300, 100)
activation_8: Activation | input: (None, 300, 100) | output: (None, 300, 100)
MaxPooling1D_1: MaxPooling1D | input: (None, 300, 100) | output: (None, 300, 100)
MaxPooling1D_2: MaxPooling1D | input: (None, 300, 100) | output: (None, 300, 100)
MaxPooling1D_3: MaxPooling1D | input: (None, 300, 100) | output: (None, 300, 100)
concatenate_2: Concatenate | input: [(None, 300, 100), (None, 300, 100), (None, 300, 100)] | output: (None, 900, 100)
flatten_2: Flatten | input: (None, 900, 100) | output: (None, 90000)
dense_4: Dense | input: (None, 90000) | output: (None, 100)
batch_normalization_9: BatchNormalization | input: (None, 100) | output: (None, 100)
activation_9: Activation | input: (None, 100) | output: (None, 100)
dense_5: Dense | input: (None, 100) | output: (None, 36)
batch_normalization_10: BatchNormalization | input: (None, 36) | output: (None, 36)
activation_10: Activation | input: (None, 36) | output: (None, 36)
dense_6: Dense | input: (None, 36) | output: (None, 6)

Figure 3. Illustration of the CNN model architecture

4. EXPERIMENTS

This section describes the datasets used and the various hyperparameters chosen for evaluating the performance of our system, it also gives details about how the doc2vec and CNN were trained.

4.1. Datasets

In these experiments, we employ three public datasets for the training, validation and testing our proposed system respectively, a summary statistic of these datasets is listed in Table 1. We describe each dataset in detail below:

In the first dataset, we use the official NTCIR-12 IMine-2 vertical intent collection for English Subtopics [44] which was designed to explore and evaluate the technologies of understanding user intents behind the query. IMINE-2 includes a set of 100 topics (i.e. queries). Each topic is labeled by a set of intents with probabilities, and there is a set of subtopics as relevance judgment for each vertical intent. A subtopic of a given query is viewed as a search intent that specializes and/or disambiguates the original query, the number of these subtopics is 533. The general idea is to first generate a more complete representation of the different possible intents associated with the input query, and then to perform vertical selection for each intent separately.

In addition, this dataset includes five types of queries, namely “ambiguous”, “faceted”, “very clear”, “task-oriented”, and “vertical-oriented”, this allows us to investigate the performances of our system with diverse queries and varied topics. The details of the five query types are as follows [44]:

a. Ambiguous: The concepts/objects behind the query are ambiguous (e.g., "Jaguar" -> car, animal, etc.).
b. Faceted: The information needs behind the query include many facets or aspects (e.g., “harry potter” -> movie, book, Wikipedia, etc.).
c. Very clear: The information need behind the query is very clear so that usually a single relevant document can satisfy his information needs. (e.g., “apple.com homepage”)
d. Task-oriented: The search intent behind the query relates the searcher’s goal (e.g., “lose weight” -> exercise, healthy food, medicine, etc.).
e. Vertical-oriented: The search intent behind the query strongly indicates a specific vertical (e.g., “iPhone photo” -> Image vertical).

We use this dataset as training data from which we construct a validation set by selecting 10% randomly. The second dataset used in this paper is FedWeb’14 [45] that is used in the TREC FedWeb track 2014, the collection contains search result pages from 108 web search engines (e.g. Google, Yahoo!, YouTube and Wikipedia). For each engine, 75 test topics were provided, from which 50 will be used for vertical selection evaluations. We conduct a prediction task for these 50 test queries and compare the result obtained with the real values in the dataset.

The last dataset is TREC 2009 Million Query Track [46], this collection contains 40000 queries that were sampled from two large query logs. To help anonymize and select queries with relatively high volume, they were processed by a filter that converted them into queries with roughly equal frequency in a third query log. This dataset is used as training data for doc2vec algorithm to improve its performance with large data. Moreover, the different queries used in these experiments have different lengths, from short to long queries as shown in Figures 4, 5 and 6. The structure of our model allows us to learn both kinds.

Table 1. Statistical information of various datasets

Dataset	Description	Size (number of queries)	Max Query Length (number of words)	Task
NTCIR-12 IMINE-2 English Subtopic Mining [44]	It contains the Query understanding subtask and the Vertical Incorporating subtask.	533	12	90% Training 10% Validation
FedWeb’14 [45]	It is designed to resource selection, results merging and vertical selection tasks.	50	9	Testing
TREC Million Query Track 2009 [46]	It is an exploration of ad hoc retrieval over a large set of queries and a large collection of documents, and it investigates questions of system evaluation.	40000	16	doc2vec' training data

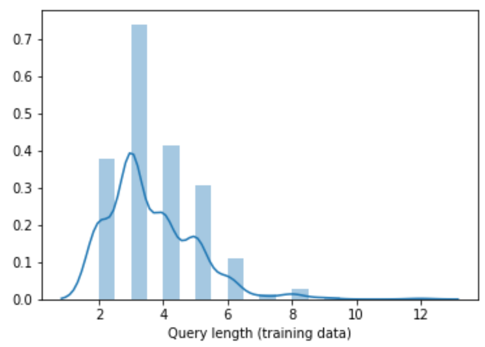

Figure 4. Query length for NTCIR-12 IMine-2

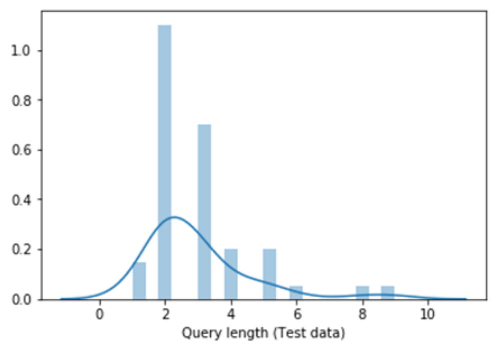

Figure 5. Query length for FedWeb’14

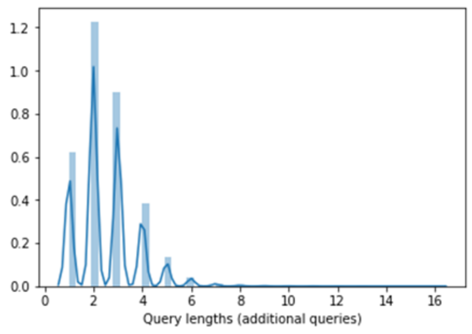

Figure 6. Query length for TREC million query track 2009

4.2. Hyperparameters and training

Firstly, in order to train and evaluate our proposed system, we need to adjust several hyperparameters. Indeed, regarding the first part of our architecture, to use doc2vec for our datasets, we first trained doc2vec model (Python gensim library implementation) on TREC 2009 Million Query Track datasets using Distributed Memory model. Then we transformed all the queries on both training and testing sets to Doc2Vec vectors. The various parameters used for training doc2vec model are shown in Table 2.

In the second part, the implementation of our network is made using Keras framework with TensorFlow backend. For the input layer, we used Gensim's doc2vec embeddings and created input data from it, instead of using keras embedding layer. To build our CNN architecture there are many hyperparameters to choose from. Therefore, we first consider the performance of a baseline CNN configuration using the parameters described in Table 3.

Table 2. Different parameters used to train Doc2Vec model

Parameters	alpha	min_alpha	min_count	Vector size	Number of epochs	Library used
Value	0.025	0.0025	1	300	100	Gensim

Table 3. Parameters of the baseline CNN configuration

filter region sizes	Feature maps	l2 regularization	Dropout rate	Batch size	optimizer	Activation function	pooling	Loss function	Number of epochs
[3-5]	100	0.01	0.5	64	adam	ReLU	max pooling	Mean squared error	100

Then, we evaluated the effect of each of the other parameters by holding all other settings constant and vary only the factor of interest. During these experiments, we chose to use ReLU activation function and max-pooling strategy for our CNN model with mean squared error loss function, these parameters are mostly used in CNN architectures and gives good performances. Finally, we combined all the good variation results obtained from these experiments, and used them for our suggested CNN model.

5. RESULTS AND DISCUSSION

In this section, we present and discuss our experimental results obtained by each part of our vertical selection system.

5.1. Semantic representation of the query using Doc2vec

Starting with the first part of our proposed system, once doc2vec model is trained, we use it to generate an embedding vector for each query in our collections (train set and validation set + test set). These embedding vectors must capture semantic meanings of each of these queries. In this respect, to make sure that this model achieved this goal, we used our doc2vec model to find the most similar queries for two sample queries in our dataset, the resulting queries, and corresponding scores are presented in Table 4. As we can see in this table, the similarity scores between the first queries and the second ones are significant. We figure out that this doc2vec model is a meaningful model and can successfully recognize semantic information among queries.

Table 4. Similarity score for two sample queries from our collections

First query	Second query	Their similarity score
Mother's Day gifts shopping site	Wallpaper computer	0.24370566066263763
Make resume online free	Make resume free templates	0.9092499447309569
Apple latest news products	home cleaning products	0.8546902948569026
bananas seeds plant	bananas seed inside	0.9979968070983887
Virginia tourism	Iraq war pictures	0.1463257649642163
wallpaper magazine	Asian culture	0.29107128845148067

5.2. Query level feature extraction using CNN

As we have evaluated the quality of the query vectors obtained in the previous part, we proceed now to the evaluation of our model architecture. First of all, we start by assessing the impact of various parameters used in order to configure our CNN model and obtain the best possible results.

5.2.1. Impact of filter sizes

We explored the effect of various filter sizes, while keeping the number of filters for each region size fixed at 100. Figure 7 shows that the plot for the filter size of [2-4] was at the top of all the other plots throughout the run, and it yielded better accuracy (69.94%) compared to filter sizes [3-5] and [4-6] (66.39% and 67.22% respectively) as reported in Table 5.

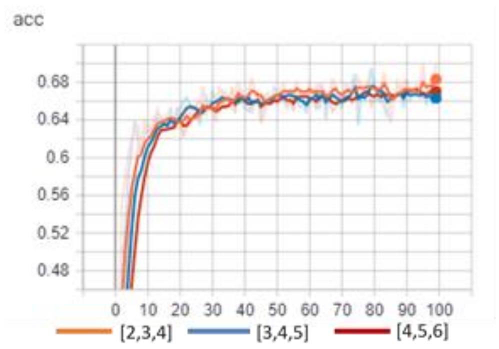

Figure 7. Accuracy comparison of filter size variations

Table 5. Accuracy comparison of filter size variations

Filter sizes	[2-4]	[3-5]	[4-6]
Accuracy	69.94%	66.39%	67.22%

5.2.2. Impact of feature maps

The variations of the number of filters per filter sizes don't help much as shown in Figure 8, but still there are a few noticeable accuracy results when the number of filters is 200 (68.48%) as shown in Table 6.

5.2.3. Impact of regularization

We have used two common regularization strategies for our CNN, that are dropout and l2 norm constraints. We explore the effect of these two strategies here. we presented the effect of the l2 norm imposed on the weight vectors in Figure 9 and Table 7. We then experimented with varying the dropout rate from 0.1 to 0.5 as shown in in Figure 10 and Table 8, fixing the l2 norm constraint to 0.01. The variations in regularization show that for the l2 norm constraint, the classification performance is higher with value=0 which produce best results compared to higher values that often hurts performance as shown in Figure 9 and Table 7. Separately, we considered the effect of dropout rate. Figure 10 indicate that when setting the dropout rate to 0.2 it gives the best accuracy results (69.73%) and this value decrease when we increase the dropout rate as reported in Table 8.

5.2.4. Impact of batch size

We next investigated the effect of the Batch size. Figure 11 and Table 9 show that varying the batch size also helps little, and the best accuracy result obtained was 67.01% with a batch size of value 60.

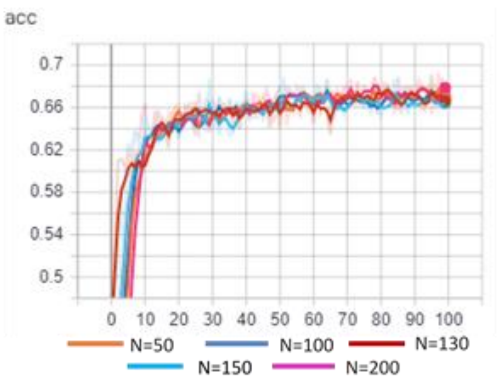

Figure 8. Impact of the number of filters per filter size

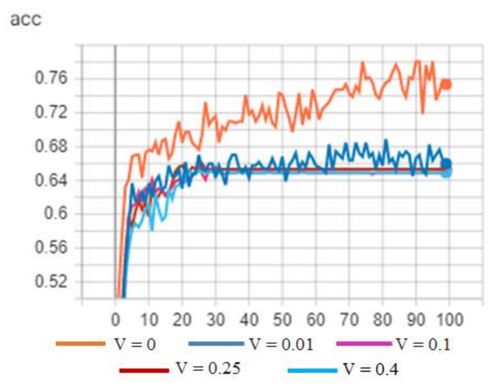

Figure 9. Accuracy comparison with various variations of l2 norm constraint

Table 6. Impact of the number of filters per filter size

Number of filters	Accuracy (%)
N=50	66.39%
N=100	66.81%
N=130	65.97%
N=150	66.60%
N=200	68.48%

Table 7. Accuracy comparison with various variations of l2 norm constraint

l2 norm constraint value	Accuracy
0	75.37%
0.01	65.97%
0.1	65.14%
0.25	65.34%
0.4	64.93%

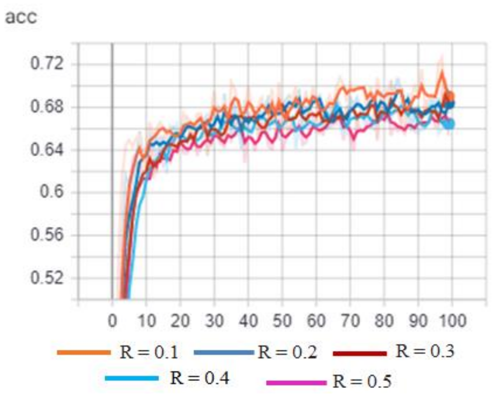

Figure 10. Accuracy comparison with various variations of dropout rate

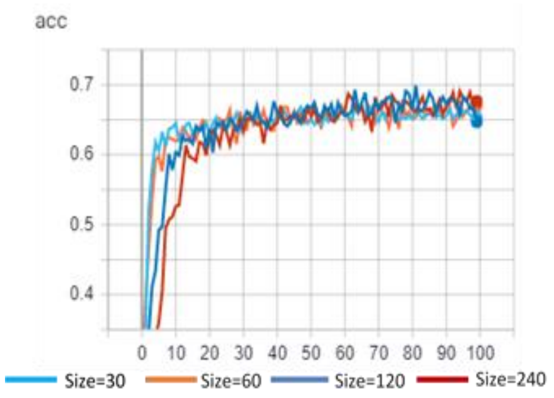

Figure 11. Accuracy comparison with various batch sizes

Table 8. Accuracy comparison with various variations of dropout rate

Dropout rate	Accuracy
0.1	67.64%
0.2	69.73%
0.3	67.43%
0.4	65.97%
0.5	65.34%

Table 9. Accuracy comparison with various batch sizes

Batch sizes	Accuracy
30	65.34%
60	67.01%
120	64.72%
240	66.64%

5.2.5. Impact of optimizers

Regarding the optimizers, we can see clearly from Figure 12 that only the curves of “adam” and “adadelta” optimizers were at the top. Thus, they produce best accuracy results (66.81% and 65.97%) compared to “sgd” (56.16%) as shown in Table 10. Next, we exploited the observations given above to configure our CNN model with the best variations of parameters. We summarize the suggested parameters in Table 11.

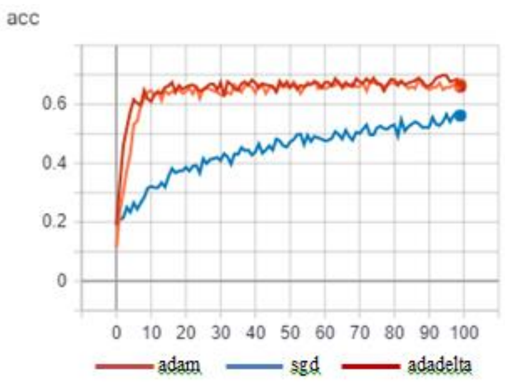

Figure 12. Accuracy comparison with various optimizers

Table 10. Accuracy comparison with various optimizers

Optimizers	adam	sgd	adadelta
Accuracy	66.81%	56.16%	65.97%

Table 11. Various parameters used for the suggested CNN model

Filter sizes	Number of filters per filter size	l2 regularization	Dropout rate	Batch size	optimizer
[2-4]	200	0	0.2	60	adam

Now, we can evaluate our model with the suggested values. In this respect, we choose to assess the CNN architecture using the accuracy and the Mean Square Error (MSE) loss, that represent the most important and intuitive metrics for classification performance. They are defined as follows:

$$accuracy = \frac{\#\, correct\ predictions}{\#\, predictions} \tag{1}$$

$$MSE = \frac{1}{n}\sum(y-\hat{y})^2 \tag{2}$$

where $(y-\hat{y})^2$ present the square of the difference between the actual and the predicted result. Figures 13 and 14 show respectively the values of the loss and the accuracy obtained on both training and validation sets. After training for 100 epochs we have obtained the results shown in Table 12. These results show that an accuracy up to 80% seems to be the best value that can be obtained in this experiment.

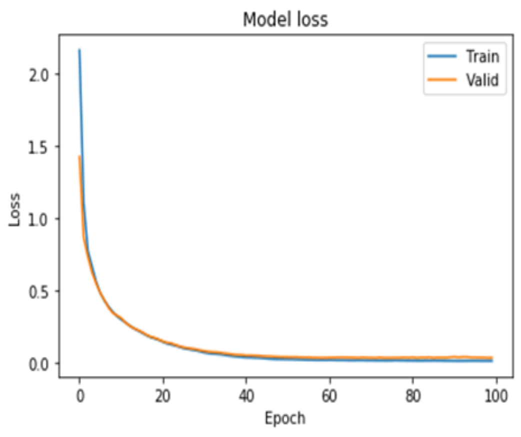

Figure 13. Loss of the proposed model for the training and the validation sets

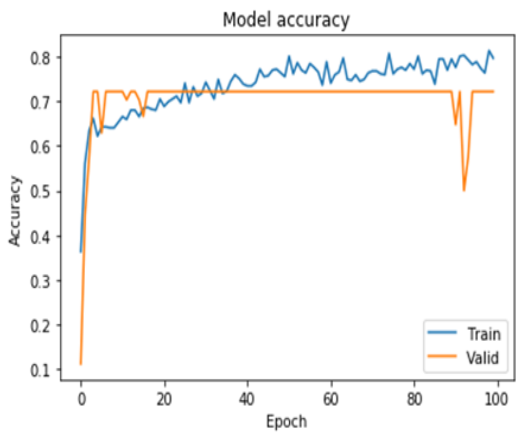

Figure 14. Accuracy of the proposed model for the training and the validation sets

Table 12. Classification accuracy and loss results of the proposed approach

	Training accuracy	Validation accuracy	Training loss	Validation loss
Baseline model	65.34%	70.37%	3.87%	4.48%
Suggested model	79.75%	72.22%	1.56%	3.73%

The difference between the results obtained from the baseline model and the suggested model show clearly the impact of the suggested values chosen. Since there is a noticeable increase in the accuracy value between the two models. Moreover, Figure 14 shows that the classification accuracy is varied, however, after 20 epochs, we obtain the higher value. There is also a difference in accuracy between training and validation sets.

In the other hand, by using mean square error (MSE) as a classification loss, we optimize by reducing the average squared distance between our predictions and the true values. In Figure 13, we can see that the validation loss is synchronized with the training loss, and the validation loss is decreasing. All these observations prove that our system achieves significant results (accuracy and loss) which demonstrate its performance even if our data isn't that big. In most cases, the size of the data is not very important, what is important is the variations in the samples we have, which is the case of our training dataset that includes a variety of samples.

5.3. The effectiveness of our vertical intent prediction system

5.3.1. Predictions on unseen data

To complete our experiments, we evaluate the performance of our model on the prediction task of vertical intents on new unseen data. Now that we have trained our model and evaluated its accuracy on the validation set, we can use it to make predictions, this step is done using the test dataset (FedWeb 14) that has not been used for any training or validation operations. This is an estimate of the efficiency of the algorithm trained on the problem when making predictions on unseen data.

As mentioned previously, we have already generated query vectors for the test dataset using doc2vec in the first part of this experimentation. Thus, we have fed these vectors to our CNN model for predictions, which allows us to obtain a vertical intent score for each query. Evaluating the quality of these predictions using mean squared error gives us 1.31% which prove that our model achieves accurate predictions.

6. CONCLUSION AND FUTURE WORK

In this paper, we proposed a vertical intent prediction architecture to improve the vertical selection task. This approach outperforms the traditional vertical selection approaches by the fact that it considers satisfying the user vertical intent automatically and without any feature engineering, unlike the others that addressed this problem by training a machine-learned model based on the features that are supposed to detect the vertical relevance to the query, which is very expensive and takes more time. Our experimental findings show that our model can achieve acceptable accuracy. The research can be continued in future work in order to increase accuracy. In addition, this work represents an advancement in the context of vertical selection, which will renew the research in this field where there is no huge literature.

REFERENCES

[1] S. Mustajab, Mohd. Kashif Adhami, and R. Ali, "An Overview of Aggregating Vertical Results into Web Search Results," *Int. J. Comput. Appl.*, vol. 69, no. 17, pp. 21-28, May 2013. Doi: 10.5120/12063-8107.

[2] Jaime Arguello, "Aggregated search," *Found. Trends Inf. Retr.*, vol. 10, no. 5, pp. 365-502, 2017.

[3] S. Achsas and E. H. Nfaoui, "An Analysis Study of Vertical Selection Task in Aggregated Search," *Procedia Computer Science, Elsevier*, vol. 148, pp. 171-180, Jan. 2019. Doi: 10.1016/j.procs.2019.01.021.

[4] X. Li, Y.-Y. Wang, and A. Acero, "Learning query intent from regularized click graphs," in *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval-SIGIR '08*, Singapore, Singapore, pp. 339, 2008. Doi: 10.1145/1390334.1390393.

[5] Fernando Diaz, "Integration of news content into web results," in *Proceedings of the Second ACM International Conference on Web Search and Data Mining-WSDM '09*, Barcelona, Spain, pp. 182, 2009.

[6] G. Tsur, Y. Pinter, I. Szpektor, and D. Carmel, "Identifying Web Queries with Question Intent," in *Proceedings of the 25th International Conference on World Wide Web-WWW '16*, Montr&#233;al, Qu&#233;bec, Canada, pp. 783-793, 2016, doi: 10.1145/2872427.2883058.

[7] J. Arguello, F. Diaz, J. Callan, and J.-F. Crespo, "Sources of evidence for vertical selection," in *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval-SIGIR '09*, Boston, MA, USA, pp. 315, 2009. Doi: 10.1145/1571941.1571997.

[8] J. Arguello, F. Diaz, and J.-F. Paiement, "Vertical selection in the presence of unlabeled verticals," in *Proceeding of the 33rd international ACM SIGIR conference on Research and development in information retrieval-SIGIR '10*, Geneva, Switzerland, pp. 691, 2010. Doi: 10.1145/1835449.1835564.

[9] F. Diaz and J. Arguello, "Adaptation of offline vertical selection predictions in the presence of user feedback," in *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval-SIGIR '09*, Boston, MA, USA, pp. 323, 2009. Doi: 10.1145/1571941.1571998.

[10] D. Bakrola and S. Gandhi, "Enhancing Web Search Results Using Aggregated Search," in *Proceedings of International Conference on ICT for Sustainable Development*, S. C. Satapathy, A. Joshi, N. Modi, and N. Pathak, Eds. Singapore: Springer Singapore, vol. 409, pp. 675-688, 2016.

[11] M. Shokouhi and L. Si, "Federated Search," *Found. Trends Inf. Retr.*, vol. 5, no. 1, pp. 1-102, 2011.

[12] J. P. Callan, Z. Lu, and W. B. Croft, "Searching Distributed Collections with Inference Networks," pp. 8.

[13] R. Aly, D. Hiemstra, and T. Demeester, "Taily: shard selection using the tail of score distributions," in *Proceedings of the 36th international ACM SIGIR conference on Research and development in information retrieval-SIGIR '13*, Dublin, Ireland, pp. 673, 2013, doi: 10.1145/2484028.2484033.

[14] L. Si and J. Callan, "Relevant Document Distribution Estimation Method for Resource Selection," pp. 8.

[15] P. Thomas and M. Shokouhi, "SUSHI: scoring scaled samples for server selection," in *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval-SIGIR '09*, Boston, MA, USA, pp. 419, 2009, doi: 10.1145/1571941.1572014.

[16] J. Zamora, M. Mendoza, and H. Allende, "Query Intent Detection Based on Query Log Mining," *Journal of Web Engineering*, vol. 13, no. 1&2, pp. 30, 2014.

[17] D. Jiang and L. Yang, "Query intent inference via search engine log," *Knowl. Inf. Syst.*, vol. 49, no. 2, pp. 661-685, Nov. 2016. Doi: 10.1007/s10115-015-0915-7.

[18] J. Gu, C. Feng, X. Gao, Y. Wang, and H. Huang, "Query Intent Detection Based on Clustering of Phrase Embedding," in *Social Media Processing*, Y. Li, G. Xiang, H. Lin, and M. Wang, Eds. Singapore: Springer Singapore, vol. 669, pp. 110-122, 2016.

[19] J.-K. Kim, G. Tur, A. Celikyilmaz, B. Cao, and Y.-Y. Wang, "Intent detection using semantically enriched word embeddings," in *2016 IEEE Spoken Language Technology Workshop (SLT)*, San Diego, CA, pp. 414-419, 2016. Doi: 10.1109/SLT.2016.7846297.

[20] H. B. Hashemi, A. Asiaee, R. Kraft, "Query Intent Detection using Convolutional Neural Networks," pp. 5, 2016.

[21] S. Makeev, A. Plakhov, and P. Serdyukov, "Personalizing Aggregated Search," in *Advances in Information Retrieval*, M. de Rijke, T. Kenter, A. P. de Vries, C. Zhai, F. de Jong, K. Radinsky, and K. Hofmann, Eds. Cham: Springer International Publishing, vol. 8416, pp. 197-209, 2014.

[22] Kopliku, K. Pinel-Sauvagnat, and M. Boughanem, "Aggregated search: A new information retrieval paradigm," *ACM Comput. Surv.*, vol. 46, no. 3, pp. 1-31, Jan. 2014. Doi: 10.1145/2523817.

[23] Mounia Lalmas, "Aggregated Search," in *Advanced Topics in Information Retrieval*, M. Melucci and R. Baeza-Y., Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, vol. 33, pp. 109-123, 2011.

[24] S. Achsas and E. H. Nfaoui, "Intelligent layer for relational aggregated search based on deep neural networks," in *2016 IEEE/ACS 13th International Conference of Computer Systems and Applications (AICCSA)*, Agadir, Morocco, pp. 1-2, 2016. Doi: 10.1109/AICCSA.2016.7945754.

[25] K. Zhou, R. Cummins, M. Halvey, M. Lalmas, and J. M. Jose, "Assessing and Predicting Vertical Intent for Web Queries," in *Advances in Information Retrieval*, R. Baeza-Yates, A. P. de Vries, H. Zaragoza, B. B. Cambazoglu, V. Murdock, R. Lempel, and F. Silvestri, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, vol. 7224, pp. 499-502, 2012.

[26] Q. V. Le and T. Mikolov, "Distributed Representations of Sentences and Documents," *ArXiv14054053 Cs*, 2014.

[27] M. Campr and K. Ježek, "Comparing Semantic Models for Evaluating Automatic Document Summarization," in *Text, Speech, and Dialogue*, P. Král and V. Matoušek, Eds. Cham: Springer International Publishing, vol. 9302, pp. 252-260, 2015.

[28] S. Lee, X. Jin, and W. Kim, "Sentiment classification for unlabeled dataset using Doc2Vec with JST," in *Proceedings of the 18th Annual International Conference on Electronic Commerce e-Commerce in Smart connected World-ICEC '16*, Suwon, Republic of Korea, pp. 1-5, 2016. Doi: 10.1145/2971603.2971631.

[29] K. Hashimoto, G. Kontonatsios, M. Miwa, and S. Ananiadou, "Topic detection using paragraph vectors to support active learning in systematic reviews," *J. Biomed. Inform.*, vol. 62, pp. 59-65, Aug. 2016.

[30] H. Palangi, et al., "Deep Sentence Embedding Using Long Short-Term Memory Networks: Analysis and Application to Information Retrieval," *IEEEACM Trans. Audio Speech Lang. Process.*, vol. 24, no. 4, pp. 694-707, Apr. 2016. Doi: 10.1109/TASLP.2016.2520371.

[31] S. Achsas and E. H. Nfaoui, "Improving relational aggregated search from big data sources using stacked autoencoders," *Cognitive Systems Research, Elsevier*, vol. 51, pp. 61-71, Oct. 2018.

[32] Q. Wu, P. Wang, C. Shen, A. Dick, and A. van den Hengel, "Ask Me Anything: Free-Form Visual Question Answering Based on Knowledge from External Sources," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Las Vegas, NV, USA, pp. 4622-4630, 2016. Doi: 10.1109/CVPR.2016.500.

[33] S. Achsas and E. H. Nfaoui, "Improving relational aggregated search from big data sources using deep learning," in *2017 Intelligent Systems and Computer Vision (ISCV)*, Fez, Morocco, pp. 1-6, 2017.

[34] N. Ben-Lhachemi and E. H. Nfaoui, "Using Tweets Embeddings for Hashtag Recommendation in Twitter," *Procedia Computer Science, ELSEVIER*, vol. 127, pp. 7-15, 2018. Doi: 10.1016/j.procs.2018.01.092.

[35] F. Kalloubi, N. El Habib, and O. El Beqqali, "Graph based tweet entity linking using DBpedia," in *2014 IEEE/ACS 11th International Conference on Computer Systems and Applications (AICCSA)*, Doha, pp. 501-506, 2014. Doi: 10.1109/AICCSA.2014.7073240.

[36] Y. Chen, L. Xu, K. Liu, D. Zeng, and J. Zhao, "Event Extraction via Dynamic Multi-Pooling Convolutional Neural Networks," in *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers),* Beijing, China, pp. 167–176, 2015. Doi: 10.3115/v1/P15-1017.

[37] M. Denil, A. Demiraj, N. Kalchbrenner, P. Blunsom, and N. de Freitas, "Modelling, Visualising and Summarising Documents with a Single Convolutional Neural Network," *ArXiv14063830 Cs Stat,* Jun. 2014.

[38] F. Meng, Z. Lu, M. Wang, H. Li, W. Jiang, and Q. Liu, "Encoding Source Language with Convolutional Neural Network for Machine Translation," *ArXiv150301838 Cs,* Mar. 2015.

[39] Y. Kim, "Convolutional Neural Networks for Sentence Classification," *ArXiv14085882 Cs,* Aug. 2014.

[40] Severyn and A. Moschitti, "Modeling Relational Information in Question-Answer Pairs with Convolutional Neural Networks," *ArXiv160401178 Cs,* Apr. 2016.

[41] E. Grefenstette, P. Blunsom, N. de Freitas, and K. M. Hermann, "A Deep Architecture for Semantic Parsing," *ArXiv14047296 Cs*, Apr. 2014.

[42] N. Kalchbrenner, E. Grefenstette, and P. Blunsom, "A Convolutional Neural Network for Modelling Sentences," *ArXiv14042188 Cs*, Apr. 2014.

[43] C. Zhu, X. Qiu, X. Chen, and X. Huang, "A Re-ranking Model for Dependency Parser with Recursive Convolutional Neural Network," *ArXiv150505667 Cs,* May 2015.

[44] T. Yamamoto, et al., "Overview of the NTCIR-12 IMine-2 Task," *Proceedings of the 12th NTCIR Conference on Evaluation of Information Access Technologies,* June 7-10, 2016 Tokyo Japan, pp. 19, 2016.

[45] T. Demeester, D. Trieschnigg, D. Nguyen, K. Zhou, and D. Hiemstra, "Overview of the TREC 2014 federated Web search track," Ghent Univ. (Belgium), 2014.

[46] B. Carterette, V. Pavlu, H. Fang, and E. Kanoulas, "Million Query Track 2009 Overview," pp. 21, 2009.

BIOGRAPHIES OF AUTHORS

Sanae Achsas is a PhD student at the University of Sidi Mohammed Ben Abdellah in Fez, Morocco. Her thesis is concerned with the improvement of the Aggregated Search with the newest Machine Learning techniques such as Deep Learning. She has published several papers in reputed journals and international conferences. She is a member of the International Neural Network Society Morocco regional chapter. Sanae can be contacted at sanae.achsas@usmba.ac.ma.

El Habib Nfaoui is currently a Professor of Computer Science at the University of Sidi Mohamed Ben Abdellah, Fez, Morocco. He received his PhD in Computer Science from the University of Sidi Mohamed Ben Abdellah, Morocco, and the University of Lyon, France, under a Cotutelle agreement (doctorate in joint-supervision), in 2008, and then his HU Diploma (accreditation to supervise research) in Computer Science, in 2013, from the University of Sidi Mohamed Ben Abdellah. His current research interests include Information Retrieval, Language Representation Learning, Web mining and Text mining, Semantic Web, Web services, Social networks, Machine learning and Multi-Agent Systems. Dr. El Habib Nfaoui has published in international reputed journals, books, and conferences, and has edited six conference proceedings and special issue books. He has served as a reviewer for scientific journals and as program committee of several conferences. He is a co-founder and an executive member of the International Neural Network Society Morocco regional chapter. He co-founded the International Conference on Intelligent Computing in Data Sciences (ICSD2017) and the International Conference on Intelligent Systems and Computer Vision (ISCV2015).