2021 IEEE/ACM 21st International Symposium on Cluster, Cloud and Internet Computing (CCGrid)

# SAED: Edge-Based Intelligence for Privacy-Preserving Enterprise Search on the Cloud

Sakib M Zobaed*, Mohsen Amini Salehi*, and Rajkumar Buyya†
*High Performance Cloud Computing (HPCC) lab,
School of Computing & Informatics, University of Louisiana at Lafayette, USA
†Cloud Computing & Distributed Systems (CLOUDS) lab, School of Computing and Information Systems
The University of Melbourne, Parkville, VIC 3010, Australia
Email: {sm.zobaed1, amini}@louisiana.edu, rbuyya@unimelb.edu.au

***Abstract*—Cloud-based enterprise search services (*e.g.,* AWS Kendra) have been entrancing big data owners by offering convenient and real-time search solutions to them. However, the problem is that individuals and organizations possessing confidential big data are hesitant to embrace such services due to valid data privacy concerns. In addition, to offer an intelligent search, these services access the user's search history that further jeopardizes his/her privacy. To overcome the privacy problem, the main idea of this research is to separate the intelligence aspect of the search from its pattern matching aspect. According to this idea, the search intelligence is provided by an on-premises edge tier and the shared cloud tier only serves as an exhaustive pattern matching search utility. We propose *Smartness at Edge* (SAED mechanism that offers intelligence in the form of semantic and personalized search at the edge tier while maintaining privacy of the search on the cloud tier. At the edge tier, SAED uses a knowledge-based lexical database to expand the query and cover its semantics. SAED personalizes the search via an RNN model that can learn the user's interest. A word embedding model is used to retrieve documents based on their semantic relevance to the search query. SAED is generic and can be plugged into existing enterprise search systems and enable them to offer intelligent and privacy-preserving search without enforcing any change on them. Evaluation results on two enterprise search systems under real settings and verified by human users demonstrate that SAED can improve the relevancy of the retrieved results by on average $\approx 24\%$ for plain-text and $\approx 75\%$ for encrypted generic datasets.**

***Index Terms*—Enterprise-search; Semantic; Edge; Context-aware**

## I. Introduction

The expeditious growth of digitalization has been producing a massive volume of data, known as *big data*, in both structured and unstructured formats. It is estimated that 95% of the generated data is in the unstructured format, produced from various sources, such as organizational documents, emails, web pages, and social networks [1]. Cloud services have been effective in relieving big data owners from the burden of maintaining these data. Recently, cloud providers began offering *enterprise search* services that enable data owners to semantically search over their datasets in the cloud. For instance, AWS has launched an enterprise search service named AWS Kendra [2] that offers real-time semantic searchability using natural language-based machine learning techniques.

Although the cloud services have been fascinating for big data owners [3], there have been numerous privacy violation incidents [4] during recent years that have made individuals and businesses with sensitive data (*e.g.,* healthcare documents) hesitant to fully embrace the data management cloud services. In one incident, confidential information of over three billion Yahoo users were exposed [5]. In another incident, information of over 14 million Verizon customer accounts were exposed from the company's cloud system [5].

Ideally, data owners desire a privacy-preserving cloud service that offers semantic and personalized searchability in a real-time manner, without overwhelming their resource-constrained (thin) client devices (*e.g.,* smartphones). A large body of research has been undertaken on privacy-preserving enterprise search services in the cloud [6], [7], [8], [9], [10] whose goals are to protect user's sensitive data from internal and external attackers. However, most of these works fall short in retrieving search results that are semantically relevant to the context and user's interest (*i.e.,* personalized search) [10], [9]. In addition, these works often rely on the client device and impose significant overhead on it to perform a secure query processing or to encrypt/decrypt user documents.

To satisfy all of the aforementioned desires of a particular user, our main idea in this research is to separate the intelligence aspect of the enterprise search from its pattern matching aspect. According to this idea, we propose to leverage on-premises edge computing [11], [12] to handle the search intelligence and user-side encryption. For that purpose, the edge-based mechanism, called *Smartness At Edge (SAED)*, is developed to extract both contextualized and personalized semantics from the search query and the user's search history as well.

Then, SAED feeds the cloud resources with proactively augmented and encrypted search queries. In this case, the high-end cloud resources are employed only to store encrypted contents and to exhaustively perform pattern matching of the fed query across the entire dataset.

Figure 1 provides a bird-eye view of the SAED mechanism. On one end, it communicates with the client device(s) to handle the security processing of the user contents and to achieve search intelligence before feeding the query set to the cloud tier. On the other end, SAED communicates with the cloud tier where an existing enterprise search service (*e.g.,* Kendra) works with the computing and storage services in the

978-1-7281-9586-5/21/$31.00 ©2021 IEEE
DOI 10.1109/CCGrid51090.2021.00046

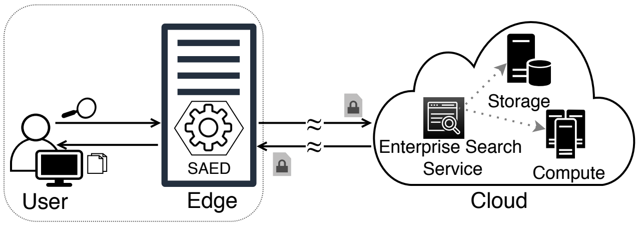


Fig. 1: Bird-eye view of SAED mechanism in a three-tier architecture to facilitate smart and privacy-preserving enterprise search service. SAED provides the secure search intelligence on the on-premises edge resources. The high-end storage and compute resources on the cloud tier are utilized by the existing enterprise search systems to exhaustively carry out pattern matching on the entire dataset.

cloud to perform exhaustive pattern matching of the encrypted query set on the uploaded dataset. Upon completion of the pattern matching process, the set of resulting documents is retrieved and ordered by SAED with respect to the user's interests. Ultimately, the ordered results are handed over to the user's device.

To identify the actual context of the query and to proactively expand it to a set of contextually-related queries, we leverage WordNet [13] that is a widely adopted knowledge-based lexical database. However, contextualizing the query cannot help in certain scenarios where the query is short and ambiguous. For instance, considering `jaguar` as the search query, it can be contextualized to both a car brand or a wild animal. For this type of queries, identifying the user's interest can complement the contextualization and navigate the search towards the semantics intended by the user (*i.e.,* achieving personalized search). For that purpose, SAED utilizes a recurrent neural network model to infer the user's interest based on his/her search history. Although proactive query expansion (*i.e.,* augmenting the user query to a set of queries) is vital to capture the search semantics, not every element of the expanded query set is equally relevant to the original query. As such, SAED assigns a weight to each expanded query that represents its semantic distance to the original query.

In summary, the contributions of the work are as follows:

- We develop the open-source SAED mechanism at the edge tier that offers personalized semantic searchability on existing cloud-based enterprise search services while maintaining data privacy.
- We propose a method to extract the context of a given search query that often appears in form of a short and incomplete sentence.
- We design a method for proactive query expansion to cover the search semantic with respect to its context.
- We develop a method based on a recurrent neural network model to personalize the search via assigning a weight to each expanded query.
- We evaluate the search accuracy and privacy of SAED via plugging it to the existing cloud-based search services.

The rest of the paper is organized as follows. In Section II, we discuss background study and related prior works. Later, we provide the architectural details of SAED mechanism in Section III. In Section IV, we discuss the pluggability of SAED in the context of AWS Kendra. We discuss results and performance analysis in Section V. Finally, Section VI concludes the paper.

## II. Background and Prior Literature

Several research works have been undertaken in semantic and/or privacy-aware search systems. Here, we introduce some notable mentions and position the contributions of SAED against them.

### A. Cloud-Based Enterprise Search Services

Cloud-based enterprise search services, such as AWS Kendra, offer semantic searchability, given that they are provided with the plain-text data. That means the semantic ability comes with the cost of compromising the users' data privacy [10], [14], [5]. This is, in fact, the trapdoor that particularly internal attackers can misuse to breach the confidentiality or even the integrity of the users' data. *It is this type of attack model that we try to make the cloud-based enterprise search services resistant against.* We note that, for encrypted datasets, the current enterprise search services cannot offer anything beyond naïve string matching.

Even for plain-text datasets, our investigations revealed that Kendra covers only ontological semantics in the search and it falls short in providing context-aware and personalized semantics. For instance, we tested Kendra to verify the ability of capturing context-aware semantics by feeding `soccer` as a query and in the result set, there were documents about `rugby`. In another test, `river bank` query returned documents about `commercial bank` that indicates the lack of context-awareness in the search.

Alternatively, SAED can offer context-aware and personalized search while maintaining data privacy. It can be plugged into any enterprise search service without enforcing any change on them and enrich their semantic search quality by incorporating context-awareness and personalization.

### B. Semantic Representation of Query Keywords

Query expansion is a process to seek keywords that are semantically related to a given query and fill the lexical gap between the user queries and the searchable documents. One of the widely-used methods of query expansion is Pseudo-Relevance Feedback (PRF) [15], [16] that extends an unsuccessful query with various related keywords and then re-ranks the search results to increase the likelihood of retrieving relevant documents. Although the PRF-based approach generally improves the retrieval effectiveness, it is sensitive to the quality of the original search results.

Latent semantic analysis [17], latent dirichlet analysis [18], and neural-based linguistic models [16], [19] are some of the query expansion methods that can obtain the semantic representation of a given query. In these methods, vectors are commonly referred to as *word embeddings* that represent

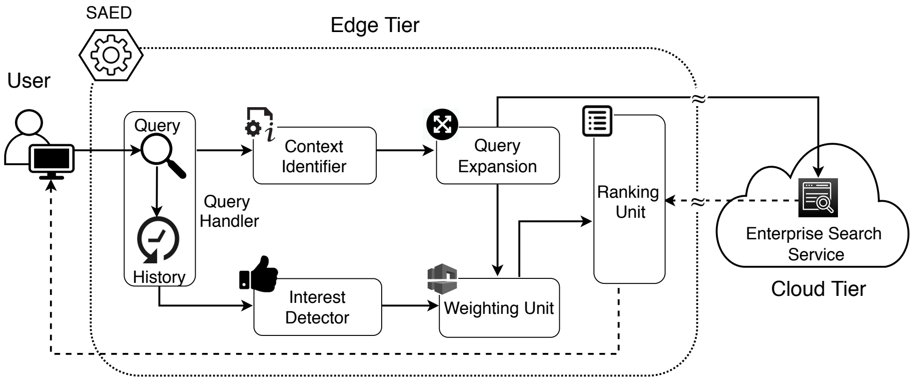


Fig. 2: Architectural overview of the SAED system within edge tier and as part of the three-tier enterprise search service. SAED provides semantic search via identifying the query context and combining that with the user's interests. Then, Query Expansion and Weighting unit of SAED, respectively, incorporate the semantic and assure the relevancy of the results. Solid and dashed lines indicate the interactions from user to the cloud tier and from the cloud tier to the user respectively.

words into a low-dimensional semantic space, where the vicinity of words demonstrates the syntactic or semantic similarity between them [20]. However, pre-trained word embedding models, such as Word2vec [20], always generate the same vector representation for an input word, regardless of the context in which the word has appeared in. Hence, if any ambiguous keyword(s) present in a query, the underlying topic of the query could not be detected.

WordNet [13] is one of the widely-used and lexically-rich resources in English that is utilized to infer the sense of ambiguous words in a given corpus. In WordNet, words containing similar meanings are grouped into synonym sets, whereby each set has a semantic and conceptual relationship with the other sets. Song *et al.* [21] and Nakade *et al.* [22] evaluate the effectiveness of utilizing WordNet for query expansion in National Institute of Standards and Technology (NIST) and Twitter datasets. They identify important key-phrases of the query and use WordNet to obtain the relevant synonym sets. Later, they utilize the synonym sets to construct the expanded query. Nevertheless, in most of the prior research on query expansion using WordNet (*e.g.,* [23]), the elements of the expanded query set are considered uniformly that undermines the relevancy and ranking of the result set.

### C. Privacy-Preserving Search Systems

In addition to plain-text data, searching is performed on privacy-preserving data ensuring negligible chances of data leakage. Therefore, various searchable encryption-based solutions are adopted to facilitate search over such data.

Few works at the time of writing have combined the ideas of semantic searching and searchable encryption. Works that attempt to provide a semantic search often only consider word similarity instead of true semantics. Li *et al.* [6] propose a system which could handle minor user typos through a fuzzy keyword search. Moataz *et al.* [24] use various stemming approaches on terms in the index and query to provide more general matching. Sun *et al.* [7] present a system that used an indexing approach over encrypted file metadata and data mining techniques to capture the semantics of queries. This approach, however, builds a semantic network only using the documents that are given to the set and only considers words that are likely to co-occur as semantically related, leaving out many possible synonyms or categorically related terms. Woodworth *et al.* propose S3BD [10], a secure semantic search system that could search semantically over encrypted confidential big data. They expand their search query by incorporating semantic data extracted blindly from an ontological network.They do not consider context-aware query expansion that created confusion for the search system while processing ambiguous or multi-context keywords in a query. To perform query processing in client devices, they end up requiring additional computational overhead in the client tier.

## III. SAED: Smart Edge-Based Enterprise Search System

### A. Architectural Overview

In this part, we provide a bird-eye view of the SAED system, that enables intelligent and secure enterprise search in the cloud. The system is structured around three tiers, shown in Figure 1, and explained as follows:

- *Client tier* (*e.g.,* smartphone, tablet) contains a lightweight application that provides a user interface for uploading documents and to search over them in the cloud. Datasets are either uploaded by the user or by the organization that owns the data.
- *Edge tier* extracts representative keywords of the documents being uploaded to the cloud tier and builds an index on the cloud tier. Upon receiving a search query from the client tier, the SAED system on the edge tier offers intelligence by considering the query semantics and the user's interest. The edge tier is located in the client's premises, hence, deemed as an honest and secure system. To offer a secure enterprise search service, the edge tier encrypts both the uploaded data and the search query.

  In addition, it decrypts the result set before delivering it back to the client tier.

- *Cloud tier* contains numerous high-end servers that are utilized for storing (encrypted) data and performing the large-scale computation required to exhaustively search against the index [9], [10]. The index can be clustered based on the underlying topics of its keywords (please refer to our prior works [10], [5] for further details).

In Figure 2, we depict the components of SAED and show the interactions between them. At first, a user-provided search query is received by the *Query Handler* that keeps track of the user's search history and initializes the *Context Identifier* unit whose job is to extract the context and disambiguate the query phrase. Then, according to the extracted context, the query is proactively expanded by the *Query Expansion* unit and a *query set* is constructed. To achieve the personalized search, the *Interest Detector* unit of SAED leverages the user's search history to recognize his/her interest and weight each element of the query set (*i.e.,* expanded queries) based on its relatedness to the user interest. Once the pattern matching phase is accomplished on the cloud tier, the resulted documents are returned to SAED on the edge tier. Next, the *Ranking Unit* utilizes the assigned weights to order the retrieved documents based on their relevance to the user's interest and generates a retrieved document list, denoted as $D_\theta$, that is sent to the user's device. In the next parts, we elaborate on each unit of the SAED system.

### B. Query Context Identification

Identifying the context of a given search phrase is vital to navigate the search to the semantics intended by the user. Considering the example of `cloud computing` as the search query, without a proper context identification the returned document set can potentially include documents about `sky` and `climate`, whereas, an efficient context identifier can recognize the right semantic and navigate the search to the topics around `distributed`, `edge`, `fog`, and `cloud computing`. In fact, identifying the context helps the Query Expansion unit to form a query set diversified around relevant keywords that semantically represent the search query and subsequently improve the relevancy of the results.

Prior context identification works (*e.g.,* [25], [26], [19]) have the following shortcomings: *first,* they often assume each keyword has the same importance in the query and recognize the query context via averaging the embeddings of its keywords. However, not all keywords in a query necessarily help in identifying the context. For example, the keyword `various` in `various cloud providers` does not bring any significance to the context and can be eliminated. *Second,* the embedding methods used by the existing works always provide the same representation for a given keyword, irrespective of the underlying context. This is particularly problematic for ambiguous keywords whose meaning vary based on the query context. For instance, the embedding of `cloud` in the aforementioned example should be different when it is used along with the `computing` as opposed to when it is used along with the `weather` in a given query. *Third,* existing methods only consider the embeddings of the common keywords, while discarding most of the name-entities (*e.g.,* names and locations) that do not exist in the vocabulary of Word2Vec [13], [27]. For instance, consider `best selling books of J.K. Rowling` as the query; `Book` and `Sell` are identified as the query context and `J.K. Rowling` is discarded. However, our analysis suggests that the context of a short query phrase often has contextual association with the discarded name-entities.

To overcome the shortcomings and identify the actual context of a given query, we propose to take a holistic approach and extract the *semantic across query keywords, proportionate to the importance of each keyword.* The main output of the Context Identification unit is a set of keywords, denoted as *C*, that collectively represent the context of the query.

Specifically, to eliminate unimportant keywords that do not contribute to the semantic of query $Q$, the Context Identification unit utilizes *Yake* [28], which is a unsupervised keyword extractor that discards unimportant keywords of the query. The remaining keywords (*i.e.,* the trimmed query, denoted as the $Q'$ set) are considered for context identification. To learn the true semantic of $Q'$, the unit leverages the Lesk algorithm [27] of WordNet to disambiguate each keyword $q \in Q'$. Lesk algorithm works based on the fact that keywords in a given sentence (query) tend to imply a certain topic. For keyword $q$, Lesk can determine its true semantics via comparing the dictionary definitions of $q$ against other keywords in $Q'$ (*i.e.,* $Q' - \{q\}$). Let $c_q$ be the set of keywords representing the context of $q$. Then, the context of $Q$ is determined as $C = \cup_{\forall q \in Q'} c_q$. Lastly, the Context Identifier recognizes name-entities from $Q$ using WordNet and considers them as part of the context, but in a separate set, denoted as $N$. The reason for considering a separate set is that we apply a different treatment on $N$ and $C$ in the other units of SAED.

**ALGORITHM 1:** Pseudo-code to detect the context of a given query in the Context Identification unit of SAED.

```
Input : query $Q$
Output: $C$: set of keywords representing context of $Q$,
        $N$: set of name-entity in $Q$
Function contextIdentification($Q$):
    $Q' \leftarrow$ extract keywords from $Q$ using Yake alg.
    foreach $q \in Q$ do
        if $q \in$ Name-entity then
            $N \leftarrow N \cup \{q\}$
        end
        else
            if $q \in Q'$ then
                $E_q \leftarrow$ define $q$ based on $Q' - q$ using Lesk alg.
                $c \leftarrow$ extract set of keywords of $E_q$ using Yake
                  alg.
                $C \leftarrow C \cup c$
            end
        end
    end
    return $C, N$
end
```

Algorithm 1 provides a pseudo-code for identifying the context of incoming query $Q$. The outputs of the pseudo-code are two sets, namely $C$ and $N$, that collectively represent the context of $Q$. In Step 2 of the pseudo-code, Yake algorithm is used to filter $Q$ by extracting its important keywords and generate the $Q'$ set. Name-entities of $Q$ are identified by checking against WordNet and form the set $N$ (Steps 4–6). Next, in Steps 8–12, for each keyword $q \in Q'$, the Lesk algorithm is employed to disambiguate $q$ and find its true definition with respect to the rest of keywords in $Q'$. Important keywords of the definitions form the context set ($C$) for $Q$.

### C. Query Expansion Unit

The *Query Expansion* unit is in charge of proactively expanding the query keywords based on their relevant synonyms that are in line with their identified context. Neglecting the query context and blindly considering all the synonyms, as achieved in [25], [26], [19], [10], leads to finding irrelevant documents. Accordingly, the unit leverages the context of $Q$ (*i.e.,* $C$ and $N$) to only find the set of synonyms, denoted as $P$, that are semantically close to the query context.

Word2Vec [20] is a shallow neural network model that can be trained to generate vector representation of keywords, such that the cosine similarity of two given keywords indicates the semantic similarity between them. Accordingly, to proactively expand each keyword $q \in Q$, the Query Expansion unit instruments Word2Vec, pre-trained with Google News dataset [29], to form the set of nominated synonyms, denoted as $s_q$. Let $s_q^i$ be a synonym of $q$ (*i.e.,* $s_q^i \in s_q$). Then, the similarity of $s_q^i$ and the query context, denoted as $sim(s_q^i, C)$, is defined based on the sum of similarities with each element of $C$, as shown in Equation 1.

$$sim(s_q^i, C) = \sum_{\forall C_j \in C} sim(s_q^i, C_j) \tag{1}$$

Then, $s_q^i$ is chosen as an element of $P$, only if it is semantically close enough to the query context. To determine the sufficient closeness, we consider $sim(s_q^i, C)$ to be greater than the mean of the pair-wise similarity across all members of $s_q$ (*i.e.,* $sim(s_q^i, C) > \mu_{\forall q \forall j}(sim(s_q^j, C))$). We note that because the elements of $C$ and $N$ represent the context of $Q$, they as well are added to $P$.

Algorithm 2 provides a high level pseudo-code for generating the expanded query set $P$. In Steps 2–7 of the pseudo-code, the synonym set for each $q$ is generated. Next, the similarity between each word $s_q^i$ and $C$ is calculated. The similarity values are used to calculate the mean similarity of all nominated queries in Step 8. In Steps 9–15, expanded query set $P$ is formed by including nominated synonyms whose semantic closeness is greater than $\mu$. Lastly, in Step 16, set $P$ is expanded by including context set and name-entities.

### D. User Interest Detection

Detecting the user's search interest is essential to deliver personalized search. In SAED, interest detection is achieved by analyzing two factors: (A) the user's search history; and (B) the user's reaction to the retrieved results of prior search queries. This can be detected based on the results chosen by the user or the time spent for browsing them.

**ALGORITHM 2:** Pseudo-code to expand query based on the context in the Query Expansion unit of SAED

```
   Input  : $Q$, $C$, $N$
   Output: $P$: the expanded query set
1  Function QueryExpansion($Q$, $C$, $N$)
2      foreach $q \in Q$ do
3          $s_q \leftarrow$ use WordNet to obtain synonym set of $q$
4          foreach $s_q^i \in s_q$ do
5              $sim(s_q^i, C) \leftarrow \sum_{\forall C_j \in C} sim(s_q^i, C_j)$
6          end
7      end
8      $\mu \leftarrow$ calculate mean $sim(s_q^j, C)$ across all $q \in Q, \forall s_q^j \in s_q$

9      foreach $q \in Q$ do
10         foreach $s_q^i \in s_q$ do
11             if $sim(s_q^i, C) > \mu$ then
12                 Add $s_q^i$ to set $P$
13             end
14         end
15     end
16     $P \leftarrow P \cup C \cup N$
17     return $P$
18 end
```

Let $\Delta'$ represent the whole resulted documents that are sent to the user and $\tau$ represent the documents where the user is interested in. We have $\tau \subseteq \Delta_{\prime}$. Accordingly, the user's interest can be derived from the topics of $\tau$. The Interest Detector unit uses an existing document classification model [30], operating based on Naïve Biased (NB) method, to determine the topics of $\tau$, denoted as $t_\tau$. We also perform majority voting on $t_\tau$ to find the user's main interest. The process is repeated to store $n$-prior search interests data of the user. The data is characterized as sequential as it is harvested from each successful search. By analyzing the user's prior search interests, the edge tier trains a recurrent neural network-based prediction model [31] that can predict the user's search interest. In case of SAED, as the data does not contain long dependency and to keep the model simple and to maintain real-timeliness, instead of a stacked (*i.e.,* deeper) model, we feed the harvested user-specific historical search data to train a many-to-one vanilla RNN model [32].

### E. Weighting Unit

Once SAED learns the user interest, the next step to accomplish a context-aware and personalized enterprise search is to determine the closeness of contextually-expanded queries (*i.e.,* elements of $P$) to the user's interest. In fact, not all expanded queries have the same significance in the interpretation of the query. Accordingly, the objective of the *Weighting unit* is defined as quantifying the closeness of each expanded query to the user's interest. Later, upon completion of the search operation on the cloud tier, the weights are used by the *Ranking unit* of SAED to prune and sort the result set.

Prior weighting schemes (*e.g.,* [9], [10], [19], [16], [26]) often use the word frequency-based approach (*e.g.,* TF-IDF [10]) and discard the user interests. Alternatively, the weighting procedure of SAED quantifies the importance of each expanded query $p \in P$ based on two factors: (A) The *type* of $p$, which means if it directly belongs to the context ($C$ and $N$ sets) or is derived from them; and (B) The *semantic similarity* of $p$ to the user interest.

In particular, those elements of $P$ that directly represent the query context or name-entities (*i.e.,* $\forall p | p \in P \cap (C \cup N)$) explicitly indicate the user's search intention, hence, weighting them should be carried out irrespective of the user interest. A deeper analysis indicates that name-entities that potentially exist in a query represent the search intention, thus, biasing the search results to them can lead to a higher user satisfaction. As such, the highest weight is assigned to $\forall p | p \in (P \cap N)$. The highest weight is determined by the domain expert, however, in the experiments we consider it as $\eta_{max} = 1$. We define the *contribution* of $q \in Q$ as the ratio of the number of keywords added to $C$ because of $q$ (denoted $C_q$) to the cardinality of $C$. Let $\eta_p$ denote the weight of $p \in P$. Then, for those elements of $P$ that are in the query context (*i.e.,* $\forall p \in (P \cap C)$), $\eta_p$ is calculated based on the contribution of the query keyword $q$ corresponding to $p$. Equation 2 formally represents how $\eta_p$ is calculated.

$$\eta_p = \frac{\eta_{max} \cdot |C_q|}{|C|} \tag{2}$$

The weight assignment for those $p$ that are derived from elements of $C$, as explained in Section III-C, (*i.e.,* $\forall p | p \in P - (C \cup N)$) is carried out via considering semantic similarity of $p$ with the user interest $\theta$. That is, $\eta_p = sim(p, \theta)$.

### F. Ranking Unit

Once the expanded query set $P$ is formed, the cloud tier performs string matching for each $p \in P$ across the index structure. We note that, if the user chooses to perform a secure search, the elements of $P$ are encrypted before delivered to the cloud tier. In addition, in our prior works [5], we proposed methods for the cloud tier to cluster the index structure and perform the pattern matching only on the clusters that are relevant to the query.

The cloud tier returns the resulted document set, denoted as $\Delta$, to the edge tier where the Ranking unit of SAED ranks them based on the relevance and the user's interest and generates a document list, called $\Delta'$ to show to the user. For a document $\delta_i \in \Delta$, the ranking score, denoted as $\gamma_i$, is calculated by aggregating the importance values of each $p \in P$ within $\delta_i$ and with respect to its weight ($\eta_p$). The importance of $p$ in $\delta_i$ is conventionally measured based on the *TF-IDF* score [33]. Accordingly, $\gamma_i$ is formally calculated based on Equation 3.

$$\gamma_i = \sum_{\forall p \in P} \Big( \eta_p \cdot TFIDF(p, \delta_i) \Big) \tag{3}$$

The TF-IDF score of $p$ in $\delta_i$ is defined based on the frequency of $p$ in $\delta_i$ versus the inverse document frequency of $p$ across all documents in $\Delta$. Details of calculating the tf-idf score can be found in [33].

Once the Ranking unit calculates the ranking score for all $\delta_i \in \Delta$, then the documents are sorted in the descending order based on their ranks and thus, the document list $\Delta'$ are formed with each$\delta_i$ and displayed to the user.

## IV. SAED As a Pluggable Module to Enterprise Search Solutions

The advantage of SAED is to be independent from the enterprise search service deployed on the cloud tier. That is, using SAED neither interferes with nor implies any change on the cloud-based enterprise search service. SAED can be plugged into any enterprise search solution. It provides the search smartness on the on-premises edge tier and leaves the cloud tier only for large-scale pattern matching. The whole SAED solution reforms the enterprise search to be semantic, personalized, and confidential services.

In this work, we set SAED to work both with AWS Kendra and S3BD. In the case of using AWS Kendra, the Query Expansion unit sends the expanded query set $P$ to Kendra to search each keyword $p$ against the dataset on the Amazon cloud. The resulted documents are received by SAED and ranked before being delivered to the client tier. In the implementation, we only show top 10 documents from the resulted list to the user. Similarly, we plugged SAED to S3BD to perform confidential semantic search on the cloud. Because S3BD maintains an encrypted index structure that has to be traversed against each search query, the elements of $P$ had to be encrypted before handing them over to the cloud tier. We also verified SAED when it is used along with AWS Kendra where the dataset was encrypted. We noticed that SAED can achieve smart search even when Kendra is set to work with encrypted dataset. The performance measurement and analysis of using SAED along with AWS Kendra and S3BD are elaborated in the next Section.

## V. Performance Evaluation

### A. Experimental Setup

We have developed a fully working version of SAED and made it available publicly in our Github[1] page. To conduct a comprehensive performance evaluation of SAED on the enterprise search solutions, we developed it to work with both S3BD [10] and AWS Kendra [2]. S3BD already has the query expansion and weighting mechanisms, but we deactivated them and set it to use the expanded queries generated by SAED. In the experiments, the combination of SAED and S3BD is shown as SAED+S3BD. Likewise, the combination of SAED and AWS Kendra is shown as SAED+Kendra.

We evaluated SAED using two different datasets, namely `Request For Comments (RFC)` and `BBC` that have distinct properties and volume. The reason we chose the `RFC` dataset is that it is domain-specific and includes $4,951$ documents about the Internet and wireless communication network.

[1] https://github.com/hpcclab/SAED-Security-At-Edge

TABLE I: Benchmark search queries developed for the RFC and BBC datasets.

| **BBC Dataset** | **RFC Dataset** |
|---|---|
| European Commission (EC) | Network Information (NI) |
| Parliament Archives (PA) | Host Network Configuration (HNC) |
| Top Camera Phones 2020 (TCP) | Data Transfer (DT) |
| Credit Card Fraud (CCF) | Service Extension(SE) |
| Animal Welfare Bill (AWB) | Transport Layer (TL) |
| Piracy and Copyright Issues (PCI) | Message Authentication (MA) |
| Car and Property Market (CPM) | Network Access (NA) |
| Rugby Football League (RFL) | Internet Engineering (IE) |
| Opera in Vienna (OV) | Fibre Channel (FC) |
| Windows Operating System (WOS) | Streaming Media Service (SMS) |

Alternatively, the BBC dataset is more diverse. It includes $2,224$ news documents in five distinct categories, including politics, entertainment, business, sports, and technology.

To conduct a comprehensive evaluation, we used both systematic metrics and human-based feedback as elaborated in Section V-C. We deployed and experimented SAED on a Virtual Machine (VM) within our local edge computing system. The VM had two 10-core 2.8 GHz E5 Xeon processors with 64 GB memory and Ubuntu 18.4 operating system.

### B. Benchmark Queries

The datasets that we use to carry out the experiments are not featured with any benchmark. Therefore, we required to develop benchmark queries for the datasets before evaluating the performance of SAED. We developed 10 benchmark queries, shown in Table I, for each one of the two datasets. The benchmark queries are proactively designed to explore the breadth and depth of the datasets in question. In addition, some of the queries intentionally contain ambiguous keywords to enable us examining the context detection capability of SAED. For the sake of brevity, we provide one acronym for each benchmark query (see Table I). For each benchmark query, we collected at most the top-20 retrieved documents. Then, the quality of the retrieved documents were measured via both automated script and human-based users.

### C. Evaluation Metrics

We have to measure the search relevancy metric to understand how related the resulted documents are with respect to the user's query and how they meet the his/her interests. For the measurement, we use TREC-Style Average Precision (TSAP) score, described by Mariappan *et al.* [34]. TSAP provides a qualitative score in a relatively fast manner and without the knowledge of the entire dataset [10]. It works based on the precision-recall concept that is commonly used for judging text retrieval systems. The TSAP score is calculated based on $\sum_{i=0}^{N} r_i/N$, where $r_i$ denotes score for $i^{th}$ retrieved document and $N$ denotes the cutoff number (total number of retrieved documents). Since we consider $N = 10$, we call the scoring metric as *TSAP@10*.

To determine $r_i$ for retrieved document $\delta'_i \in \Delta'$, we conducted a human-based evaluation. We engaged five volunteer students to judge the relevancy of each retrieved document. For every search query, the volunteers labeled each retrieved document as highly relevant, partially relevant, or irrelevant. After performing majority voting based on the provided responses for document $i$, the value of $r_i$ is determined as follows:

- $r_i = 1/i$ if a document is highly relevant
- $r_i = 1/2i$ if a document is partially relevant
- $r_i = 0$ if a document is irrelevant

We report TSAP@10 score to show the relevancy of results for each benchmark query. In addition, mean TSAP score is reported to show the overall relevancy across each dataset. As we set the top 10 documents to be retrieved for each search, the highest possible for *TSAP@10* score can be 0.292 [34].

In addition to the TSAP score, we measure *Mean F-1* score too to compare the search quality offered by the SAED-plugged enterprise search solutions against the original enterprise search solutions (*i.e.,* without SAED in place). The F-1 score maintains a balance between the precision and recall metrics, which is useful for unstructured datasets with non-uniform topic distribution.

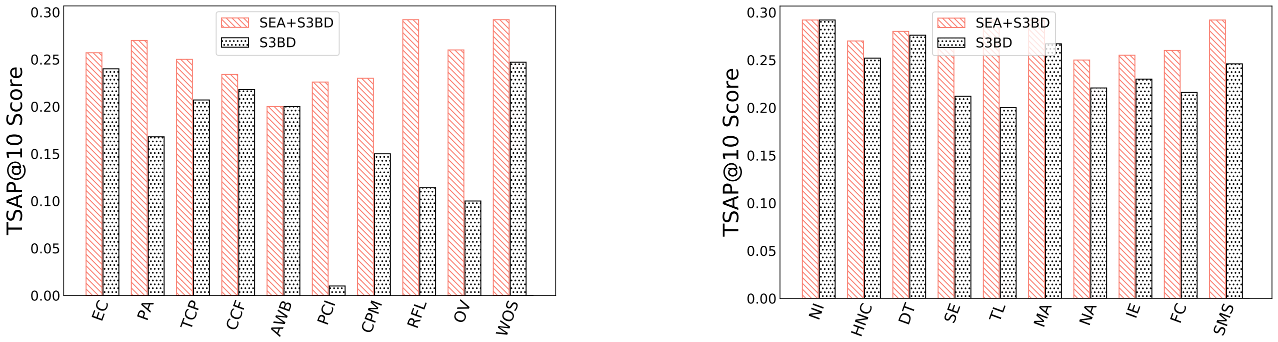
### D. Evaluating Search Relevancy

The purpose of this experiment is to evaluate the search relevancy of enterprise search systems that have SAED plugged into them and compare them against the original (unmodified) systems. To evaluate the personalized search, we set (assumed) technology as the user's interest for both datasets. We note that, in this part, the enterprise search solutions (S3BD and AWS Kendra) are set to work in the plain-text datasets.

***S3BD vs SAED+S3BD:*** Figure 3a shows the TSAP@10 score for the RFC and BBC datasets for the original S3BD and SAED+S3BD. The horizontal axes in both subfigures show the benchmark queries and the vertical axes show the search relevancy based on the TSAP@10 score.

In both Figure 3a and 3b, we observe that for all queries in both datasets, SAED+S3BD outperforms the S3BD system. In addition, we observe that S3BD produces less relevant results for the BBC dataset compared to the RFC dataset. This is because, unlike the RFC dataset, in several cases, the exact keywords of the benchmark queries do not exist in the BBC dataset. The worst case of these issues has occurred for the PCI query in S3BD, because its query expansion procedure could not capture the complete semantics. In contrast, SAED+S3BD is able to handle the cases where the exact keyword does not exist in the dataset, thus, we see that it yields to a remarkably higher relevancy.

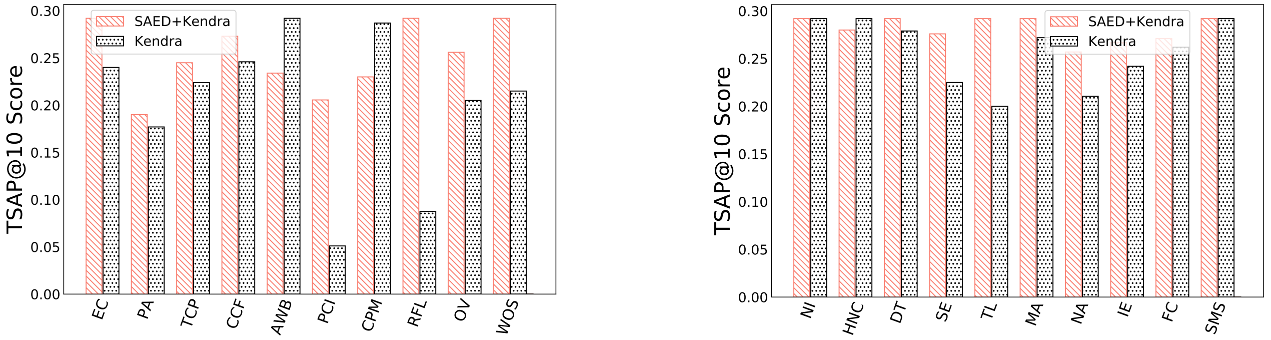
Even if we consider PCI as an outlier and exclude that from the analysis, in Figure 3a, we still notice that the TSAP@10 score of SAED+S3BD is on average $41.2\%$ higher than S3BD. Although the difference between S3BD and SAED+S3BD is less significant for the RFC dataset (in Figure 3b), we still notice some $17\%$ improvement in TSAP@10 score. This is because RFC is a domain-specific dataset and the exact keywords of queries can be found in the dataset, hence, making use of smart methods to extract the semantic is not acute to earn relevant results. From these results, we can conclude that



(a) BBC dataset

(b) RFC dataset

Fig. 3: Comparing TSAP@10 scores of SAED+S3BD and S3BD systems. Horizontal axes show the benchmark queries.



(a) BBC dataset

(b) RFC dataset

Fig. 4: Comparing TSAP@10 scores obtained from SAED+Kendra versus AWS Kendra in searching benchmark queries.

SAED can be specifically effective for generic datasets where numerous topics exist in the documents.

***AWS Kendra vs SAED+Kendra:*** In Figures 4a and 4b, we report TSAP@10 score obtained from AWS Kendra versus SAED+Kendra for BBC and RFC datasets, respectively. Specifically, in Figure 4a (BBC dataset), a significant improvement (on average 26.5%) is noticed in the TSAP@10 score of SAED+Kendra. However, unlike SAED+S3BD, SAED+Kendra does not beat Kendra for all the queries. The reason Kendra outperforms SAED+Kendra for `AWB` and `CPM` queries is that SAED injects extra keywords and sends the expanded query set to AWS Kendra. Then, Kendra returns documents that are related to the queries and to the expanded keywords. We realized that the Ranking unit of SAED occasionally prioritizes documents that include keywords of the expanded queries instead of those with the query keywords.

Similar to the S3BD experiment, we observe that the relevancy resulted from Kendra and SAED+Kendra is less significant for RFC. However, we still obtain around 12% improvement in TSAP@10 score according to Figure 4b.

### *E. Relevancy of Privacy-Preserving Enterprise Search*

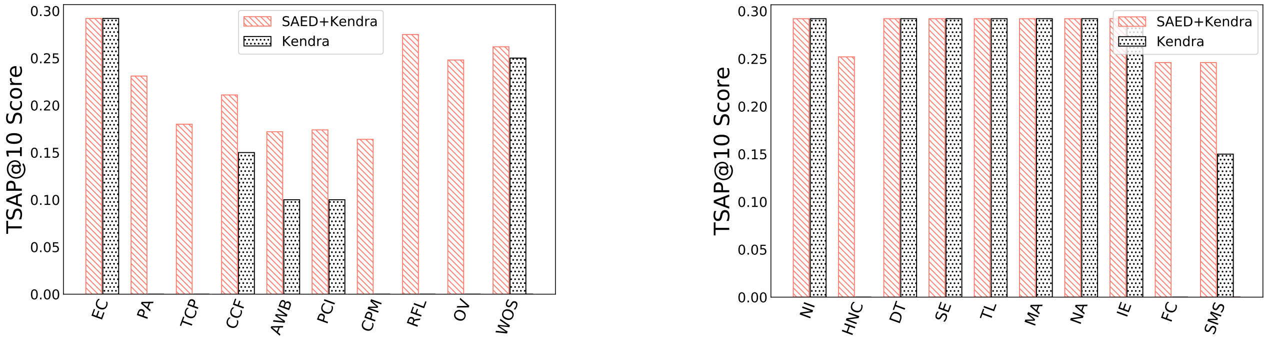
To examine the efficiency of SAED for privacy-preserving enterprise search systems, we conducted experiments using encrypted BBC and RFC datasets. The encrypted datasets were uploaded to the cloud tier and the expanded queries were also encrypted and searched on the cloud tier via Kendra.

We use the TSAP@10 score, as shown in Figure 5a and 5b, for the BBC and RFC datasets, respectively. Figure 5a indicates that SAED+Kendra substantially outperforms Kendra for all the benchmark queries. We can see that for encrypted dataset Kendra cannot do anything except pattern matching and returning documents that exactly include the encrypted query. Therefore, searching for several queries (*e.g.,* `PA`,`TCP`, `CPM`, etc.) does not retrieve any documents. We notice that, in both systems, the highest TSAP@10 score is in searching `EC`. The reason is the high number of documents in BBC that contain the exact phrase `European commission`.

The reported TSAP@10 scores for the RFC dataset in Figure 5b shows a clear improvement in compared with the BBC dataset. We observe that seven out of ten queries provide an equal TSAP@10 scores in both systems. The reason that makes Kendra competitive to SAED+Kendra is the exact availability of the benchmark queries in RFC. However, for `HNC` and `FC`, the exact query keywords are not present in the dataset, hence, Kendra fails to find any results.

### *F. Discussion of the Relevancy Results*

In Table II, we report *mean F-1* and *mean TSAP@10* scores for the SAED-plugged enterprise search systems along with



(a) Encrypted BBC dataset

(b) Encrypted RFC dataset

Fig. 5: Comparing TSAP@10 scores obtained from SAED+Kendra vs AWS Kendra systems in the encrypted domain.

their original versions upon utilizing the datasets both in the plain-text and encrypted forms. From the table, we notice that, regardless of the enterprise search system being employed, a higher search relevancy is consistently achieved for the RFC dataset as opposed to the BBC dataset.

The search relevancy is consistently improved when SAED+Kendra is used and it provides on average of 23% improvement in mean F-1 score and 21% in the mean TSAP@10 score. Although original S3BD is the underperformer, using SAED+S3BD improves its mean F-1 and mean TSAP@10 scores by on average of 40% and 32%, respectively.

| | BBC | | RFC | |
|---|---|---|---|---|
| **Systems** | **Mean F-1** | **Mean TSAP@10** | **Mean F-1** | **Mean TSAP@10** |
| S3BD | 0.50 | 0.17 | 0.80 | 0.24 |
| SAED+S3BD | 0.82 | 0.25 | 0.92 | **0.28** |
| Kendra | 0.67 | 0.20 | 0.88 | 0.26 |
| SAED+Kendra | **0.90** | **0.27** | **0.93** | **0.28** |
| Kendra (Encry.) | 0.31 | 0.09 | 0.75 | 0.22 |
| SAED+Kendra (Encry.) | 0.73 | 0.22 | 0.90 | 0.27 |

TABLE II: Comparing the mean F-1 and the mean TSAP@10 scores obtained from SAED-plugged enterprise search systems versus their original forms. The highest resulted scores are shown in bold font.

In the encrypted domain, we notice that SAED+Kendra offers a substantially higher (up to 130%) search relevancy for BBC dataset. As the exact keywords of the given search queries are not present in the encrypted form of BBC dataset, AWS Kendra fails to perform semantic search, rather does only a pattern matching, which makes it an underperformer for this dataset. On the other hand, search relevancy is improved for RFC dataset since mean F-1 and mean TSAP@10 scores are improved by at least 20%. This is because, most of the queries are present exactly in the dataset and Kendra retrieves most of the relevant documents by relying only on pattern matching.

### G. Evaluating the Search Time

Figure 6 presents the total incurred search time of the experimented queries for each dataset. The search time is calculated as the summation of the elapsed time taken by a query to be processed (*e.g.,* expansion, weighting) and turnaround time until the result set is received. To eliminate the impact of any randomness in the computing system, we searched each set of experimented queries 10 times and reported the results in the form of box plots. The figure indicates that S3BD system has the highest search time overhead for both datasets which could impact real-time searchability in case of big data. SAED+S3BD incurs less query processing time overhead compared to the original (unmodified) S3BD system.

On the other hand, AWS Kendra causes the lowest time overhead for both datasets compared to SAED+Kendra. SAED+Kendra causes around 4 times more time overhead compared to original Kendra. However, in the prior set of experiments, we determine that SAED+Kendra achieves a substantially higher search relevancy for most of the queries and, particularly, for datasets with privacy constraints.

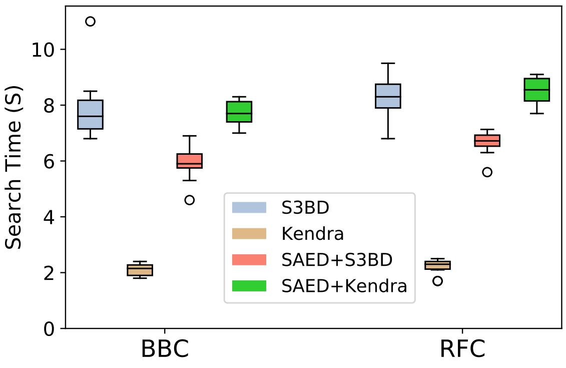


Fig. 6: Search time comparison among S3BD, Kendra, SAED+S3BD, and SAED+Kendra systems.

## VI. Conclusions and Future Work

A context-aware, personalized, and privacy-preserving enterprise search service is the need of the hour for data owners who wish to use cloud services. Our approach to address this demand was to separate the search intelligence and privacy aspects from the pattern matching aspect. We developed SAED that achieves privacy and intelligence at the edge tier and leaves the large-scale pattern matching for the cloud tier. SAED is pluggable and can work with any enterprise search solution (*e.g.,* AWS Kendra and S3BD) without dictating any change on them. Utilizing edge computing on the user's premises preserves the user's privacy and makes SAED a lightweight solution. Leveraging recurrent neural network-based prediction models, WordNet database, and Word2Vec, SAED proactively expands a search query in a proper contextual direction and weights the expanded query set based on the user's interest. In addition, SAED provides the ability to perform semantic search while the data are stored in the encrypted form on the cloud. In this case, the existing enterprise search solutions just perform the pattern matching without knowing the underlying data. Evaluation results, verified by human users, show that SAED can improve the relevancy of the retrieved results by on average $\approx 24\%$ for plain-text and $\approx 75\%$ for encrypted generic datasets. There are several avenues to improve SAED. One avenue is to cover domain-specific and trendy keywords. Another avenue is to make the SAED flexibly deployed on various devices. For instance, when the user is on the move and does not have access to the edge, SAED should shrink to the bare minimum search intelligence and vice versa.

## Acknowledgments

The research was supported by Chameleon cloud, Amazon Cloud research credit, and NSF grant IRES-1940619. We are thankful to the students helped creating the benchmark queries and verified the relevancy of the search results.

## References

[1] G. Bello-Orgaz, J. J. Jung, D. Camacho, Social big data: Recent achievements and new challenges, Journal of Information Fusion 28 (2016) 45–59.

[2] Amazon Kendra, www.aws.amazon.com/kendra (Accessed April 10, 2020).

[3] D. G. Samani, C. Denninnart, J. Bacik, M. A. Salehi, The art of cpu-pinning: Evaluating and improving the performance of virtualization and containerization platforms, Proceedings of the 49th International Conference on Parallel Processing (August 2020).

[4] The 15 biggest data breaches of the 21st century, www.csoonline.com/article/2130877/data-breach/the-biggest-data-breaches-of-the-21st-century.html (Accessed November 21, 2019).

[5] S. Zobaed, S. Ahmad, R. Gottumukkala, M. A. Salehi, Clustcrypt: Privacy-preserving clustering of unstructured big data in the cloud, in: Proceedings of the 21st International Conference on High Performance Computing and Communications (HPCC), 2019, pp. 609–616.

[6] J. Li, Q. Wang, C. Wang, N. Cao, K. Ren, W. Lou, Fuzzy keyword search over encrypted data in cloud computing, in: Proceedings of the 29th International Conference on Computer Communications, INFOCOM '10, 2010, pp. 1–5.

[7] X. Sun, Y. Zhu, Z. Xia, L. Chen, Privacy preserving keyword based semantic search over encrypted cloud data, Journal of Security and Its Applications 8 (3) (May 2014).

[8] M. Amini Salehi, T. Caldwell, A. Fernandez, E. Mickiewicz, D. Redberg, E. W. D. Rozier, S. Zonouz, RESeED: Regular Expression Search over Encrypted Data in the Cloud, in: Proceedings of the 7th International Cloud conference, Cloud '14, 2014, pp. 673–680.

[9] J. Woodworth, M. A. Salehi, V. Raghavan, S3c: An architecture for space-efficient semantic search over encrypted data in the cloud, in: Proceedings of the 4th International Conference on Big Data, Big Data'16, 2016, pp. 3722–3731.

[10] W. Jason, M. A. Salehi, S3BD: secure semantic search over encrypted big data in the cloud, Journal of Concurrency and Computation:Practice and Experience (CCPE) 28 (11) (December 2018).

[11] S. Zobaed, M. A. Salehi, Big data in the cloud, in: L. A. Schintler, C. L. McNeely (Eds.), Encyclopedia of Big Data, Springer, 2018.

[12] F. N. Nur, S. Islam, N. N. Moon, A. Karim, S. Azam, B. Shanmugam, Priority-based offloading and caching in mobile edge cloud, Journal of Communications Software and Systems 15 (2) (2019) 193–201.

[13] G. A. Miller, Wordnet: a lexical database for english, Journal of Communications of the ACM 38 (11) (1995) 39–41.

[14] S. Ahmad, S. Zobaed, R. Gottumukkala, M. A. Salehi, Edge computing for user-centric secure search on cloud-based encrypted big data, in: Proceedings of the 21st International Conference on High Performance Computing and Communications (HPCC), 2019, pp. 662–669.

[15] F. Liang, R. Qiang, J. Yang, Exploiting real-time information retrieval in the microblogosphere, in: Proceedings of the 12th ACM/IEEE-CS joint conference on Digital Libraries, 2012, pp. 267–276.

[16] Y. Wang, H. Huang, C. Feng, Query expansion with local conceptual word embeddings in microblog retrieval, IEEE Transactions on Knowledge and Data Engineering (October 2019).

[17] S. Deerwester, S. T. Dumais, G. W. Furnas, T. K. Landauer, R. Harshman, Indexing by latent semantic analysis, Journal of the American society for information science 41 (6) (1990) 391–407.

[18] K. Albishre, Y. Li, Y. Xu, Effective pseudo-relevance for microblog retrieval, in: Proceedings of the Australasian Computer Science Week Multiconference, 2017, pp. 1–6.

[19] F. Diaz, B. Mitra, N. Craswell, Query expansion with locally-trained word embeddings, in: Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), 2016, pp. 367–377.

[20] T. Mikolov, K. Chen, G. Corrado, J. Dean, Efficient estimation of word representations in vector space, www.pub-tools-public-publication-data.storage.googleapis.com/pdf/41224.pdf (2013).

[21] M. Song, I.-Y. Song, X. Hu, R. B. Allen, Integration of association rules and ontologies for semantic query expansion, Journal of Data & Knowledge Engineering 63 (1) (2007) 63–75.

[22] V. Nakade, A. Musaev, T. Atkison, Preliminary research on thesaurus-based query expansion for twitter data extraction, in: Proceedings of the Southeast Regional Conference, ACMSE'18, 2018, pp. 1–4.

[23] C. H. Leung, Y. Li, A. Milani, V. Franzoni, Collective evolutionary concept distance based query expansion for effective web document retrieval, in: Proceedings of the International Conference on Computational Science and Its Applications, 2013, pp. 657–672.

[24] T. Moataz, A. Shikfa, N. Cuppens-Boulahia, F. Cuppens, Semantic search over encrypted data, in: Proceedings of International Conference on Telecommunications (ICT), 2013, pp. 1–5.

[25] A. Silva, M. Mendoza, Improving query expansion strategies with word embeddings, in: Proceedings of the ACM Symposium on Document Engineering, 2020.

[26] S. Kuzi, A. Shtok, O. Kurland, Query expansion using word embeddings, in: Proceedings of the 25th international on conference on information and knowledge management, 2016.

[27] C. Fellbaum, Wordnet: An electronic lexical resource, in: The Oxford Handbook of Cognitive Science, Routledge, 2017, pp. 301–314.

[28] R. Campos, V. Mangaravite, A. Pasquali, A. M. Jorge, C. Nunes, A. Jatowt, YAKE! collection-independent automatic keyword extractor, in: Proceedings of the 40th European Conference on Information Retrieval, 2018, pp. 806–810.

[29] A. Khatua, A. Khatua, E. Cambria, A tale of two epidemics: Contextual word2vec for classifying twitter streams during outbreaks, Journal of Information Processing & Management 56 (1) (2019) 247–257.

[30] Z. Kastrati, A. S. Imran, S. Y. Yayilgan, The impact of deep learning on document classification using semantically rich representations, Journal of Information Processing & Management 56 (5) (2019) 1618–1632.

[31] X. Pang, Y. Zhou, P. Wang, W. Lin, V. Chang, An innovative neural network approach for stock market prediction, The Journal of Supercomputing 76 (3) (2020) 2098–2118.

[32] Vanilla recurrent neural network, http://calvinfeng.gitbook.io/machine-learning-notebook/supervised-learning/recurrent-neural-network/recurrent_neural_networks (Accessed September,2020).

[33] F. Viegas, S. Canuto, C. Gomes, W. Luiz, T. Rosa, S. Ribas, L. Rocha, M. Gonçalves, Cluwords: exploiting semantic word clustering representation for enhanced topic modeling, in: Proceedings of the 12th International Conference on Web Search and Data Mining, 2019, pp. 753–761.

[34] A. K. Mariappan, R. M. Suresh, V. S. Bharathi, A comparative study on the effectiveness of semantic search engine over keyword search engine using tsap measure, Journal of Computer Applications EGovernance and Cloud Computing Services (2012) 4–6.