Computing and Informatics, Vol. 38, 2019, 851–882, doi: 10.31577/cai_2019_4_851

PROCESS MATCHING: PERFORMANCE TRADE-OFF BETWEEN SUMMARY AND FULL-LENGTH DESCRIPTIONS

Syed Irtaza Muzaffar, Khurram Shahzad, Faisal Aslam
Madiha Khalid, Kamran Malik

Punjab University College of Information Technology
University of the Punjab
The Mall, Lahore, Pakistan
e-mail: {phdcsf17m503, khurram, faisal.aslam, madiha.khalid, kamran.malik}@pucit.edu.pk

Abstract. Business process models are used by modeling experts to concisely depict the workflow of an organization that plays a pivotal role in the development of ERP systems. A growing number of organizations also maintain the textual process descriptions of these process models as the descriptions are understandable across the board. A recent study has revealed that these textual descriptions can also be used for an accurate process model search. However, the use of textual descriptions is a resource-intensive task due to the sheer size of the descriptions. To that end, in this paper, we have proposed an approach that relies on the use of summary textual descriptions, instead of full-length descriptions, to enhance the performance of process matching. To evaluate the proposed approach, we have used four diverse text summarization techniques, including a state-of-the-art deep learning based technique, for generating summary descriptions, and seven text-matching techniques for finding relevant process specifications. Our empirical study has established that the Vector Space Model is the most effective technique for process matching. Furthermore, the use of Lingo generated summaries, at a compression rate of 50 %, can achieve a higher efficiency as well as effectiveness than the full-length textual process descriptions.

Keywords: Information retrieval systems, process retrieval, text-matching, summary-full description for process matching

1 INTRODUCTION

Business process models are widely established as key artifacts to visually represent, analyze, and enhance the business operations of an enterprise [1, 2]. As these artifacts are developed by modeling experts, they may not be readily understandable by all the stakeholders. In particular, the business users who actually execute processes have difficulties with reading and comprehension of models due to their limited knowledge of process modeling [3, 4]. Therefore, several studies have emphasized maintaining textual descriptions alongside process models. The availability of textual process descriptions has also prompted the use of these descriptions for process model validation [5], inconsistencies detection [6], and process matching [7].

A recent study [8], has proposed to employ the combination of process models with textual descriptions of its activities to enhance the accuracy of querying processes from a process repository. In contrast, another notable study has proposed the use of textual descriptions as an alternative to process models [7]. The two approaches have established that the use of textual descriptions enhances the effectiveness of process matching. However, we contend that the use of textual descriptions could be a time-consuming task due to the sheer size of these descriptions. For instance, an Austrian bank's process collection has 119 textual descriptions of processes with an average length of 13 130 words, and the longest description is composed of 60 558 words [9]. In the presence of such large textual descriptions, the use of full-length textual descriptions may impede the efficiency of the process matching.

To enhance the performance of matching, in this paper, we have proposed a summary description-based approach that relies on the compressed versions of full-length textual descriptions. The significantly reduced size of the summary descriptions should enhance the efficiency of process matching. However, we recognize that such a reduction in the descriptions may impede the accuracy of matching. Therefore, in this paper, we analyze the trade-offs between the summarized descriptions and full-length textual descriptions in terms of efficiency and effectiveness. As far as we are aware, no study has been conducted to evaluate the effectiveness of summary descriptions for process matching. In particular, the key contributions to this paper are as follows:

Proposed Approach: We have proposed a process matching approach that takes an input textual or model-based specifications of a process and returns the specifications of relevant processes. In essence, the approach generates summary textual descriptions by using text summarization techniques. Subsequently, it computes the similarity between query-source pairs using text-matching techniques and returns the specifications of the relevant processes.

Corpora Generation: We have generated corpora of 669 full-length textual descriptions and their summary textual descriptions at five different compression rates, using four diverse summarization techniques, including a state-of-the-art deep learning based approach. Thus, in total, we have generated $(1 + 4 \times 5 =)$ 21 corpora of textual process descriptions. The techniques used to generate summary descriptions are: a) TF-IDF, which generated a collection of important words, b) LexRank, which employs a graph-based approach to rank, and subsequently choose sentences of higher rank [10], c) Lingo, which employs a clustering-based approach to identify the sentences that include key phrases of the descriptions, and d) K-means clustering with skip-thought embeddings which relies on the use of deep learning based technique to identify key sentences for inclusion in a summary description.

Analysis of Summaries: We have compared the summary textual descriptions generated by all the four summarization techniques, TF-IDF, LexRank, Lingo, and K-means clustering with skip-thought embeddings. To this end, we first generated the pairs of these summary descriptions at each compression rate. Subsequently, we employed two established text-matching techniques, n-gram overlap, and Longest Common Subsequence, to compute the similarity between each pair of textual description. The results have been used to provide valuable insights into the generated summaries.

Efficiency and Effectiveness Experiments: We have performed numerous experiments for full-length textual descriptions and summary descriptions generated by each summarization technique. For that, we have used seven text-matching techniques for each type of experiments. The results have been analyzed to empirically establish the benefit of using summary descriptions as an alternative to the full-length descriptions. Furthermore, the trade-offs between efficiency and effectiveness have been analyzed.

The rest of the paper is organized as follows: Section 2 provides an overview of the proposed approach. Section 3 presents the procedure we have used for generating full and summary textual descriptions corpora and the specifications of the corpora. Section 4 introduces the text-matching techniques that are used for experimentation. Section 5 analyzes the similarity between the summary descriptions generated by the four summarization techniques. Section 6 presents the experimental setup. Analysis of results, as well as the trade-off between efficiency and effectiveness, is presented in Section 7. Related work is presented in Section 8. Finally, conclusions are drawn in Section 9.

2 CONCEPTUAL APPROACH: AN OVERVIEW

In this section, we present an overview of the proposed approach which relies on the use of summary textual descriptions for retrieving the desired specifications of relevant processes, instead of full-length textual process descriptions. The reason to use summary descriptions over full-length descriptions stems from the potential size of process descriptions, which are particularly sizeable for end-to-end processes. For instance, a recent study [9] has highlighted that the collection of an Austrian bank contains 119 real-world processes having an average length of 13 130 words and a maximum length of 60 558 words. The presence of such sizeable textual descriptions makes process matching a resource-intensive task which impedes the efficiency of matching. A conceptual overview of the proposed approach is presented in Figure 1. As depicted in the figure, the repository is composed of a source collection of process models and their corresponding corpus of textual process descriptions. Furthermore, a mapping can be defined between the two types of process specifications. While recent notable studies [8, 7] advocate that keeping textual process descriptions alongside process models increases the comprehension of business operations of enterprises among users, we have proposed to use summary textual description queries for process matching.

Our approach relies on the use of an automatic approach to generate textual descriptions of a process model using Natural Language Generation System (NLGS) [5]. As far as we are aware, NLGS is the only available tool that can automatically and comprehensively generate textual descriptions of a process model. It uses a well-established technique that takes a process model in the JSON format as input and generates its textual process description. In particular, the input to our proposed approach could be a model-based or textual specification as a query, whereas, the output is the specifications of relevant business processes. The approach involves three main steps: generating textual description, finding similar process descriptions, and returning specifications of the relevant processes.

In the first step, if the input query is a model-based specification, the textual process description of the query is generated using the NLGS. Secondly, the generated textual description of the input query process is compared with the textual descriptions of all the source process models available in the repository, and a similarity score of each query-source pair is computed using a text-matching technique. Subsequently, a ranked list of processes is generated, where the processes are sorted in the descending order of the similarity scores. Finally, the specifications of the top K source processes are returned. In the rest of the paper, we have used summary textual process descriptions of query processes for the process matching experiments and compared its performance with the corresponding full-length textual descriptions.

For a formal specification of the proposed approach, let p_i be a business process whose model-based specification is represented by M_{pi}. A function β can be defined that generates textual process description D_{pi} of the process model M_{pi}. Formally, $D_{pi} = \beta(M_{pi})$. Furthermore, consider P_M be a collection of process models in a repository, and C_D be a corpus of the corresponding textual descriptions. A function γ can be defined that maps textual descriptions of business processes to the model-based specifications of the processes. Formally, $\gamma : M_{pi} \rightarrow D_{pi}$. For an input query process Q_p, the relevant process specifications the query can be extracted using Algorithm 1.

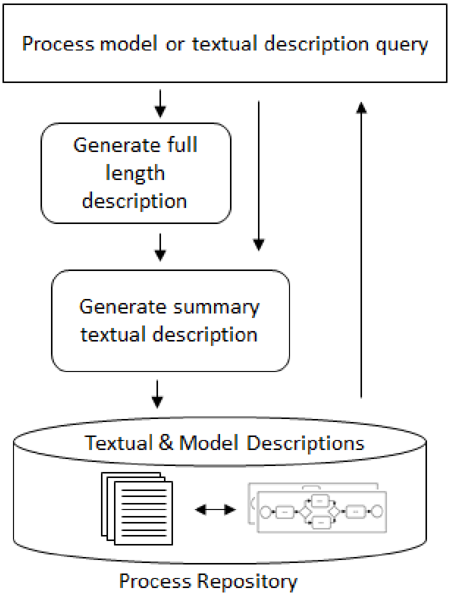

Process model or textual description query
Generate full length description
Generate summary textual description
Textual & Model Descriptions
Process Repository

Figure 1. Overview of the proposed approach

Algorithm 1 Summary description for process matching

```
Input: $Q_p$, $C_D$, $P_M$          /* query, process corpus, model collection */
Output: $List[P_{ID},\ int\ y]$

$sim_{Score} = 0$
 $L_1 = List[x,\ y]$
if $(Q_p = M_{Q_p})$ then
    $\beta(Q_p) \rightarrow D_{Q_p}$
      $\alpha(D_{Q_p}) \rightarrow SD_{Q_p}$
else
    $\alpha(Q_p) \rightarrow SD_{Q_p}$
end
while $D_{p_i} \in C_D$ do
    $sim_{Score} = similarity(SD_{Q_p}, D_{p_i})$
    $L_1.append(D_{p_i}, sim_{Score})$
end
$sort_{dec}(L_1, sim_{Score})$
 return $L_1$
```

3 CORPORA GENERATION

The investigation of performance trade-off between summary and full-length textual descriptions require three artifacts: a) a corpus of full-length textual descriptions of process models, b) corpora of query descriptions used for the matching experiments, and c) corpora of summary textual descriptions of business processes. An overview of the process that we have used for generating these corpora is shown in Figure 2.

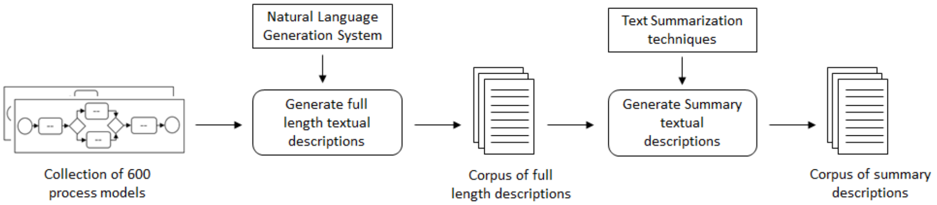

Natural Language Generation System
Text Summarization techniques
Generate full length textual descriptions
Generate Summary textual descriptions
Collection of 600 process models
Corpus of full length descriptions
Corpus of summary descriptions

Figure 2. Comparison of textual process descriptions

3.1 Generation of Full-Length Textual Descriptions

We have access to a collection of 669 process models that are designed using Business Process Model and Notation (BPMN), which is the most prominent process modeling language. The process models in the collection are designed in the most recommended process modeling tool [12], Signavio [34]. The two key reasons for choosing this collection of models are the following: a) the collection contains process models with a diverse label and structural features [13, 11], and from several genres, hence, the results generated using such collections are valid for several domains, and b) a recent study [7] has emphasized that a necessary and sufficient pool of queries and human-generated results against these queries are also available, hence, making it a feasible test-bed for the matching experiments.

More precisely, the collection includes: a) 150 Original process models (O), including the two datasets (University Admissions Processes and Birth Registration Processes) used in the Process Model Matching Contest 2015 [14], and b) three other handcrafted variants of these 150 models, Near Copy (NC), Light Revision (LR), and Heavy Revision (HR).

Note that the variants are generated by employing a systematic and rigorous procedure to impart diversity in labels and structure of models in order to challenge the abilities of the matching techniques [13, 11]. The NC variant of a model is generated by slightly changing the formulation of each label of the model, whereas the LR variant is generated by substantially changing the formulation of each label of the model. The HR variant is generated by significantly changing the formulation of each label.

The smallest model in the collection contains 11 activities and the largest model contains 54 activities. Another unique feature of the collection is that the models included in the collection comply most of the process modeling guidelines, presented in [15]. For instance, there is no process model in the collection that contains a split gateway node, without a corresponding join gateway node. The human effort involved in generating the collection can be understood by the number of operations performed while generating the three variants. That is, 24 092 insertion, deletion, and substitution of words were performed to generate three variants of process models.

For generating the full-length textual descriptions of 669 process models, we have used Natural Language Generation System (NLGS). Figure 3 shows an example textual description generated by the NLGS. As far as we are aware, NLGS is the only established tool that can automatically and comprehensively generate a textual description of process models. Note, an empirical evaluation of the textual descriptions generated by NLGS has established that the NLGS generated textual descriptions are superior to the human-generated textual descriptions, in terms of completeness, structure, and linguistic complexity [16]. Furthermore, a users' evaluation of the NLGS generated textual descriptions demonstrate that the descriptions are understandable, and they effectively allow the reader to interpret the semantics of process models [16]. An example textual description of a healthcare process generated using the NLGS is presented in Figure 3. Accordingly, the full-length descriptions' corpus contains 87 772 words, that include 29 493 (33.6 %) stop words. These numbers indicate that the textual descriptions are not merely a collection of activity labels, rather a significant amount of stop words are used in generating the textual descriptions.

3.2 Generating Summary Descriptions

In this section, we first provide an overview of the four diverse techniques that we have used for generating summary descriptions. Subsequently, in Section 3.2.2, the procedure that we have employed for generating the corpora of summary descriptions is presented.

3.2.1 Summarization Techniques

We have used four diverse text summarization techniques, ranging from a collection of most important words to a state-of-the-art deep learning based technique, for generating summary descriptions. In particular, we have used TF-IDF [19] which is a collection of important words based approach, LexRank [10] that employs a graph-based approach to rank sentences, Lingo [32] which is a state-of-the-art approach to identify the sentences that includes key phrases of the input description, and K-means with skip thoughts embeddings [37], which employs a deep learning based approach for generating summaries. A brief overview of each summarization technique is as follows:

The process begins when the hospital inquiry checks data. Then,

- The hospital inquiry finds the information is missing. Afterwards, the hospital inquiry requests the parents to the complete information. Subsequently, the hospital inquiry conducts the information received. Then, the hospital inquiry informs the civil court.
- The hospital inquiry finds the information complete.

Once was the hospital administration confirms the parents accepted baby or not. Afterwards, is.

- The hospital administration sees the parents don't accept baby. Subsequently, the hospital administration sends the information. Then, the hospital administration forwards the case to the higher authority.
- The hospital administration sees the parents accept baby. Afterwards, the hospital administration checks the parents nationality. Subsequently, the hospital administration checks the parents Russian citizenship or not. Then, is.
  - The hospital administration finds the no one has Russian nationality. Afterwards, the hospital administration conducts the trial in court of nationality affairs. Subsequently, the hospital administration receives the citizenship decision. Then, the hospital administration registers the citizenship of the baby.
  - The hospital administration finds the at least 1 is Russian. Afterwards, the hospital administration registers the baby as Russian.

Once was the hospital administration takes the decided name of the baby. Subsequently, the hospital inquiry creates the birth certificate. Then, is.

- The hospital inquiry registers the baby.
- The hospital inquiry sends the birth information to the parents.

Once was the hospital inquiry finds at the least one of the parents has registration. Afterwards, the hospital inquiry getting the citizenship stamp on the birth certificate. Subsequently, the hospital inquiry sends the request about money benefits. Then, the hospital administration formalizes the moth payments.
Afterwards, the process is finished.

Figure 3. Textual description of order process model generated by NLGS

Term Frequency-Inverse Document Frequency (TF-IDF) relies on the importance of words in a document. That is, firstly, a frequency matrix is generated in which columns represent the vocabulary set of all the textual descriptions, whereas, rows represent the identities of textual descriptions in the collection. Secondly, using the formulas presented in Equation (1), the values of the matrix are populated. In the equation given below, t is a vocabulary term, whereas D_i is the i^{th} textual description in the collection. Finally, TF-IDF scores are used to select top N words for each document for inclusion in summary, where N is the number of words that should be included in the summary. Note that the generated summary using this technique is a mere collection of important words that may not be usable for the comprehension of the workflow of the process.

$$TF - IDF = TF(t) \times IDF(t) \tag{1}$$

where

$$TF(t) = \frac{Freq_t^{Di}}{|t \ \forall t \in Di|}$$

and

$$IDF(t) = \log \frac{|Di|}{|Di, \ such \ that, \ t \in Di|}.$$

LexRank is a sentence ranking based approach that relies on the use of Eigenvector Centrality in a graph to compute the importance of each sentence [10]. In the first step of the technique, the source text is tokenized into sentences and each sentence is represented as a vertex in a graph. In the second step, edges between the vertices are marked on the bases of Inverse Document Frequency (IDF). Note that we have adapted the notion of IDF to Inverse Sentence Frequency (ISF). That is, we take the log of the total number of sentences in the process description and divide it the number of sentences in which the word occurs, as shown in Equation (2). Subsequently, if the generated similarity score between two sentences is above a certain threshold value, a value 1 is stored in the respective index of the sentence matrix and increment 1 is performed to the degree values. Otherwise, no increment is performed to the degree value. Lastly, the final score of each sentence is computed using the power method followed by vertices sort.

$$Similarlity(S_i, S_j) = \frac{\sum_{w \in S_i, S_j} tf_{w,S_i} \times tf_{w,S_j} \times (idf_w)^2}{\alpha \times \beta}, \tag{2}$$

$$\alpha = \sqrt{\sum_{x_k \in S_i} (tf_{x_k,S_i} \times idf_{x_k})^2},$$

$$\beta = \sqrt{\sum_{y_k \in S_j} (tf_{y_k,S_j} \times idf_{y_k})^2},$$

$$p(u) = \frac{d}{N} + (1-d) \times \gamma, \tag{3}$$

$$\gamma = \sum_{v \in adj[u]} \frac{similarity(u,v)}{\sum_{z \in adj[v]} similarity(z,v)} p(v).$$

Lingo is a state-of-the-art technique that employs a clustering-based approach to identify the important sentences that include the key phrases of the given description [32]. In the first step, pre-processing is performed on the input text by removing stop-words and applying stemming. In the second step, the phrases are extracted based on the recurring ordered sequences of terms appearing in the document. Subsequently, a term-document (t-d) matrix is generated for each key phrase in the document. In the third step, the matrix is factorized using Singular Value Decomposition (SVD) to find cluster labels, formally called the topic label of the document. In particular, we have used a publicly available implementation [33]. Finally, we generate a summary by extracting those sentences which contain the most important topics.

K-means clustering is an extraction based approach in which sentences are extracted using k-means clustering technique with skip-thought embeddings. As a starting point, each document is decomposed into its constitute sentences. In the second step, encoder, which is the main part of the skip thought mode, encodes the sentences using Recurrent Neural Network with Gated Recurrent Unit. Meaning that a fixed-length vector representation for each sentence is generated [26]. Equations (4)–(8) describe the sequences of steps which are performed to encode the sentences.

$$r^t = \sigma(W_r x^t + U_r h^{t-1}), \tag{4}$$

$$z^t = \sigma(W_z x^t + U_z h^{t-1}), \tag{5}$$

$$\bar{h}^t = \tanh(Wx^t + U(r^t \bigodot h^{t-1})), \tag{6}$$

$$h^t = (1 - z^t) \bigodot h^{t-1} + z^t \bigodot \bar{h}^t \tag{7}$$

where r^t is the reset gate, $\bigodot$ represents the element-wise multiplication, z^t is the update gate, and $\bar{h}^t$ is the proposed state update at time t. In the third step, the encoded sentences are clustered using K-means clustering techniques as shown in Equation (8).

$$Kmeans = \sum_{j=1}^{k} \sum_{i=1}^{n} ||x_i^j - c_j|| \tag{8}$$

where $||x_i^j - c_j||$ is selected distance measure between a data point x_i^j and cluster c_j. Finally, the sentences corresponding to sentence embeddings closest to the cluster centers are chosen for inclusion in the summary. The implementation of the technique that we have used in this study can be downloaded from [31].

3.2.2 Generating Summary Descriptions

We have generated the summary descriptions of 669 full-length textual descriptions using the four text summarization techniques discussed above at five different compression rates, 10 %, 20 %, 30 %, 40 % and 50 %. The x % compression rate indicates that the top x % important sentences are preserved in the generated summary. In case, the number of sentences to be preserved is in decimal (50 % of 7 is 3.5), the decimal value was truncated. Accordingly, we yielded a total of twenty summary descriptions corpora, each containing summary descriptions of 669 process models. Table 1 shows an example summary generated by each technique, TF-IDF, LexRank, Lingo, and K-means clustering with skip-thought embeddings, at a compression rate of 50 %. It can be observed from the table that the TF-IDF generated summary is a collection of words rather than complete sentences. Therefore it is not usable for compression of the workflow of the process. On the contrary, the summaries generated by the three other techniques are readable as the techniques rely on ranking sentences and subsequently choosing a subset of sentences for generating summaries. In the example, the bold text represents the sentences that are common between all the three sentence-level summaries, whereas the italic text represents the sentences that are common between two sentence-level summaries.

3.2.3 Query Descriptions and Human Annotations

Typically, the existing studies, such as [17], use merely ten randomly selected queries and a source collection of 100 processes, to evaluate the effectiveness of the matching technique. A key limitation of using such a small and randomly selected queries is that the findings generated in these settings may not be reliable. In contrast to those studies, we have chosen a very large number of 56 queries, a collection of 669 process descriptions, and a recently developed human benchmark [7, 13], for our experimentation.

As discussed in Section 3.1, a key motivation for the choice of the source collection is that it includes a large and carefully handcrafted collection of over 600 process models, which includes three variants of each process model, NC, LR, and HR. A key reason for the choice of 56 queries over a randomly collected pool of merely ten queries is that our queries are selected by employing a systematic and rigorous procedure, without having a pre-defined number in mind. Furthermore, the principal purpose of the procedure was to ensure the inclusion of a necessary and sufficient set of query processes. Essentially, the procedure is composed of four main steps. In the first step, the values of the widely use structural features of each process model were computed. These structural features are size, diameter, sequentiality,

TF-IDF	[the, hospital, administration, inquiry, then, is, parents, afterwards, subsequently, finds, baby, information, citizenship, of, checks, to, was, sends, registers, birth, process, information, conducts , baby , or , not, sees, accept, nationality, russian, one, has, russian, once, certificate, begins, when, a, data, missing, requests, complete]
LexRank	**The process begins when the hospital inquiry checks a data.** *Subsequently, the hospital inquiry conducts the information received.* **Then, the hospital inquiry informs the civil court. Once was the hospital administration confirms the parents accepted baby or not. The hospital administration sees the parents don't accept baby.** Then, the hospital administration forwards the case to the higher authority. The hospital administration finds the no one has Russian nationality. *Afterwards, the hospital administration conducts the trial in court of nationality affairs.* **Subsequently, the hospital administration receives the citizenship decision.** The hospital administration finds the at least 1 is Russian. *Once was the hospital administration takes the decided name of the baby.* Subsequently, the hospital inquiry creates the birth certificate. **Once was the hospital inquiry finds at the least one of the parents has registration. Afterwards, the hospital inquiry getting the citizenship stamp on the birth certificate. Subsequently, the hospital inquiry sends the request about money benefits.** Then, the hospital administration formalizes the moth payments.
Lingo	**The process begins when the hospital inquiry checks a data.** Then is . The hospital inquiry finds the information is missing. *Subsequently the hospital inquiry conducts the information received.* **Then the hospital inquiry informs the civil court. Once was the hospital administration confirms the parents accepted baby or not.** Afterwards is . **The hospital administration sees the parents don't accept baby.** The hospital administration sees the parents accept baby. *Afterwards the hospital administration conducts the trial in court of nationality affairs.* **Subsequently the hospital administration receives the citizenship decision.** *Then the hospital administration registers the citizenship of the baby.* **Once was the hospital inquiry finds at the least one of the parents has registration. Afterwards the hospital inquiry gettings the citizenship stamp on the birth certificate. Subsequently the hospital inquiry sends the request about money benefits.** *Afterwards the process is finished.*
RNN	**The process begins when the hospital inquiry checks a data.** Afterwards, the hospital inquiry requests the parents to the complete information. **Then, the hospital inquiry informs the civil court. Once was the hospital administration confirms the parents accepted baby or not. The hospital administration sees the parents don't accept baby.** Then, is . Afterwards, the hospital administration checks the parents nationality.The hospital inquiry finds the information complete. **Subsequently, the hospital administration receives the citizenship decision.** The hospital administration finds the no one has russian nationality. *Then, the hospital administration registers the citizenship of the baby. Once was the hospital administration takes the decided name of the baby.* **Afterwards, the hospital inquiry gettings the citizenship stamp on the birth certificate. Once was the hospital inquiry finds at the least one of the parents has registration. Subsequently, the hospital inquiry sends the request about money benefits.** *Afterwards, the process is finished.*

Table 1. Summaries generated by TF-IDF, LexRank, Lingo and RNN at compression rate 50 %

network connectivity, token split, etc. [20]. Further details of these metrics can be found in [20]. In the second step, a correlation was computed between every pair of structural metrics. Subsequently, for each pair of metrics having a very high co-relation of over 0.9 one metric was excluded. Hence, ensuring that the values of only adequate metrics are taken into consideration. Thirdly, to ensure the diversity, the process models with the highest, lowest, and average values of each metric were selected as query models. Finally, the steps were repeated for each variant in the collection, NC, LR, and HR, while avoiding redundancy. Accordingly, the generated collection includes 14 process models of each type, Original, NC, LR, and HR, as well as the process model with diverse structural properties.

The full-length textual descriptions of the selected 56 query models are used as an input to the four summarization techniques to generate the summaries of the query processes at different compression rates (10 %, 20 %, 30 %, 40 %, and 50 %). As a result, the summary descriptions of 1 120 queries were generated. These summary descriptions have been used as queries in the rest of the paper for experimentation.

4 MATCHING TECHNIQUES

A notable study [18] has classified text-matching approaches into seven broad categories: overlapping of grams, lexical similarity, string and sequence comparison, fingerprinting, probabilistic methods, NLP methods, and structural methods. The approaches in the former three categories primarily rely on the actual content of the query-source descriptions, whereas, the latter four rely on the use of structural or textual features of the query-source descriptions. In this study, we limit our choice of matching techniques to the former three categories of techniques due to two reasons:

1. summarization may have substantially changed the structure or textual features of the descriptions, which may ultimately affect the matching performance, and
2. the latter four categories of approaches increase the computational overhead of computing structural or textual features of the query-source descriptions.

Below, we provide an overview of the matching techniques used in this study. In particular, we present one technique from the first and second categories (N-gram overlap and Vector Space Model, respectively) and three techniques (Longest Common Subsequence, Local Alignment, and Global Alignment) from the third category.

4.1 N-Gram Overlap

N-gram overlap computes the similarity between a query-source descriptions pair by dividing them into a set of tokens, called grams [21]. It then counts the number of common tokens in the two descriptions and divides it by the number of tokens in one or both descriptions, to get a normalized score between 0 and 1. The value of n determines the number of words in each token. Formally, it is defined as follows:

$$S(Q,S) = \frac{|T(Q) \cap T(S)|}{\min(|T(Q)|, |T(S)|)} \tag{9}$$

where $T(Q)$ and $T(S)$ is the number of token in query and source description, respectively.

4.2 Vector Space Model (VSM)

VSM computes the similarity between a query-source descriptions pair by first representing each description in a vector space, where each word in the description represents a dimension in a vector space [22]. The similarity is then measured by computing angle between them. Formally, the normalized score is computed as follows:

$$S(Q,S) = \frac{\sum_{i=1}^{n} Q_i \times S_i}{\sqrt{\sum_{i=1}^{n}(Q_i)^2 \times \sum_{i=1}^{n}(S_i)^2}}. \tag{10}$$

4.3 Longest Common Subsequence (LCS)

Longest Common Subsequence computes the similarity between a query-source descriptions pair by identifying the longest consecutive sequence of tokens that are common between the two descriptions and dividing it with the length of the smaller description, to compute a normalized similarity score [18]. Formally, it is defined as follows:

$$S(Q,S) = \frac{|LCS|}{\min(|Q|,|S|)} \tag{11}$$

where LCS is the longest sequence of tokens that are common between the two descriptions.

4.4 Local Alignment (LA)

Local Alignment computes the similarity between a query-source descriptions pair by identifying the identical portion of tokens (small regions) between the two sequences [24]. In particular, for each matching pair of tokens the matching score is incremented by 1, and for each mismatched pair of tokens the matching score is decremented by 1. Subsequently, the normalized score is computed by dividing the similarity score with the minimum length of the query-source description

$$S(Q,S) = \frac{L_{score}}{\min(|Q|,|S|)}. \tag{12}$$

4.5 Global Alignment (GA)

Global Alignment computes the similarity between a query-source descriptions pair by representing both descriptions as a sequence of words and then identifying the identical text between the entire length of the two descriptions [25]. Generally, the technique is recommended for a sequence of equal and near-equal lengths. For each matching pair of tokens, the matching score is incremented by 1, and for each mismatch, the score is decremented by 1. The normalized score is then computed by the following equation:

$$S(Q,S) = \frac{G_{score}}{\min(|Q|,|S|)}. \tag{13}$$

5 COMPARISON OF SUMMARY DESCRIPTIONS

In this section, we have computed the similarity between summary descriptions of queries generated by the four summarization techniques to evaluate how similar or dissimilar are the summary descriptions. The process that we have employed for the comparison of summary descriptions is presented in Figure 4. In particular, we have generated 20 corpora, each containing summary textual descriptions of 56 query processes, i.e., a corpus of summary descriptions generated by each summarization technique at each compression rate 10 %, 20 %, 30 %, 40 %, and 50 %. The comparison of these $56 \times 5 \times 4 = 1\,120$ summary descriptions would require creating numerous pairs of summary descriptions.

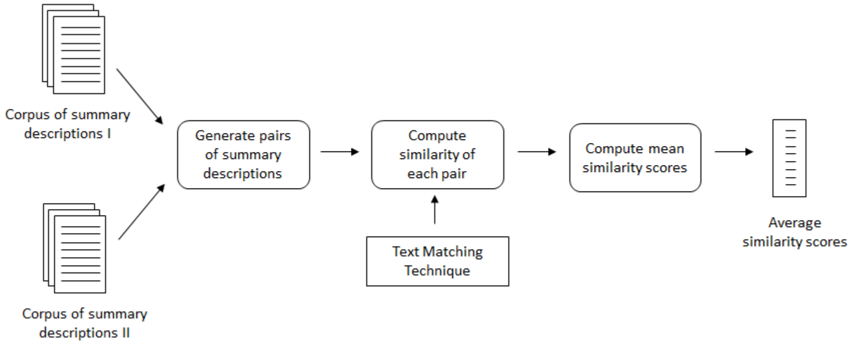

Corpus of summary descriptions I
Corpus of summary descriptions II
Generate pairs of summary descriptions
Compute similarity of each pair
Text Matching Technique
Compute mean similarity scores
Average similarity scores

Figure 4. Comparison of textual process descriptions

The manual comparison of these many pairs is a tedious task which requires a substantial human effort. Therefore, we have used two similarity estimation techniques (n-gram overlap and Longest Common Subsequence) to compute the similarities between these pairs. N-gram computes the degree of similarity between a query-source pair by counting the number of unique tokens (common words) and dividing it by the length of the short description to get a normalized score. The similarity score thus represents the content overlap between the query-source pair without taking into consideration the ordering of the words. Due to that limitation, we have also used a variant of LCS – an order-preserving similarity estimation method. LCSnorm, a variant of LCS, computes the similarity by counting the number of edit operations required to transform one text into another and dividing it with the length of the short text.

Table 2 shows the average similarity scores of all possible combinations of pairs of summary descriptions. In the table, the average similarity score of 0.62 at a compression rate 50 % for the 1-gram technique represents that 62 % of the unique words (vocabulary) used by these two algorithms overlap. The key observations from the results are as follows:

Techn.	Comp.	TF-IDF &			LexRank &		Lingo &
		LexRank	Lingo	K-means	Lingo	K-means	K-means
Unigram	50 %	0.62	0.64	0.60	0.86	0.86	0.84
	40 %	0.46	0.50	0.47	0.78	0.77	0.78
	30 %	0.31	0.37	0.38	0.67	0.66	0.72
	20 %	0.23	0.29	0.33	0.58	0.58	0.68
	10 %	0.16	0.20	0.26	0.44	0.43	0.57
LCS	50 %	0.39	0.37	0.34	0.60	0.63	0.60
	40 %	0.35	0.36	0.32	0.54	0.58	0.56
	30 %	0.29	0.34	0.34	0.48	0.52	0.52
	20 %	0.28	0.33	0.39	0.44	0.48	0.49
	10 %	0.26	0.36	0.52	0.36	0.35	0.48

Table 2. Average similarities scores between summaries of 56 query descriptions

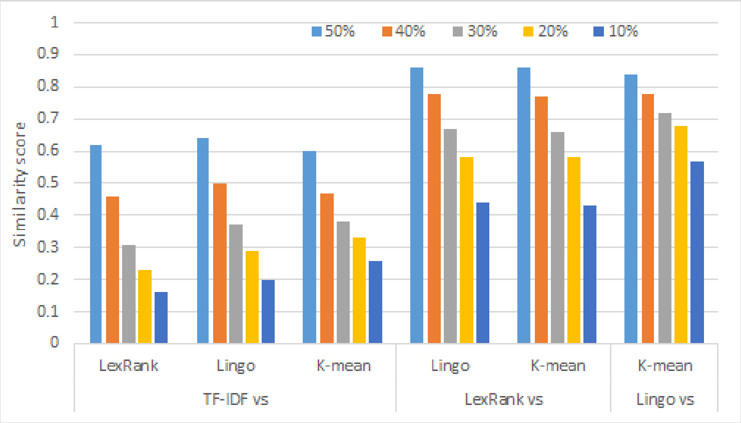


Figure 5. Unigram based comparison of all pairs

Vocabulary overlap. It can be observed from Table 2 that the unigram similarity score for a large majority of the cases, 53 out of 60, are less than or equal to 0.67. These lower values indicate that at least one-third of the vocabulary between these pairs is different. For the remaining seven cases, which are highlighted with gray color, the unigram score is substantially high, i.e., 0.86, 0.86, 0.84, 0.78, 0.77, 0.78, and 0.72. However, it can be observed from the table that the LCS scores of the pairs, where the vocabulary overlap, are also higher, i.e., 0.60, 0.63, and 0.60, 0.54, 0.58, 0.56, and 0.52. These lower values represent that the ordering of the words in these summaries is significantly different from each other, hence, indicating a significant difference between the summaries.

Similarity between types of pairs. Figure 5 plots the n-gram similarity scores between all the pairs of summary descriptions. From the figure, it can be observed that the similarity scores between TF-IDF generated summaries and the ones generated by the remaining techniques are substantially low. On the contrary, the corresponding similarity scores between the other pairs are on the higher side. That is, the similarity score of LexRank & Lingo, LexRank & K-means, and Lingo & K-means are higher than that of TF-IDF & LexRank, TF-IDF & Lingo, and TF-IDF & K-means. A key reason for the differences in the similarity scores stems from the fact that TF-IDF employs an entirely different mechanism from the other three techniques for generating summaries. That is, TF-IDF employs a word-based approach to rank and identify important words for a summary, whereas, the other three techniques employ a sentence-ranking approach to identify a subset of sentences for inclusion in the summary.

Impact of compression rate on the similarity. It can be observed from Figure 5 that as the compression rate decreases from 50 % to 10 %, the vocabulary overlap between the summaries decreases gradually. These decreasing numbers represent that the differences between the summaries in the pair widen with the decrease in the compression rate. Hence, indicating that all the techniques employ a different mechanism to rank words or sentences which becomes more visible when a smaller number of words or sentences are chosen for generating a summary.

From the above discussion, we conclude that the summary textual descriptions generated by the four techniques, TF-IDF, LexRank, Lingo, and K-means, are significantly different from each other. Hence, the choice of the summarization technique is a non-trivial task. This raises several questions, such as, what is the impact of different summarization techniques on process matching? Which summarization technique generates the most appropriate summary for effective, as well as efficient process matching results? What level of compression rate is most appropriate for effective, as well as efficient process matching? To answer these questions, in the remaining part of the paper, we have performed several process matching experiments.

6 PROCESS MATCHING EXPERIMENT SETUP

This section presents the details of the experimental setup that we have used for analyzing the performance trade-off between summary and full-length textual descriptions for process matching. An overview of the experimental setup is presented in Figure 6.

6.1 Dataset and Evaluation Measures

For the experiments, the full-length descriptions of 669 process models have been used as a source collection, and two sets of queries have been used for the process matching:

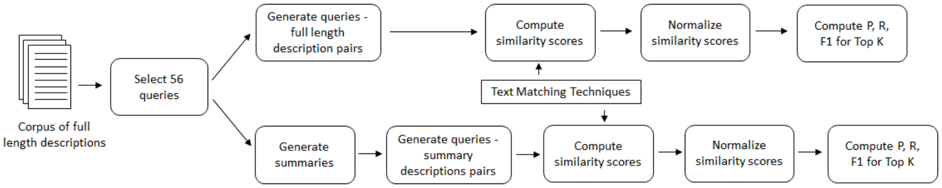

Corpus of full length descriptions
Select 56 queries
Generate queries - full length description pairs
Compute similarity scores
Normalize similarity scores
Compute P, R, F1 for Top K
Text Matching Techniques
Generate summaries
Generate queries - summary descriptions pairs
Compute similarity scores
Normalize similarity scores
Compute P, R, F1 for Top K

Figure 6. Experimental setup

1. full textual descriptions of 56 query models,
2. four sets of summary textual descriptions of 56 query models at five different compression rates 10 %, 20 %, 30 %, 40 % and 50 %.

The details of the summary descriptions and the human annotations have been discussed in Section 3, whereas the trade-offs between the use of summary and full-length textual descriptions have been analyzed in terms of *effectiveness* and *efficiency*.

For the *effectiveness* of matching, we have used three established measures, Precision (P), Recall (R) and F1 score. Precision represents the percentage of source process models that are retrieved and are relevant. Recall represents the percentage of source process models that are relevant and retrieved. F1 score is a harmonic mean of Precision and Recall.

For the *efficiency*, we have used Retrieval Time (RT) as a measure. For a full-length query description, RT is the time taken by a technique to match the full-length query description with all the source descriptions in the collection. Whereas, for a summary description query, RT is the sum of the time spent to generate a summary description of the query process and the time taken by a technique to match the summary query descriptions with all source descriptions in the collection.

6.2 Evaluation Methodology

We have implemented all the four summarization techniques, TF-IDF, LexRank, Lingo, and K-means, as described in Section 3.2.1. Each implementation takes a full-length textual description as input and generates its summary descriptions at five compression rates 10 %, 20 %, 30 %, 40 %, and 50 %. For measuring the effectiveness, the matching techniques presented in Section 4 are used for experiments. Each technique has been implemented as a program, where each implementation takes a query process description as input and generates its pairs with 669 source process descriptions, formally called query-source pairs. Subsequently, each implementation computes a similarity score between 669 query-source pairs and saves them in a text file in descending order of the similarity score, meaning that the most relevant processes are on the top. Furthermore, top K processes have been generated by varying the value of K between 4 and 16, with a step size of four. The reason for the varying value of K lies in the nature of the source process collection, i.e., the source collection contains four variants of each model, 150 original process specifications, and three handcrafted variants of each model. Consequently, keeping the step size as 4 has helped us evaluate whether or not all the variants are ranked in the top slots. Finally, Precision, Recall, and F1 scores have been computed after applying pre-processing in order to compare the effect of each pre-processing step. In particular, query and source descriptions have been pre-processed by removing stop words, stemming (using Snowball stemmer), and a combination of both. The process was repeated for full-length query descriptions, as well as for the summary query descriptions generated by TF-IDF, LexRank, Lingo, and K-means, at five different compression rates.

For measuring the efficiency, the implementations of the summarization techniques were modified to include the computation of the summary generation time. Similarly, the implementations of the matching techniques were modified to compute the retrieval time, as defined in Section 6. These implementations take a query-source pair as input and compute the retrieval time of each pair. Subsequently, the retrieval time was saved in a text file. Note that the efficiency experiments have been performed 10 times for 56 queries and at each compression rate. The results presented in this paper are the average of the 10 iterations. Similarly, the summary generation time used in this paper is the average of the 10 iterations.

Technique	**Full Desc.**			**TF-IDF**			**LexRank**		
	P	**R**	**F1**	**P**	**R**	**F1**	**P**	**R**	**F1**
1-Gram	0.719	0.455	0.557	0.741	0.471	0.576	0.705	0.444	0.545
2-Gram	0.723	0.454	0.558	0.411	0.253	0.313	0.696	0.441	0.540
3-Gram	0.688	0.438	0.535	0.058	0.031	0.040	0.647	0.420	0.535
GA	0.656	0.426	0.517	0.754	0.475	0.583	0.714	0.449	0.517
LCS	0.634	0.428	0.511	0.737	0.462	0.568	0.674	0.429	0.511
LA	0.616	0.388	0.476	0.250	0.152	0.189	0.580	0.366	0.476
VSM	**0.772**	**0.487**	**0.597**	**0.763**	**0.480**	**0.589**	**0.786**	**0.502**	**0.597**

Table 3. Effectiveness comparison of full and summary descriptions (top 4)

7 RESULTS AND ANALYSIS

7.1 Effectiveness Results

Precision, Recall, and F1 scores provide three different types of measures to gauge the effectiveness of a matching technique. Therefore, we have included the results of all the three measures in Table 3 and Table 4, for all the matching techniques,

Tech.	**Full Desc.**			**Lingo**			**RNN**		
	P	**R**	**F1**	**P**	**R**	**F1**	**P**	**R**	**F1**
1-Gram	0.719	0.455	0.557	0.688	0.439	0.536	0.670	0.416	0.513
2-Gram	0.723	0.454	0.558	0.705	0.448	0.548	0.683	0.430	0.528
3-Gram	0.688	0.438	0.535	0.661	0.426	0.518	0.625	0.410	0.495
GA	0.656	0.426	0.517	0.705	0.450	0.549	0.674	0.420	0.518
LCS	0.634	0.428	0.511	0.728	0.465	0.568	0.603	0.385	0.470
LA	0.616	0.388	0.476	0.629	0.405	0.493	0.554	0.341	0.422
VSM	**0.772**	**0.487**	**0.597**	**0.790**	**0.500**	**0.612**	**0.795**	**0.504**	**0.617**

Table 4. Effectiveness comparison of full and summary descriptions (top 4)

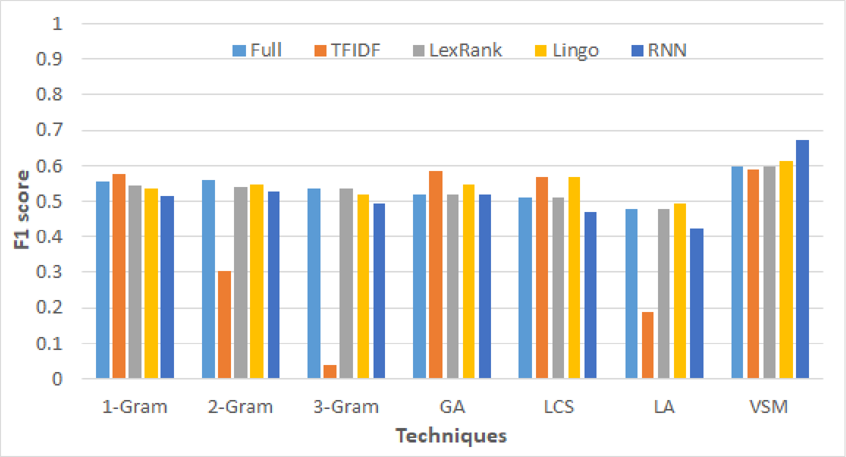


Figure 7. Performing variation across summarization techniques

where full-length textual descriptions and the summary descriptions are used as queries. Note that each value in the table is an average score of the 56 query descriptions. It can be observed from the Table 3 that the Precision scores of full-length descriptions are significantly higher than the Recall scores for all the matching techniques. Furthermore, a similar trend can be observed from Table 3 and Table 4, for each summarization technique. A higher value of Precision represents that among the processes retrieved by a technique the majority of the processes were relevant, whereas, a lower Recall score represents that the majority of the variants were not retrieved. Our synthesis of the Recall results revealed that for each query the identical processes were retrieved, whereas, for a majority of the queries the NC variants were also retrieved. Furthermore, the LR variants of some queries were retrieved, whereas, HR variants of a few queries were retrieved. The key observations based on the analysis of results are as follows:

Most suitable matching technique. It can be observed from the table that the Vector Space Model outperformed all the matching techniques for all types of summary descriptions, as well as for the full-length descriptions. That is, the Precision, Recall, as well as the F1 scores of the VSM are higher than all the other techniques, and for both types of descriptions, summary and full-length descriptions. These highest values are highlighted using gray background color in the table. A possible explanation to the higher values of the evaluation measures stems from the fact that the VSM firstly represents both query and source documents as vectors in a high-dimensional space, and subsequently the similarity is computed by the degree of angle between query-source vectors, rather than merely the overlap in the contents or the alignment of the text.

Variation across summarization techniques. A comparison of the F1 scores of the summary and full-length descriptions is presented in Figure 7. It can be observed from the figure that there is no single summarization technique that outperforms all the other summarization techniques for all the matching techniques. Furthermore, the summary generated by the state-of-the-art deep learning based summarization technique achieved the highest F1 score of 0.617 using the VSM matching technique, whereas Lingo achieved a comparable F1 score of 0.612. These two observations indicate that the most appropriate combination of a summarization and matching technique is the VSM and K-means clustering with skip-thought embeddings, which is a deep learning based approach.

Tech.	**Full Desc.**	**LexRank**	**TF-IDF**	**Lingo**	**RNN**
1-Gram	559	465	359	380	972
2-Gram	558	523	402	433	1 030
3-Gram	561	477	397	414	1 038
GA	2 763	1 180	468	918	1 491
LCS	855	415	241	346	922
LA	6 871	2 955	928	1 116	1 478
VSM	**21**	**20**	**28**	**17**	**603**

Table 5. Retrieval time (in milliseconds) comparison of full and summary descriptions

7.2 Synthesis of Effectiveness

For a thorough analysis of the results, we have synthesized the effectiveness score of 56 queries by dividing them into four types, such that each type has an equal number of queries. The query-types are Original Queries (OQ), Near Copy Queries (NCQ), Light Revision Queries (LRQ), and Heavy Revision Queries (HRQ). For each type of queries, all the experiments have been repeated, and the Precision, Recall, and F1 scores have been recorded in Table 6. The key observations elicited from these results are as follows.

It can be observed from the shaded elements in Table 6 that the Precision, Recall, and F1 score achieved by VSM for *each* query type is higher than the corresponding scores achieved by any other matching technique when RNN generated summary descriptions were used for the process matching. These results are consistent with the results of VSM presented in Table 4, where VSM outperformed all the other techniques based on the average F1 score of 56 queries. Hence, reinforcing that VSM is the most effective matching technique.

It can also be observed from the bold values in Table 6 that the average F1 scores achieved by HRQs are always significantly less than the F1 scores achieved by OQs, NCQs, and LRQs, for all the matching techniques. This observation is valid for the full-length descriptions, as well as for the RNN generated summary descriptions. This lower value is due to the significant differences in the specifications of HR variants from O, NC, and LR variants of processes. Hence, indicating that the identification of HR variant is a challenging task for the matching techniques. It is thus desirable to invent new matching techniques that can effectively retrieve heavily modified variants of processes.

Another notable observation is that for the all types of queries, the F1 score achieved using the RNN generated summary descriptions is either higher than or comparable with the F1 scores achieved by the full-length generated descriptions. This observation is valid across all the matching techniques. In particular, the average differences in F1 scores between the matching techniques are 0.028, 0.062, 0.058, and 0.033 for OQs, NCQs, LRQs, and HRQs, respectively. These results indicate that the RNN generated summary descriptions are equally effective for all query-types and the matching techniques.

7.3 Efficiency Analysis

Table 5 shows a comparison of the Retrieval Time (RT) for the matching techniques where the full-length and summary generated textual descriptions are used as queries. Recall from Section 6 that, for a full-length description query, RT is the time taken by a technique to match the query with all the source descriptions in the collection. On the contrary, for a summary description query, RT is the sum of the time consumed in generating a summary description and the time taken by a technique to match the summarized query description with all the source descriptions in the collection. Each value in Table 5 represents the average RT of 56 queries for 10 iterations. The key observations about efficiency analysis are as follows:

Efficiency of the matching techniques. VSM is the most efficient technique for process matching, as its retrieval time is merely 21 milliseconds in case of full-length query descriptions. Furthermore, it can be observed from the table that the use of summary descriptions substantially reduces the RT of all the matching techniques, with the exception of VSM. That is, the RT of VSM does not decrease substantially. Further analysis of the RT of VSM revealed that the RT for full description queries is minuscule, 21 milliseconds only. Hence, the

		Full-length description			**RNN summary**		
Queries	**Technique**	**P**	**R**	**F1**	**P**	**R**	**F1**
OQ-14	Unigram	0.804	0.441	0.570	0.804	0.438	0.567
	Bigram	0.786	0.439	0.563	0.786	0.428	0.554
	Trigram	0.786	0.439	0.563	0.732	0.422	0.535
	GA	0.804	0.454	0.580	0.821	0.464	0.593
	LCS	0.786	0.462	0.582	0.768	0.443	0.562
	LA	0.768	0.418	0.541	0.696	0.377	0.489
	VSM	0.821	0.444	0.576	0.893	0.498	0.639
NCQ-14	Unigram	0.714	0.537	0.613	0.661	0.482	0.557
	Bigram	0.696	0.519	0.595	0.679	0.513	0.584
	Trigram	0.714	0.537	0.613	0.714	0.536	0.612
	GA	0.696	0.527	0.600	0.732	0.542	0.623
	LCS	0.732	0.575	0.644	0.661	0.506	0.573
	LA	0.714	0.525	0.605	0.571	0.419	0.483
	VSM	0.804	0.614	0.696	0.786	0.596	0.678
LRQ-14	Unigram	0.696	0.489	0.574	0.625	0.430	0.509
	Bigram	0.714	0.491	0.582	0.625	0.432	0.511
	Trigram	0.607	0.428	0.502	0.500	0.372	0.427
	GA	0.643	0.456	0.534	0.589	0.373	0.457
	LCS	0.589	0.420	0.490	0.518	0.332	0.405
	LA	0.482	0.338	0.397	0.464	0.303	0.367
	VSM	0.696	0.469	0.560	0.714	0.496	0.585
HRQ-14	Unigram	0.661	0.352	**0.459**	0.589	0.315	**0.410**
	Bigram	0.696	0.368	**0.481**	0.643	0.346	**0.450**
	Trigram	0.643	0.349	**0.452**	0.554	0.309	**0.397**
	GA	0.482	0.269	**0.345**	0.554	0.300	**0.389**
	LCS	0.429	0.256	**0.321**	0.464	0.258	**0.332**
	LA	0.500	0.270	**0.351**	0.482	0.263	**0.340**
	VSM	0.768	0.420	**0.543**	0.786	0.426	**0.553**

Table 6. Effectiveness comparison of query types (Top 4)

overhead of summary generation time becomes dominant. That is, the summary generation time of LexRank is 3 milliseconds, whereas its matching time is 17 milliseconds, which represents a slight decrease in the matching time. Similarly, for the other techniques, TF-IDF, Lingo, and K-means, the time required for generating summaries is also minute, 0.015, 0.030, and 584 milliseconds, respectively.

Efficiency of the summarization techniques. It can be observed from the Figure 8 that the RT of a large majority of the summarization techniques is significantly less than that of the full-length query descriptions. A notable observation is that, in contrast to other summarization techniques, the retrieval time of K-means generated summaries is significantly higher than that of the full-length descriptions. It is due to the reason that K-means employs a deep learning technique that takes into consideration several parameters for generating summaries, hence, making it a resource-intensive task. Furthermore, it can be observed from Figure 8 f) that the RT of K-means generated summaries is manifolds higher than that of the full-length generated descriptions. Our synthesis of the RT revealed that the retrieval time of summary descriptions actually reduces from 21 to 19 milliseconds. However, due to the higher amount of summary generation time, i.e. 584 milliseconds, the overall retrieval time inflates significantly.

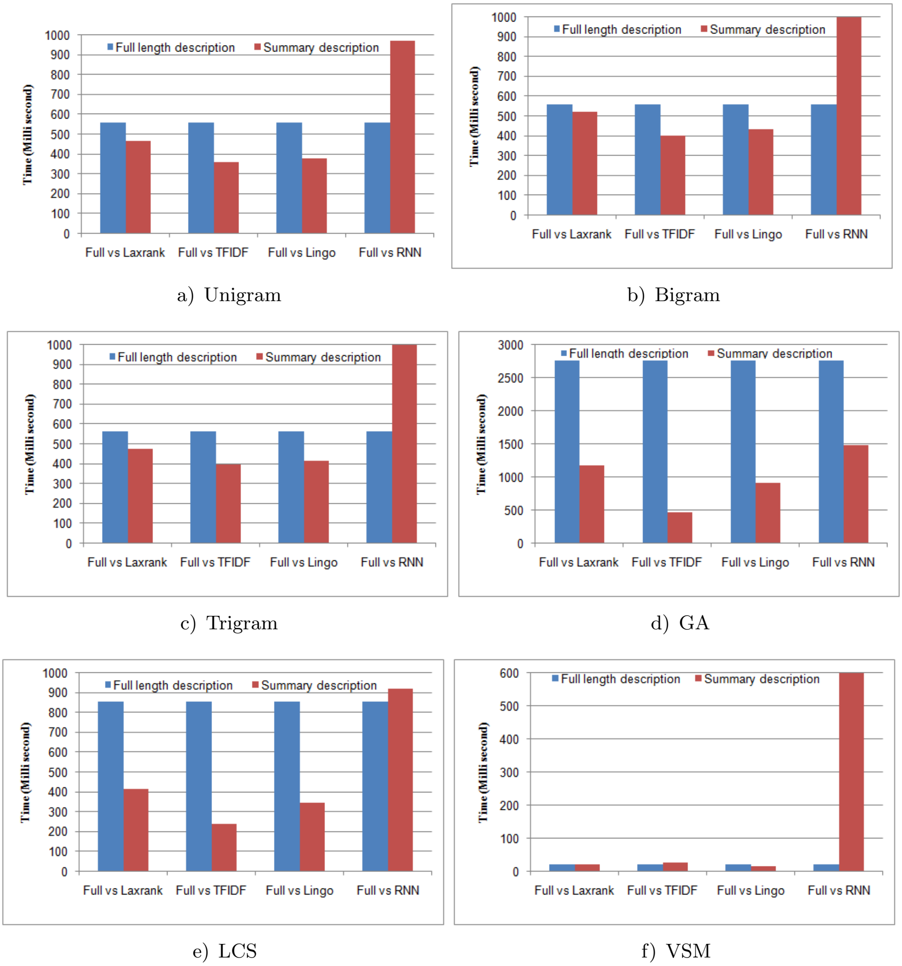
From the discussion we conclude that the choice of summarization technique, as well as the matching technique, significantly contributes to the efficiency of matching. However, one must take into consideration the trade-off between effectiveness and efficiency. Therefore, the subsequent section focuses on analyzing this trade-off in detail.

	Effectiveness				**Efficiency**			
Tech.	TF-IDF	LexRank	Lingo	K-means	TF-IDF	LexRank	Lingo	K-means
1-Gram	+	–	–	–	+	+	+	–
2-Gram	–	–	–	–	+	+	+	–
3-Gram	–	NA	–	–	+	+	+	–
GA	+	NA	+	+	+	+	+	+
LCS	+	NA	+	–	+	+	+	–
LA	–	NA	+	–	+	+	+	+
VSM	–	NA	+	+	–	+	+	–

Table 7. Trade-off between Efficiency (EF) and Effectiveness (EC)

7.4 Efficiency-Effectiveness Trade-Off Analysis

In this section, we discuss the performance trade-off between the summary and full-length textual descriptions. Table 7 provides an overview of the trade-off between these descriptions in terms of efficiency and effectiveness. In the table, a '+' sign for *effectiveness* represents that the use of summary description has a positive impact on the effectiveness of matching, i.e., the average F1 score achieved by the summary queries is higher than the full-length description. Similarly, a '−' sign represents that the use of summaries impedes the effectiveness of matching. On the contrary, a '+' sign for *efficiency* represents that the performance of matching, in terms of efficiency, increases when summary descriptions are used for process matching. That is, the average RT of matching of summary query descriptions is less than the full-length query descriptions, whereas, the '−' sign represents that the matching time of summary descriptions is higher than the full-length descriptions.



a) Unigram

b) Bigram

c) Trigram

d) GA

e) LCS

f) VSM

Figure 8. Performance comparison of summary and full-length description

It can be observed from Table 7 that the efficiency of the matching improves for a large majority of the cases when summary query descriptions are used for process matching. In particular, the efficiency increases when the LexRank generated summary queries are used for matching. However, the use of the LexRank generated summary queries does not increase the effectiveness of matching.

It can also be observed from the table that the efficiency, as well as the effectiveness of matching, does not increase for a large majority of the matching techniques when K-means generated summary descriptions are used as queries. In contrast to that, the use of the summary description queries generated by TF-IDF and Lingo increases the efficiency as well as effectiveness for multiple matching techniques.

Among these two summarization techniques, the effectiveness scores of the Lingo generated summary is higher than that of TF-IDF for a majority of the matching techniques making it more suitable for process matching.

Furthermore, for the most effective matching technique, VSM, the F1 score of 0.612 achieved by Lingo generated summaries is comparable with the F1 score of 0.617 achieved by K-means generated summaries, whereas, the retrieval time of Lingo generated summaries is merely 17 milliseconds, which is less than that of TF-IDF, 28 milliseconds. Hence, we conclude that the summary textual descriptions generated by Lingo at a compression rate of 50 % can achieve a comparable or higher efficiency as well as effectiveness than the summary descriptions generated by the other three techniques, as well as by using full-length textual descriptions.

8 RELATED WORK

The approaches to process matching can be classified into two broad categories:

1. effectiveness enhancement approaches, and
2. efficiency enhancement approaches.

Effectiveness enhancement approaches: This category includes the approaches that aim to enhance the effectiveness of process matching. Several approaches, such as Ref. [17, 27], have proposed to combine structural and behavioral features with label features to decide a query-source pair as equivalent or not equivalent. To compute the similarity between label features, a large majority of the techniques employ syntactic measures, such as distance-based measures [28], to simply count the number of edit operations required to convert one label into another. More advanced techniques, to compute the similarity between label features, use semantic and contextual measures [29]. These measures rely on a lexical database, WordNet [30], to compute semantic similarity between labels. The similarity between label features is combined with graph matching techniques to compute the similarity between a query-source process model pair. Behavioral feature-based approaches [35] compute the similarities between a pair of process models using their execution behaviors, formally called the causal relationship between activities.

Recent studies, such as [36], enhance the accuracy of process matching by integrating the specification of a process model with the textual descriptions of its elements. Another study [7] has proposed the use of textual descriptions as an alternative to the process models.

Efficiency enhancement approaches: This category includes the approaches that aim to enhance the efficiency of the process matching. To the best of our knowledge, this category includes only two approaches, Ref. [38] and [39]. The first approach [38] aims to extract features from process models, and subsequently uses these features to categorize processes as relevant, irrelevant, or potentially relevant. Whereas, the second approach [39] proposes to use a novel feature of the process models, called Feature-Net (FNet). This approach consists of two phases: indexing and querying. In the first phase, each process graph $\{G1, G2, \ldots\}$ in the collection of process models, is indexed. Subsequently, each indexed process graph is split into basic features $\{PF1, PF2, \ldots\}$ to construct an FNet, which is used for computing similarity between a query-source pair.

9 CONCLUSION

In this paper, we argue that the use of full-length text descriptions may impede the efficiency of matching techniques, particularly when the textual descriptions are very long. To mitigate this, we promote the use of summary textual descriptions as an alternative to the full-length textual descriptions. To this end, we have thoroughly investigated the trade-off between efficiency and effectiveness between full-length textual descriptions and our proposed alternative of summary textual descriptions.

We have generated a corpus of full-length textual descriptions of 669 process models and use them to generate 20 corpora of summary descriptions. The full-length textual descriptions corpus is generated from the process models in JSON format using an established tool for generating textual descriptions, called NLGS. Whereas, the 20 summary corpora are generated by using diverse text summarization techniques, at five different compression rates, 10 %, 20 %, 30 %, 40 %, and 50 %. The techniques include a word-based summarization technique, TF-IDF, an established graph-based summarization technique, LexRank, a state-of-the-art clustering technique, Lingo, and another state-of-the-art deep learning based technique, K-means clustering with skip-thought embeddings. To establish that the generated summary corpora are substantially different from each other, we have used two text-matching techniques, N-gram overlap, and LCS. For that, we have first generated 1 120 pairs of summary descriptions and subsequently used the two text-matching techniques to compute the similarity between each pair. The results show that the summaries generated by the two summarization techniques are significantly different from one another, hence, the choice of summarization technique is a non-trivial task. Therefore, we conducted process matching experiments to compare the performance of the summary descriptions generated by the four summarization techniques.

The process matching experiments are performed using 56 full-length textual descriptions as queries and 669 full-length textual descriptions as a source. For matching, we have used seven text-matching techniques: Unigram, Bigram, Trigram, Global Alignment, Longest Common Subsequence, Local Alignment, and Vector Space Model. Furthermore, we have performed experiments using 20 sets of 56 summarized query descriptions generated by the four summarization techniques at five compression rates. Our results show that the use of summary description queries, generated by Lingo at a compression rate of 50 %, can achieve a comparable or higher efficiency as well as effectiveness than the full-length descriptions. In the future, we aim to use other summarization techniques and study their impact on process matching.

REFERENCES

[1] Aysolmaz, B.—Leopold, H.—Reijers, H. A.—Demirörs, O.: A Semi-Automated Approach for Generating Natural Language Requirements Documents Based on Business Process Models. Information and Software Technology, Vol. 93, 2018, pp. 14–29, doi: 10.1016/j.infsof.2017.08.009.

[2] Dumas, M.—La Rosa, M.—Mendling, J.—Reijers, H.: Fundamentals of Business Process Management. Second Edition. Springer, 2018.

[3] Chakraborty, S.—Sarker, S.—Sarker, S.: An Exploration into the Process of Requirements Elicitation: A Grounded Approach. Journal of the Association for Information Systems, Vol. 11, 2010, No. 4, Art. No. 1, doi: 10.17705/1jais.00225.

[4] Sànchez-Ferreres, J.—van der Aa, H.—Carmona, J.—Padró, L.: Aligning Textual and Model-Based Process Descriptions. Data and Knowledge Engineering, Vol. 118, 2018, No. 1, pp. 25–40, doi: 10.1016/j.datak.2018.09.001.

[5] Leopold, H.—Mendling, J.—Polyvyanyy, A.: Supporting Process Model Validation Through Natural Language Generation. IEEE Transactions on Software Engineering, Vol. 40, 2014, No. 8, pp. 818–840, doi: 10.1109/TSE.2014.2327044.

[6] van der Aa, H.—Leopold, H.—Reijers, H. A.: Comparing Textual Descriptions to Process Models – The Automatic Detection of Inconsistencies. Information Systems, Vol. 64, 2017, pp. 447–460, doi: 10.1016/j.is.2016.07.010.

[7] Rana, M.—Shahzad, K.—Adeel Nawab, R. M.—Leopold, H.—Babar, U.: A Textual Description Based Approach to Process Matching. In: Horkoff, J., Jeusfeld, M., Persson, A. (Eds.): The Practice of Enterprise Modeling (PoEM 2016). Springer, Cham, Lecture Notes in Business Information Processing, Vol. 267, 2016, pp. 194–208, doi: 10.1007/978-3-319-48393-1_14.

[8] Leopold, H.—van der Aa, H.—Pittke, F.—Raffel, M.—Mendling, J.—Reijers, H. A.: Searching Textual and Model-Based Process Descriptions Based on a Unified Data Format. Software and Systems Modeling, Vol. 18, 2019, No. 2, pp. 1179–1194, doi: 10.1007/s10270-017-0649-y.

[9] Leopold, H.—van der Aa, H.—Pittke, F.—Raffel, M.—Mendling, J.—Reijers, H. A.: Integrating Textual and Model-Based Process Descriptions for Comprehensive Process Search. In: Schmidt, R., Guédria, W., Bider, I., Guerreiro, S. (Eds.): Enterprise, Business-Process and Information Systems Modeling (BPMDS 2016, EMMSAD 2016). Springer, Cham, Lecture Notes in Business Information Processing, Vol. 248, 2016, pp. 51–65. doi: 10.1007/978-3-319-39429-9_4.

[10] Erkan, G.—Radev, D. R.: LexRank: Graph-Based Lexical Centrality as Salience in Text Summarization. Journal of Artificial Intelligence Research, Vol. 22, 2004, pp. 457–479, doi: 10.1613/jair.1523.

[11] Shahzad, K.—Shareef, K.—Ali, R. F.—Adeel Nawab, R. M.—Abid, A.: Generating Process Model Collection with Divers Label and Structural Features.

2016 Sixth International Conference on Innovative Computing Technology (INTECH 2016), IEEE, 2016, pp. 644–649, doi: 10.1109/intech.2016.7845083.

[12] Snoeck, M.—Moreno-Montes de Oca, I.—Haegemans, T.—Scheldeman, B.—Hoste, T.: Testing a Selection of BPMN Tools for Their Support of Modelling Guidelines. In: Ralyté, J., España, S., Pastor, Ó. (Eds.): The Practice of Enterprise Modeling (PoEM 2015). Springer, Cham, Lecture Notes in Business Information Processing, Vol. 235, 2015, pp. 111–125, doi: 10.1007/978-3-319-25897-3_8.

[13] Shahzad, K.—Adeel Nawab, R. M.—Abid, A.—Sharif, K.—Ali, F.—Aslam, F.—Mazhar, A.: A Process Model Collection and Gold Standard Correspondences for Process Model Matching. IEEE Access, Vol. 7, 2019, pp. 30708–30723, doi: 10.1109/access.2019.2900174.

[14] Antunes, G.—Bakhshandeh, M.—Borbinha, J.—Cardoso, J.—Dadashnia, S.—Di Francescomarino, Ch.—Dragoni, M.—Fettke, P.—Gal, A.—Ghidini, C. et al.: The Process Model Matching Contest 2015. In: Kolb, J., Leopold, H., Mendling, J. (Eds.): Enterprise Modelling and Information Systems Architectures (EMISA 2015). Gesellschaft für Informatik e.V., Bonn, Lecture Notes in Informatics (LNI), Vol. P248, 2015, pp. 127–155.

[15] Mendling, J.—Reijers, H. A.—van der Aalst, W. M. P.: Seven Process Modeling Guidelines (7PMG). Information and Software Technology, Vol. 52, 2010, No. 2, pp. 127–136, doi: 10.1016/j.infsof.2009.08.004.

[16] Leopold, H.—Mendling, J.—Polyvyanyy, A.: Generating Natural Language Texts from Business Process Models. In: Ralyté, J., Franch, X., Brinkkemper, S., Wrycza, S. (Eds.): Advanced Information Systems Engineering (CAiSE 2012). Springer, Berlin, Heidelberg, Lecture Notes in Computer Science, Vol. 7328, 2012, pp. 64–79, doi: 10.1007/978-3-642-31095-9_5.

[17] Dijkman, R.—Dumas, M.—Van Dongen, B.—Käärik, R.—Mendling, J.: Similarity of Business Process Models: Metrics and Evaluation. Information Systems, Vol. 36, 2011, No. 2, pp. 498–516, doi: 10.1016/j.is.2010.09.006.

[18] Adeel Nawab, R. M.: Mono-Lingual Paraphrased Text Reuse and Plagiarism Detection. Ph.D. thesis, University of Sheffield, 2012.

[19] Baeza-Yates, R.—Ribeiro-Neto, B.: Modern Information Retrieval. ACM Press New York, 1999.

[20] Mendling, J.: Metrics for Process Models: Empirical Foundations of Verification, Error Prediction, and Guidelines for Correctness. 1st Edition. Springer, Berlin, Heidelberg, Lecture Notes in Business Information Processing, Vol. 6, 2008, doi: 10.1007/978-3-540-89224-3.

[21] Bosanac, S.—Štefanec, V.: N-Gram Overlap in Automatic Detection of Document Derivation. 3rd International Conference "The Future of Information Sciences: INFuture2011 – Information Sciences and e-Society", Zagreb, 2011, pp. 373–382.

[22] Salton, G.—Wong, A.—Yang, C.-S.: A Vector Space Model for Automatic Indexing. Communications of the ACM, Vol. 18, 1975, No. 11, pp. 613–620, doi: 10.1145/361219.361220.

[23] Barrón-Cedeño, A.—Rosso, P.—Benedí, J.-M.: Reducing the Plagiarism Detection Search Space on the Basis of the Kullback-Leibler Distance. In: Gel-

bukh, A. (Ed.): Computational Linguistics and Intelligent Text Processing (CICLing 2009). Springer, Berlin, Heidelberg, Lecture Notes in Computer Science, Vol. 5449, pp. 523–534, doi: 10.1007/978-3-642-00382-0_42.

[24] Smith, T. F.—Waterman, M. S.: Identification of Common Molecular Subsequences. Journal of Molecular Biology, Vol. 147, 1981, No. 1, pp. 195–197, doi: 10.1016/0022-2836(81)90087-5.

[25] Needleman, S. B.—Wunsch, Ch. D.: A General Method Applicable to the Search for Similarities in the Amino Acid Sequence of Two Proteins. Journal of Molecular Biology, Vol. 48, 1970, No. 3, pp. 443–453, doi: 10.1016/0022-2836(70)90057-4.

[26] Kiros, R.—Zhu, Y.—Salakhutdinov, R. R.—Zemel, R.—Urtasun, R.—Torralba, A.—Fidler, S.: Skip-Thought Vectors. In: Cortes, C., Lawrence, N. D., Lee, D. D., Sugiyama, M., Garnett, R. (Eds.): Advances in Neural Information Processing Systems 28 (NIPS 2015). Springer, 2015, pp. 3294–3302.

[27] Weidlich, M.—Dijkman, R.—Mendling, J.: The ICoP Framework: Identification of Correspondences Between Process Models. In: Pernici, B. (Ed.): Advanced Information Systems Engineering (CAiSE 2010). Springer, Berlin, Heidelberg, Lecture Notes in Computer Science, Vol. 6051, 2010, pp. 483–498, doi: 10.1007/978-3-642-13094-6_37.

[28] Jabeen, F.—Leopold, H.—Reijers, H. A.: How to Make Process Model Matching Work Better? An Analysis of Current Similarity Measures. In: Abramowicz, W. (Ed.): Business Information Systems (BIS 2017). Springer, Cham, Lecture Notes in Business Information Processing, Vol. 288, 2017, pp. 181–193, doi: 10.1007/978-3-319-59336-4_13.

[29] Dijkman, R.—Dumas, M.—García-Bañuelos, L.: Graph Matching Algorithms for Business Process Model Similarity Search. In: Dayal, U., Eder, J., Koehler, J., Reijers, H. A. (Eds.): Business Process Management (BPM 2009). Springer, Berlin, Heidelberg, Lecture Notes in Computer Science, Vol. 5701, 2009, pp. 48–63, doi: 10.1007/978-3-642-03848-8_5.

[30] Wordnet. `https://wordnet.princeton.edu/`, accessed: 2018-06-23.

[31] Skipthought Source. `https://github.com/ryankiros/skip-thoughts/blob/master/skipthoughts.py`, accessed: 2019-05-25.

[32] Osiński, S.—Stefanowski, J.—Weiss, D.: Lingo: Search Results Clustering Algorithm Based on Singular Value Decomposition. In: Kłopotek, M. A., Wierzchoń, S. T., Trojanowski, K. (Eds.): Intelligent Information Processing and Web Mining. Springer, Berlin, Heidelberg, Advances in Soft Computing, Vol. 25, 2004, pp. 359–368, doi: 10.1007/978-3-540-39985-8_37.

[33] Carrot2 Project Kernel Description. `https://project.carrot2.org/download-workbench-win32-64bit.htm`, accessed: 2019-05-25.

[34] Signavio Process Modeling Tool. `https://www.signavio.com/`, accessed: 2019-05-25.

[35] Baumann, M.—Baumann, M. H.—Jablonski, S.: On Behavioral Process Model Similarity Matching: A Centroid-Based Approach. In: Mayr, H. C., Pinzger, M. (Eds.): Informatik 2016. Gesellschaft für Informatik e.V., Bonn, Lecture Notes in Informatics (LNI), Vol. P259, 2016, pp. 731–732.

[36] Meilicke, Ch.—Leopold, H.—Kuss, E.—Stuckenschmidt, H.—Reijers, H. A.: Overcoming Individual Process Model Matcher Weaknesses Using Ensemble Matching. Decision Support Systems, Vol. 100, 2017, pp. 15–26, doi: 10.1016/j.dss.2017.02.013.

[37] Padmakumar, A.—Saran, A.: Unsupervised Text Summarization Using Sentence Embeddings. Technical Report, University of Texas at Austin, 2016, pp. 1–9.

[38] Yan, Z.—Dijkman, R.—Grefen, P.: Fast Business Process Similarity Search with Feature-Based Similarity Estimation. In: Meersman, R., Dillon, T., Herrero, P. (Eds.): On the Move to Meaningful Internet Systems: OTM 2010. Springer, Berlin, Heidelberg, Lecture Notes in Computer Science, Vol. 6426, 2010, pp. 60–77, doi: 10.1007/978-3-642-16934-2_8.

[39] Yan, Z.—Dijkman, R.—Grefen, P.: FNet: An Index for Advanced Business Process Querying. In: Barros, A., Gal, A., Kindler, E. (Eds.): Business Process Management (BPM 2012). Springer, Berlin, Heidelberg, Lecture Notes in Computer Science, Vol. 7481, 2012, pp. 246–261, doi: 10.1007/978-3-642-32885-5_20.

Syed Irtaza Muzaffar is a Ph.D. scholar at the Punjab University College of Information Technology (PUCIT), University of the Punjab, Lahore. He has M. Phil. in computer science from PUCIT. Currently, he is Lecturer (Visiting) at the University of the Punjab. He has three years of teaching and development experience.

Khurram Shahzad is Assistant Professor at the Punjab University College of Information Technology (PUCIT), University of the Punjab, Lahore. He has his Masters and Ph.D. from KTH – Royal Institute of Technology, Stockholm. He is associated with Information Systems Groups at the Technical University Eindhoven, Eindhoven, and the University of Fribourg, Fribourg. He has published more than 35 papers in international conferences and journals.

Faisal Aslam obtained his Ph.D. from the University of Freiburg, Germany and worked as post-doc at the TU Delft, the Netherlands. He was also a research fellow at the Lund University, Sweden. Currently, he is Assistant Professor at the University of the Punjab. He has seven years of his post-Ph.D. experience. He has published papers in reputed journals and conferences.

Madiha Khalid is a faculty member and Ph.D. candidate at the Punjab University College of Information Technology (PUCIT), University of the Punjab, Lahore, Pakistan. She holds her M.Sc. and B.Sc. in computer science. She has published several papers in reputed journals.

Kamran Malik is Assistant Professor at the Punjab University College of Information Technology (PUCIT), University of the Punjab, Lahore, Pakistan. He has more than 15 years of teaching and development experience. He holds his Ph.D. degree in computer science and he has authored 1 US patent and published 25 papers in reputed journals and conferences. His research interests include natural language processing, machine learning, and data science. He has provided consultancy to many multinational firms on natural language processing, machine learning, and data science.