




# Learning Workflow Embeddings to Improve the Performance of Similarity-Based Retrieval for Process-Oriented Case-Based Reasoning

Patrick Klein , Lukas Malburg , and Ralph Bergmann 

Business Information Systems II, University of Trier, 54286 Trier, Germany

{kleinp,malburgl,bergmann}@uni-trier.de

<http://www.wi2.uni-trier.de>

**Abstract.** In process-oriented case-based reasoning, similarity-based retrieval of workflow cases from large case bases is still a difficult issue due to the computationally expensive similarity assessment. The two-phase MAC/FAC (“Many are called, but few are chosen”) retrieval has been proven useful to reduce the retrieval time but comes at the cost of an additional modeling effort for implementing the MAC phase. In this paper, we present a new approach to implement the MAC phase for PO-CBR retrieval, which makes use of the StarSpace embedding algorithm to automatically learn a vector representation for workflows, which can be used to significantly speed-up the MAC retrieval phase. In an experimental evaluation in the domain of cooking workflows, we show that the presented approach outperforms two existing MAC/FAC approaches on the same data.

**Keywords:** Process-Oriented Case-Based Reasoning · MAC/FAC Retrieval · Graph Embeddings

## 1 Introduction

As more and more workflows are supported and executed electronically, the amount of available data in process repositories as well as the procedural knowledge gathered through past problem-solving experience increases. Such workflows can represent business processes, scientific experiments, repair instructions, or activities from daily life such as cooking recipes. It is valuable to reuse this procedural knowledge since creating workflows from scratch is typically a complex and time-consuming task [18]. Process-Oriented Case-Based Reasoning (PO-CBR) [2,15] can be used for retrieving, reusing, revising, and retaining procedural experiential knowledge represented as workflows. A case base in PO-CBR specifies best-practice workflows that can be (re-)used in similar situations. A critical factor for the performance of a CBR system is the efficiency of case retrieval [7]. The retrieval time is of importance due to its impact on the user’s system acceptance as well as being a requirement in time critical environments where decisions need to be made within clearly defined time boundaries. However, obtaining an acceptable retrieval time is particularly difficult for PO-CBR,

as cases in POCBR are usually represented by semantically labeled graphs leading to a similarity assessment that requires a kind of inexact sub-graph matching, which is computationally expensive [2,12,16].

POCBR research has addressed the issue of efficient retrieval [5,11,12,17] through a two-phase retrieval following the MAC/FAC (“Many are called, but few are chosen”) [9] principle. The retrieval is divided into two phases: The first phase (MAC) utilizes a simplified and often knowledge-poor similarity measure for a fast pre-selection. The second phase (FAC) then applies the computationally intensive graph-based similarity measure to the results of the MAC phase. This method improves the retrieval performance, if the MAC stage efficiently selects a small number of relevant cases. However, there is a risk that the MAC phase reduces the retrieval quality, as it might disregard highly similar cases due to its simplified assessment of the similarity. As a consequence, the retrieval approach for the MAC phase must be designed very carefully. Today, existing approaches [5,12] are based on a manually designed simplified domain specific case representation as well as a related method for the pre-selection of cases, which leads to a significantly increased development effort for the CBR system. The cluster-based retrieval approach introduced by Müller & Bergmann [17] avoids this additional effort but only works well for case bases with a strong cluster structure.

The aim of this paper is to present a novel approach for the design of a MAC phase for POCBR, which automatically learns an appropriate simplified case representation in the form of *workflow embeddings*. Thereby, we avoid the manual domain modeling for the MAC phase. For learning workflow embeddings, we investigate the general-purpose neural embedding model *StarSpace* [22], which has shown impressive performance on many different tasks such as text classification, entity ranking, and also graph embeddings. We aim at applying StarSpace to entities, which are workflows described by sets of linked task and data nodes.

The next section introduces previous work on POCBR, including semantic workflow representation, similarity assessment, and MAC/FAC approaches for POCBR. In Section 3, we present our approach for learning workflow embeddings and their use for retrieval in the MAC phase. An experimental evaluation of our approach in the domain of cooking recipes is presented in Section 4. Finally, Section 5 concludes the results and discusses future work.

## 2 Foundations and Previous Work

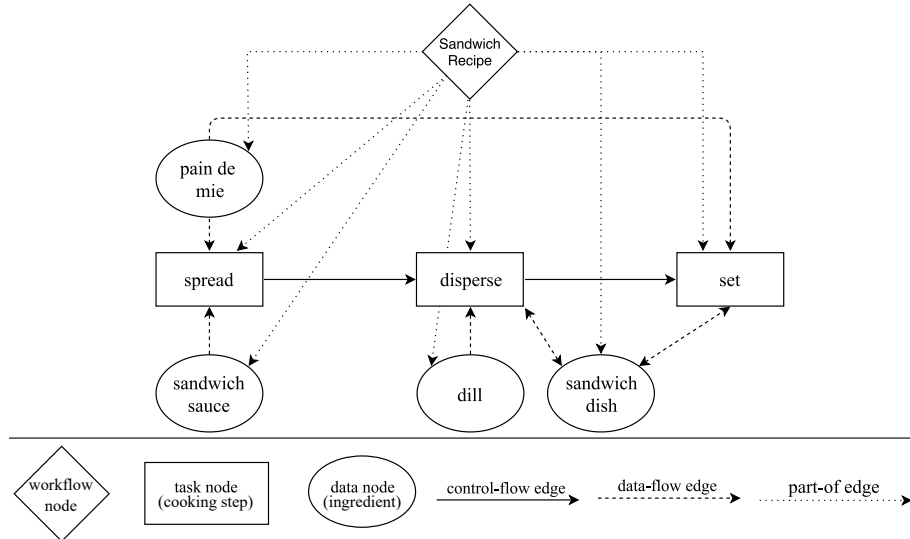
Process-Oriented Case-Based Reasoning [2,15] deals with the integration of CBR with Process-Aware Information Systems (PAISs) [8]. An example of a certain type of PAISs are workflow management systems [8]. Using POCBR, workflow developers are supported with best-practice workflows from a case base during their development process. Thus, POCBR supports the development of workflows as an experience-based activity [2,15]. POCBR methods require an appropriate case representation for workflows as well as a similarity measure that assesses the suitability of a workflow for a new problem situation.

## 2.1 Semantic Workflow Representation

In general, workflows are used for the automation of a defined sequence of tasks that can exchange inputs and outputs in order to achieve an overarching corporate objective [10]. Therefore, the ordering of tasks is modeled through structures such as sequences, parallel (AND split/join) as well as alternative (XOR split/join) and loops, which result together in the so-called control-flow. Additionally, tasks consume inputs and produce outputs, both of which can be physical or virtual in nature, and, along with their relationship between tasks, they form the data-flow.

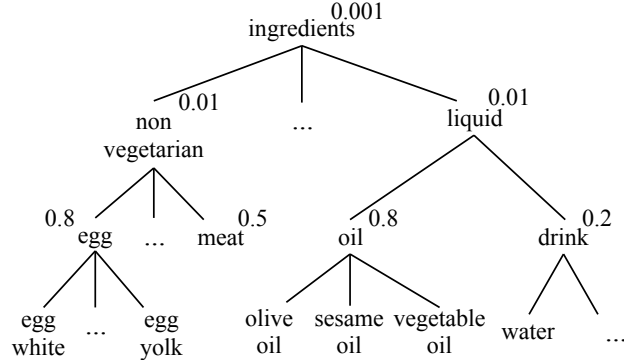
In order to represent workflows, we use semantically labeled directed graphs named *NEST* graphs by Bergmann & Gil [2]. A *NEST* graph is a quadruple  $G = (N, E, S, T)$  where  $N$  is a set of nodes and  $E \subseteq N \times N$  represents the edges between nodes. Semantic descriptions  $S : N \cup E \rightarrow \Sigma$  can be used for semantic enrichment of nodes or edges. Semantic descriptions are based on a semantic meta data language  $\Sigma$  and are domain-dependent. Additionally, each node and edge is annotated with a type  $T : N \cup E \rightarrow \Omega$ . The set  $\Omega$  is predefined for nodes (e.g., task and data nodes) and edges (e.g., control-flow and data-flow edges).

In this paper, we use the well-known cooking domain to illustrate our approach and to perform an experimental evaluation. Thus, workflows are cooking recipes, tasks represent cooking steps, and data items take the role of ingredients. Figure 1 shows an example of a *NEST* graph representing a simple sandwich recipe, consisting of three task nodes (cooking steps) and four data nodes (ingredients).



**Fig. 1.** Exemplary Cooking Workflow for a Sandwich Dish

In the cooking domain, the semantic meta data language is defined by taxonomic ontologies, one for ingredients and one for cooking steps. These ontologies are complete in the sense that all items occurring in the recipes are also included in the ontology. Figure 2 illustrates a fraction of the used ingredients taxonomy.



**Fig. 2.** Part of the Data Taxonomy Showing the Modeled Similarity between Ingredients

## 2.2 Similarity Assessment

Based on the *NEST* graph format, an assessment of the similarity between workflows through consideration of its constituents as well as the link structure is possible. Therefore, the local similarity of nodes and edges is defined based on the semantic meta data language  $sim_{\Sigma} : \Sigma \times \Sigma \rightarrow [0, 1]$ . For task and data nodes, the taxonomies are used to derive their similarity using a taxonomic similarity measure that is based on the assignment of similarity values to the inner nodes of the taxonomy [4]. Building on that, the global similarity between a query workflow  $QW$  and a workflow from the case base  $CW$  is calculated by a type-preserving, partial, injective mapping function  $m$  from the nodes and edges of  $QW$  to those of  $CW$ . With respect to a mapping  $m$ , the local similarities are aggregated, leading to a similarity value  $sim_m(QW, CW)$  based on which the overall workflow similarity  $sim(QW, CW)$  is determined by the best possible mapping  $m$  as follows (see [2] for more details):

$$sim(QW, CW) = \max \{ sim_m(QW, CW) \mid \text{admissible mapping } m \} \quad (1)$$

Thus, computing the similarity between a query and a single case requires solving an optimization problem, for which we have proposed to use  $A^*$  search. Bergmann & Gil also developed a parallelized version of the  $A^*$  search, which is complete but still not sufficiently fast for large case bases.

### 2.3 MAC/FAC Retrieval for POCBR

To overcome the issue of long retrieval times in POCBR, two-phase MAC/FAC retrieval approaches for workflows have been introduced [5,11,12]. The major difficulty with MAC/FAC retrieval in general is the definition of the filter condition of the MAC stage. Since cases that are not selected by the MAC stage will not appear in the overall retrieval result, the completeness of the retrieval can be easily violated if the filter condition is too restrictive. Hence, retrieval errors, i.e., missing cases will occur. On the other hand, if the filter condition is less restrictive, the number of pre-selected cases may become too large, resulting in a low retrieval performance. To balance retrieval error and performance, the filter condition should be a good approximation of the similarity measure used in the FAC stage, while at the same time it must be efficiently computable to be applicable to a large case base in the MAC stage.

Bergmann & Stromer [5] addressed this problem by adding a feature-based domain specific case representation of workflows, which simplifies the original representation while maintaining the most important properties relevant for similarity assessment. The MAC stage then selects cases by performing a similarity-based retrieval using an appropriately modeled similarity measure. The number of cases selected in the MAC phase can be controlled by a parameter called *filter size*  $FS$ , i.e., the MAC stage retrieves the  $FS$ -most similar cases using feature-based retrieval. The choice of the filter size determines the behavior of the overall retrieval method with respect to retrieval speed and error in the following manner: the smaller the filter size, the faster the retrieval but the larger the retrieval error will become.

In order to avoid the additional modeling effort for the feature-based representation, Müller & Bergmann [17] developed a MAC/FAC approach that is based on the structuring of the case base into clusters of similar cases. Therefore, a binary cluster-tree is learned, which hierarchically partitions the case base into sets of similar cases. Traversing the cluster-tree allows finding clusters with cases similar to the query, thus reducing the number of required similarity computations. Again, a filter size parameter  $FS$  is used to determine the number of cases transferred to the FAC phase. This algorithm did not reach quality and retrieval speed of the feature-based MAC/FAC approach, but shows acceptable performance if the case base has a clear cluster structure.

## 3 Learning Workflow Embeddings for MAC Retrieval

We now present our approach for the design of a MAC phase for POCBR retrieval that avoids the manual construction of a simplified representation and a related similarity measure. The main idea is to automatically transform the original semantic workflow representation by use of an embedding method.

As workflows are represented as graphs, *graph embedding techniques* [6] are useful, as their main purpose is to address the complexity problems of many graph analytic methods by converting the graph data into a low dimensional

space. The transformation is performed such that the graph structural information and the graph properties are preserved as best as possible. Thus, we expect that graph embeddings are also helpful for designing a MAC retrieval approach for POGBR. There is already a large range of graph embedding methods reported in the literature [6], which enable to learn embeddings for nodes, edges, or whole graphs or sub-graphs. We decided to chose a recently proposed algorithm called *StarSpace* [22], a general purpose neural embedding model, which provides an efficient strong baseline on various tasks. In particular, it can be used for the embedding of multi-relational graphs.

In the remainder of this section, we first introduce the general StarSpace algorithm before we describe how we apply it to learn workflow embeddings as simplified representations of POGBR cases. Afterwards, the straightforward application during the MAC retrieval phase is explained.

### 3.1 Embedding Learning with StarSpace

StarSpace embeds entities of different types into a single, fixed-length dimensional space with the result of having entity representations that are comparable to each other. It learns to rank a set of entities, documents, or objects given a query entity, document, or object, where the query is not necessarily of the same type as the items in the set [22]. As we will further see, this is an important property explored in our application.

The basic concept of StarSpace is that it learns embeddings for entities that consist of one or more features. As we will see, in our case a feature is a node, an edge, or a whole workflow. The algorithm maintains a dictionary  $D$  of features and assigns to each feature an embedding vector, which is stored in an embedding matrix  $F \in R^{|D| \times d}$ , where  $|D|$  is the number of features and  $d$  is the size of the dimension of the embedding space. Each row  $F_i$  is the embedding for a feature  $i$ . An entity  $a$  is embedded by a vector representing the sum of the embeddings  $F_i$  of the features  $i$  it consists of so that  $a = \sum_{i \in a} F_i$ .

To learn the embeddings for each feature, StarSpace compares the similarity of two entities  $a$  and  $b$  that are provided from the set of training data  $E^+$  against randomly generated entities, which constitute the set of negative examples  $E^-$ . The goal is that entities, which are labeled as similar based on the training data  $E^+$  will be rated higher by a margin  $h$  than randomly generated samples from  $E^-$ . During learning, the following ranking loss function  $L$  is minimized by stochastic gradient descent:

$$\sum_{\substack{(a,b) \in E^+ \\ b^- \in E^-}} L^{batch}(sim(a,b), sim(a,b_1^-), \dots, sim(a,b_k^-)) \quad (2)$$

The similarity function  $sim$  can be cosine or dot product, while cosine usually leads to better results. The margin ranking loss for  $sim(a,b)$  and  $sim(a,b^-)$  is calculated by  $\max\{0, h - sim(a,b) + sim(a,b^-)\}$ .

How exactly positive training examples are generated from the set  $E$  is task-specific. StarSpace provides a large number of options, which makes StarSpace

a general-purpose embedding approach. Since a straightforward word embedding approach to transform the labels of the graph components into a bag of words leads to less promising results in our previous experiments, we apply the multi-relational knowledge graph approach for generating training examples. This option enables to learn embeddings of a graph represented as triples  $(h, r, t)$  consisting of a head concept  $h$ , a relation  $r$ , and a tail concept  $t$ .

### 3.2 Learning Embeddings for Workflow Graphs

In order to learn embeddings for workflow graphs, a workflow becomes an entity in StarSpace that is described by a set of tasks and data nodes, each of which are features for which an embedding vector is learned. Also the main workflow node (i.e., the top node in Figure 1) is a feature of the workflow entity.

To construct the training input for StarSpace, all *NEST* graphs from the case base are serialized into a triple format similar to the *Turtle* notation for RDF graphs<sup>1</sup>. This representation fully represents the semantic workflow and can be used for learning embeddings by StarSpace. The following triples can be extracted from the workflow example depicted in Figure 1:

Sandwich_Recipe	hasTask	spread
Sandwich_Recipe	hasInput	pain de mie
Sandwich_Recipe	hasInput	sandwich sauce
Sandwich_Recipe	hasTask	disperse
Sandwich_Recipe	hasInput	dill
Sandwich_Recipe	hasInput	sandwich dish
Sandwich_Recipe	hasTask	set

Moreover, we can add further triples that represent the connections between tasks and data nodes:

pain de mie	dataflow	spread
sandwich sauce	dataflow	spread
pain de mie	dataflow	set
dill	dataflow	disperse
sandwich dish	dataflow	disperse
disperse	dataflow	sandwich dish
sandwich dish	dataflow	set
set	dataflow	sandwich dish
spread	controlflow	disperse
disperse	controlflow	set

Based on this data, we can use the method proposed by StarSpace to learn multi-relational graphs. For this purpose, each triple  $(h, r, t)$  results in two training examples: the left entity ( $h$ ) is predicted based on the relation ( $r$ ) and the right feature vector ( $t$ ) and the right entity ( $t$ ) is predicted based on the left

<sup>1</sup> <https://www.w3.org/TR/2014/REC-turtle-20140225/>

( $h$ ) and the reverse relation ( $\bar{r}$ ) feature vector. Since we are only interested in the prediction of workflow graphs based on their task and data nodes, we have adapted the training such that only triples with *hasTask* and *hasInput* relations are used and we set the main workflow node as the label to predict. In other words, by using the relation feature and the feature of the task or data node, we try to predict the workflow. The learning algorithm of StarSpace then optimizes our feature vectors accordingly.

As a result of the learning phase, StarSpace produces feature vectors for each node of each workflow in the case base. Please note that equivalent nodes in different cases (e.g., nodes referring to the same ingredient or the same cooking step) are represented only once, thus have the same embedding vector. However, each workflow in the case base has its own main workflow node. We use the embedding vector learned for this main workflow node as workflow embedding representation of the case to be used in the MAC phase of the POCBR retrieval.

### 3.3 Plausibility of Similarities of Learned Embeddings

As a side effect of the described learning approach, all items of the task and data ontology used as semantic annotation for the nodes in the case workflows also occur as features in the StarSpace dictionary and thus have an embedding vector attached. Consequently, their similarity can be assessed by using the similarity measure for which StarSpace performs its optimization, i.e., the cosine similarity. In order to check whether the resulting similarity values are plausible, we performed a spot-checking of selected ingredient pairs. We compared the similarity value resulting from the embedding with the similarity value determined using the ingredient taxonomy (see Figure 2), i.e., the local node similarity measure used in the graph-based similarity measure of the FAC phase (see Section 2.1.). Table 1 illustrates selected similarity comparisons with value ranges adjusted to the interval  $[0, 1]$ . For all three pairs of ingredients, which all have a rela-

**Table 1.** Comparing Selected Similarity Values

<b>Query</b>	<b>Result</b>	<b>Embedding Similarity</b>	<b>Taxonomic Similarity</b>
Egg White	Egg Yolk	0.854	0.8
Coconut	Pineapple	0.705	0.6
Bananas	Strawberries	0.695	0.6

tively high similarity according to the manually modeled similarity measure, the learned embedding similarity is also quite high, which is a first indication that both measures are inline with each other. This observation could also be made for many more similarity pairs that we have checked in a random fashion but



certain negative examples could also be found. In total, however, this inspection is a first hint that the embedding approach is able to learn useful similarity knowledge.

### 3.4 Embedding-Based Workflow Retrieval

Using the learned workflow embeddings, the implementation of the MAC retrieval stage is quite straightforward and similar to the approach used for the feature-based MAC/FAC approach [5]. Prior to retrieval, the workflow embeddings must be learned for each case in the case base in an offline-phase. MAC retrieval then simply performs a linear search for the *FS*-most similar cases using the workflow embedding representation and the similarity measure used by StarSpace, i.e., the cosine measure. Thus, the parameter *FS* is the filter size for the MAC phase.

In this process, however, one aspect is less obvious, i.e., how the embedding vector of the query is determined. Typically, a query is not an already existing workflow in the case base, but a new workflow or even only a partial workflow, just consisting of a small number of nodes and edges. Consequently, there is no workflow embedding vector for the main workflow node of the query available, as this node does not exist in the StarSpace directory. To construct the embedding vector for the query, we make use of the StarSpace property that all items are embedded in the same common embedding space; there is no difference between different types of features. Given this, we construct the embedding vector of the query using the bag of features approach, i.e., by adding the embedding vectors of the task and data nodes the query consists of. This can be done at least for those nodes that previously occurred in the case base and that are thus also present in the dictionary of StarSpace.

As an example, consider we would use the workflow from Figure 1 as a query. The resulting query embedding  $q$  would be determined as follows:

$$q = F_{spread} + F_{disperse} + F_{set} + F_{pain\_de\_mie} + F_{sandwich\_sauce} + F_{dill} \\ + F_{sandwich\_dish}$$

## 4 Experimental Evaluation

In this section, we present the evaluation setup and the results. To determine the suitability of our approach, we compare our results with the MAC/FAC retriever by Bergmann & Stromer [5] and with the results of the cluster-based retriever by Müller & Bergmann [17]. For this purpose, we implemented the presented approach in the POCBR component of the CAKE framework<sup>2</sup>. We used the StarSpace implementation available at GitHub<sup>3</sup> and integrated it into the CAKE framework. The StarSpace implementation is started via a command line statement.

<sup>2</sup> <http://procake.uni-trier.de>

<sup>3</sup> <https://github.com/facebookresearch/StarSpace>

## 4.1 Hypotheses

In our experimental evaluation, we investigate the following hypotheses:

- H1** The embedding-based retriever provides at least as good results as the MAC/FAC retriever using the feature-based representation in the MAC phase [5].
- H2** The embedding-based retriever achieves better results than the cluster-based retriever [17] for case bases without cluster structure.

The first hypothesis expresses the expectation that the embedding-based retriever is as good as the feature-based retriever although no manual modeling effort has been invested. This leads to a significant benefit, since manual knowledge modeling is often complex and time-consuming. The second hypothesis claims that the embedding-based retriever is independent of the case distribution and thus more universally applicable compared to the alternative approach, which also avoids manual modeling of the MAC phase.

## 4.2 Experimental Setup

The evaluation is conducted on a case base with 1529 case workflows and 200 query workflows, taken from the extraction of case workflows from cooking recipes from Allrecipes<sup>4</sup> by Schumacher et al. [20]. The case workflows in the case base and the query workflows are the same as those used in the evaluation of the cluster-based retriever – named as CB-I [17] – and the MAC/FAC retriever using the feature-based representation [5]. Each workflow case contains 11 nodes on average and a corresponding taxonomic ontology of 208 individual ingredients and 225 cooking preparation steps is used. For learning our embedding model, the 1529 case workflows are serialized into 18.169 triples that represent *part-of* (*hasTask* or *hasInput*) edges. For simplicity reasons, parallel (*AND*) and alternative (*XOR*) sequences are disregarded as training data. Since our embedding model that learned on the completed graph structure has performed slightly worse than those only learned on triples with *hasTask* and *hasInput* relations, we only discuss results for our best model learned on this smaller training data set.

As a starting point for hyper-parameter optimization, the default settings are used. We adjusted the values of the *margin* to 0.35, the embedding’s *dimension size* to 200, the number of *training epochs* to 200, and the *similarity measure* to cosine. These changes are based on manual inspections of nearest neighbor results by using the projection of individual workflow cases from the case base and thus maximizing the similarity to the case itself and magnifying the selectivity to other workflow cases. The StarSpace learning phase only took approximately 4 minutes on a PC with an Intel i9-7900X CPU @3.30 GHz and 64 GB RAM running Linux Ubuntu.

<sup>4</sup> <https://allrecipes.com/>

In order to assess the validity of the two hypotheses, we evaluate the retrieval time (MAC+FAC phase) and the retrieval quality for various parameter combinations. For retrieval quality, we use the same quality criterion (see Formula 3) as in previous work by Müller & Bergmann [17]. For this purpose, it is examined if workflow cases from the set of the  $k$ -most similar case workflows ( $MSC(QW, k)$ ) that are retrieved by the  $A^*$  parallel retriever (i.e., the gold standard without any MAC pre-selection), are also retrieved by the MAC/FAC retriever under investigation. If not all case workflows are contained in the *result list* ( $RL$ ) of the corresponding MAC/FAC approach, the quality decreases proportional to the similarity of this workflow to the query. Thus, if a highly similar case is omitted, the negative impact on the quality is stronger than if a case with a low similarity is missing.

$$quality(QW, RL) = 1 - \frac{1}{|RL|} \cdot \sum_{CW \in \{MSC(QW, |RL|) \setminus RL\}} sim(QW, CW) \quad (3)$$

### 4.3 Experimental Results

We compared the three MAC/FAC approaches using different numbers of case workflows to be retrieved ( $k$ ) and using different filter sizes ( $FS$ ) for the MAC phase. We show the retrieval time in seconds and the quality value according to Formula 3. All results are average values over all queries.

Table 2 shows the results of the comparison of the feature-based retriever with our embedding-based approach. Each row in the table represents one particular parameter setting. Please note that the  $k$ -value is the same for both retrievers in a row (so both have to solve the same retrieval task) but the used filter size parameter is different and optimized for each of the two approaches. In particular, we have chosen the  $FS$  value for the embedding-based retriever in a way that the achieved retrieval quality is quite the same as what is achieved by the feature-based retriever. In addition, we illustrate the number of matches (*Hits*) without considering the corresponding rank. When we investigate the retrieval time, we can see that the embedding-based retriever is as fast as the feature-based retriever for larger values of  $k$  but clearly faster for small values of  $k$ . Thus, Hypothesis H1 is clearly confirmed. With respect to the complete  $A^*$  parallel retriever (its retrieval time is shown in the right column of Table 2), we achieve a speedup of a factor 2.3 to 10.8. When looking at the results in more detail, we can see that the embedding-based retriever requires a significantly higher filter size to achieve the same quality values. Thus, it does not approximate the FAC similarity as well as the feature-based retriever, which leads to more irrelevant cases among within the list of top-ranked cases resulting from the MAC phase. However, the speed-up in MAC similarity assessment is so large that we can compensate this by affording a larger filter size, thereby shifting retrieval effort from the MAC phase to the FAC phase. Overall, this seems to be an effective approach in our experiments.

To compare the embedding-based retriever with the cluster-based retriever, we report the values for  $k$ ,  $FS$ , quality, and retrieval time as published in [17].

**Table 2.** Comparison of the Feature-Based and the Embedding-Based Retriever

Feature Graph MAC-FAC					Embedding Graph MAC-FAC					A* Parallel
FS	k	Hits	Quality	Time	FS	k	Hits	Quality	Time	Time
5	5	2.37	0.79	0.166	25	5	2.44	0.79	0.083	0.896
50	5	4.69	0.98	0.262	250	5	4.61	0.97	0.251	
10	10	5.24	0.80	0.190	50	10	5.75	0.81	0.126	0.982
80	10	9.48	0.98	0.336	300	10	9.26	0.97	0.340	
25	25	14.37	0.82	0.259	100	25	15.77	0.84	0.228	1.098
100	25	23.17	0.97	0.465	350	25	22.75	0.97	0.469	

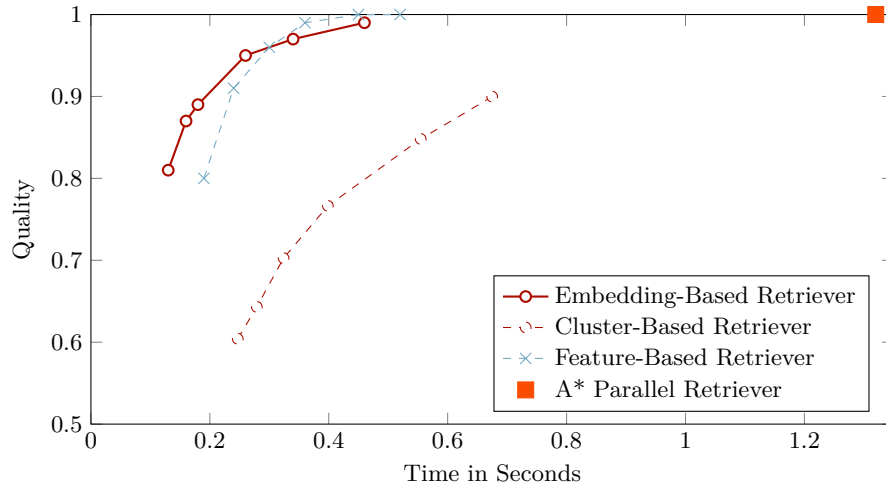
To compensate for the improved hardware capabilities under which we measure the results of the embedding-based approach, we adjust the previously reported time values by a factor of 0.80556. This value is carefully determined based on the average improvement of the  $A^*$  parallel and the feature-based MAC/FAC retriever. The results are shown in Table 3. Please note that in this experiment, the filter size is the same for both approaches. The results clearly demonstrate that the embedding-based retriever outperforms the cluster-based approach in retrieval time and quality in all examined parameter settings. Since the case base used for evaluation throughout the whole experiment has no cluster structure, Hypothesis H2 is clearly confirmed.

**Table 3.** Overview of Results between the Cluster-Based and the Embedding-Based Approach

Cluster Graph		Embedding Graph			
MAC-FAC		MAC-FAC			
FS	k	Quality	Time	Quality	Time
10	10	0.60	0.199	0.65	0.066
50	10	0.70	0.261	0.81	0.126
100	10	0.77	0.321	0.89	0.181
25	25	0.61	0.254	0.69	0.101
50	25	0.67	0.300	0.75	0.166
100	25	0.74	0.371	0.84	0.228
50	50	0.65	0.338	0.74	0.169
100	50	0.72	0.420	0.81	0.308

Finally, we summarize our results in a way that allows to compare all three MAC/FAC retrieval approaches. In Figure 3, we present a plot that characterizes each retrieval approach as it directly relates retrieval time and retrieval quality for various values of  $FS$ . The value of  $k$  is fixed to 10 for this comparison. As shown by the plots, the embedding-based MAC/FAC retriever provides the high-

est quality in relation to retrieval time but does not fully reach the perfect quality value of 1 such as the feature-based approach. Hence, the speed advantage of the embeddings used to compensate for the poorer quality through increasing the filter size comes to an end at high-quality values because the marginal utility of an increased filter size diminishes. However, the loss of quality in this range is so small that it does not justify the effort involved in implementing a manually designed feature-based approach. Figure 3 also clearly shows the impressive advantage of the embedding-based retriever over the cluster-based approach.



**Fig. 3.** Retrieval Time and Quality for all Retrievers with  $k = 10$

## 5 Conclusion, Related and Future Work

We presented a new MAC/FAC approach for the retrieval of semantic workflows in POCBR, which is based on a novel general-purpose neural embedding approach. It enables to learn a vector representation for workflows that can be efficiently compared using the cosine similarity measure. The fact that this embedding approach is able to embed features of different kinds in the same embedding space allows us to efficiently determine also an embedding for a query workflow for which an embedding vector cannot be determined in advance. We could show that the presented approach achieves a performance, which is comparable to a feature-based MAC/FAC retrieval that works with manually modeled case representation and similarity measures for the MAC phase. As a result, an efficient MAC phase is now available fully automatically by means of machine learning and comes at no cost (except for the offline computation time

to be invested for training). The only previously available alternative approach to construct a MAC phase through learning is clearly outperformed in terms of retrieval time and quality.

To our knowledge, the use of neural embedding approaches for implementing the MAC phase in CBR has not yet been discussed in the literature before. Most similar is probably our work on retrieval of argumentation graphs [3] in which we use word embeddings as local similarity measures within a graph similarity measure but also as similarity measure for the MAC phase of retrieval. Not for MAC/FAC retrieval, but for case indexing in general, Metcalf & Leake [14] proposed several embedding techniques, include a knowledge graph embedding method in the domain of medical cases. In addition, the use of neural embedding approaches has been recently discussed in the CBR literature primarily for local similarity measures in textual CBR applications (see e.g., [1,21]).

In future work, we aim to extend our experimental evaluation by using case bases from other domains and including workflow cases with higher complexity. Given the fact that the current embedding approach is not particularly designed to predict the modeled graph-based similarity measure, we still see potential for further improvements. Thus, we aim to investigate the idea to train a siamese network on top of an embedding network using the graph-based similarity values of case pairs from the case base. Furthermore, we propose to examine how the graph structure (e.g., data-flow and control-flow edges) and semantic annotations (e.g., amounts of ingredients) could be better considered during learning. An promising approach that will be considered in future work is presented by Li et al. [13]. Thereby, we hope to improve prediction of the current MAC phase, which would lead to further improvements in retrieval time and quality. In addition, we will explore the idea of an incremental MAC/FAC approach [19], which successively increases the filter size based on an analysis of the FAC similarity of the found cases.

**Acknowledgments.** This work is funded by the German Research Foundation (DFG) under grant No. BE 1373/3-3.

## References

1. Amin, K., Kapetanakis, S., Althoff, K., Dengel, A., Petridis, M.: Answering with Cases: A CBR Approach to Deep Learning. In: Case-Based Reasoning Research and Development - 26th International Conference, ICCBR 2018, Stockholm, Sweden, July 9-12, Proceedings. LNCS, vol. 11156, pp. 15–27. Springer (2018)
2. Bergmann, R., Gil, Y.: Similarity assessment and efficient retrieval of semantic workflows. *Information Systems* **40**, 115–127 (2014)
3. Bergmann, R., Lenz, M., Ollinger, S., Pfister, M.: Similarity Measures for Case-Based Retrieval of Natural Language Argument Graphs in Argumentation Machines. In: Proc. of the Thirty-Two International Florida Artificial Intelligence Research Society Conference, FLAIRS 2019, Florida, May 19-22. (2019)
4. Bergmann, R., Stahl, A.: Similarity Measures for Object-Oriented Case Representations. In: Advances in Case-Based Reasoning, 4th European Workshop, EWCBR-98, Dublin, Ireland, September 1998, Proceedings. pp. 25–36 (1998)

5. Bergmann, R., Stromer, A.: MAC/FAC Retrieval of Semantic Workflows. In: Proc. of the Twenty-Sixth International Florida Artificial Intelligence Research Society Conference, FLAIRS 2013, Florida, May 22-24. AAAI Press (2013)
6. Cai, H., Zheng, V.W., Chang, K.C.: A Comprehensive Survey of Graph Embedding: Problems, Techniques, and Applications. *IEEE Trans. Knowl. Data Eng.* **30**(9), 1616–1637 (2018)
7. Dalal, S., Athavale, V., Jindal, K.: Case retrieval optimization of Case-based reasoning through Knowledge-intensive Similarity measures. *Int. Journal of Computer Applications* **34**(3), 12–18 (2011)
8. Dumas, M., van der Aalst, W.M.P., ter Hofstede, A.H.M. (eds.): *Process-Aware Information Systems: Bridging People and Software Through Process Technology*. Wiley (2005)
9. Forbus, K.D., Gentner, D., Law, K.: MAC/FAC: A Model of Similarity-Based Retrieval. *Cognitive Science* **19**(2), 141–205 (1995)
10. Hollingsworth, D.: *Workflow Management Coalition - The Workflow Reference Model: Document Number TC00-1003 - Version 1.1* (1995)
11. Kendall-Morwick, J., Leake, D.: A study of two-phase retrieval for process-oriented case-based reasoning. In: *Successful Case-based Reasoning Applications-2*, pp. 7–27. Springer (2014)
12. Kendall-Morwick, J., Leake, D.: On tuning two-phase retrieval for structured cases. In: *ICCBR-Workshop on Process-oriented CBR*. pp. 25–34. Lyon (2012)
13. Li, Y., Gu, C., Dullien, T., Vinyals, O., Kohli, P.: Graph Matching Networks for Learning the Similarity of Graph Structured Objects. In: *Proc. of the 36th Int. Conference on Machine Learning, ICML 2019, 9-15 June, California*. Proc. of Machine Learning Research, vol. 97, pp. 3835–3845. PMLR (2019)
14. Metcalf, K., Leake, D.: Embedded Word Representations for Rich Indexing: A Case Study for Medical Records. In: *Case-Based Reasoning Research and Development - 26th International Conference, ICCBR 2018, Stockholm, Sweden, July 9-12, Proceedings*. LNCS, vol. 11156, pp. 264–280. Springer (2018)
15. Minor, M., Montani, S., Recio-García, J.A.: Process-oriented Case-based Reasoning. *Inf. Syst.* **40**, 103–105 (2014)
16. Montani, S., Leonardi, G.: Retrieval and clustering for supporting business process adjustment and analysis. *Information Systems* **40**(0), 128 – 141 (2014)
17. Müller, G., Bergmann, R.: A Cluster-Based Approach to Improve Similarity-Based Retrieval for Process-Oriented Case-Based Reasoning. In: *ECAI 2014 - 21st European Conference on Artificial Intelligence*. pp. 639–644. IOS Press (2014)
18. Müller, G.: *Workflow Modeling Assistance by Case-based Reasoning*. Springer Fachmedien Wiesbaden (2018)
19. Schumacher, J., Bergmann, R.: An Efficient Approach to Similarity-Based Retrieval on Top of Relational Databases. In: *Advances in Case-Based Reasoning, 5th European Workshop, EWCBR 2000, Trento, September 6-9, Proceedings*. LNCS, vol. 1898, pp. 273–284. Springer (2000)
20. Schumacher, P., Minor, M., Walter, K., Bergmann, R.: Extraction of procedural knowledge from the web: a comparison of two workflow extraction approaches. In: *Proceedings of the 21st World Wide Web Conference*. pp. 739–747. ACM (2012)
21. Terada, E.: The Writer’s Mentor. In: *Proc. of ICCBR 2018 Workshops co-located with the 26th Int. Conference, ICCBR 2018, Sweden, July 9-12*. pp. 229–233 (2018)
22. Wu, L.Y., Fisch, A., Chopra, S., Adams, K., Bordes, A., Weston, J.: Starspace: Embed all the things! In: *Thirty-Second AAAI Conference on Artificial Intelligence* (2018)