MDPI

Article

Smart Multimedia Information Retrieval

Stefan Wagenpfeil [1,*], Paul Mc Kevitt [2], and Matthias Hemmje [1]

[1] Faculty of Mathematics and Computer Science, University of Hagen, Universitätsstrasse 1, D-58097 Hagen, Germany; felix.engel@fernuni-hagen.de (F.E.); Matthias.hemmje@fernuni-hagen.de (M.H.)
[2] Academy for International Science & Research (AISR), Derry/Londonderry, Ireland; p.mckevitt@aisr.org.uk
* Correspondence: stefan.wagenpfeil@fernuni-hagen.de

Abstract: The area of Multimedia Information Retrieval (MMIR) faces two major challenges: the enormously growing number of Multimedia Objects (i.e., images, videos, audio, text files), and the fastly increasing level-of-detail of these objects (e.g., the number of pixels in images). Both challenges lead to a high demand of scalability, semantic representations, and explainability of MMIR processes. Smart MMIR solves these challenges by employing Graph Codes as an indexing structure, attaching semantic annotations for explainability, and employing application profiling for scaling, which results in human understandable, expressive, and interoperable MMIR. The mathematical foundation, the modeling, implementation detail, and experimental results are shown in this paper, which confirm, that Smart MMIR improves MMIR in the area of efficiency, effectiveness, and human understandability.

Keywords: indexing, retrieval, explainability, semantic, multimedia, feature graph, graph code, information retrieval

Citation: Wagenpfeil, S.; Mc Kevitt, P.; Hemmje, M. Smart Multimedia Information Retrieval. *Journal Not Specified* **2021**, *1*, 0. https://doi.org/

Received:
Accepted:
Published:

1. Introduction and Motivation

Multimedia is everywhere! – This describes the current state of the art of information and digital media representation in everyone's daily life. All of us are living in a world, where digital media (i.e., multimedia objects like images, video, text, audio) communicate and represent information of any kind, at any time, for any topic, and any target group. Remarkable statistics from Social Media [1] outline, that every single minute as of April 2022, 66,000 photos are shared in Instagram, 500 hours of video are uploaded to Youtube, 2,430,000 snaps are shared on Snapchat, 1,700,000 elements of multimedia content are posted on Facebook, and 231,400,000 E-Mails with media are sent. These large volumes are constantly increasing, which, of course, leads to challenges for the underlying infrastructure and information retrieval systems. In addition, all these digital media objects continue evolving and, e.g., also constantly increase their level-of-detail (i.e., the amount of transported information), as well. Current Smartphones, like the Xiaomi 12T Pro have camera sensors with 200 Megapixel producing images with an enhanced level-of-detail. And the greater level-of-detail a multimedia object has, the more information can be stored, which needs to be maintained, indexed, visualized, distributed, and also retrieved.

In this paper, we summarize previous work from an application perspective and provide solutions for the open challenges of each problem area. The resulting callenges for Multimedia Information Retrieval (MMIR) can be summarized in three major problem areas: 1) Interoperability and Integration, 2) Scalability, and 3) Explainability and Expressiveness:

- in the area of *interoperability and integration*, applications require flexible, configurable, and exchangeable processing flows which are also distributable through organisational units or computational instances. This means, that the extraction of Multimedia features and their integration can be different depending on an application's focus. Furthermore, the increasing number of feature extractors requires a mechanism to integrate features from various extractors, detect inconsistencies, and calculate the relevance of each feature.

- in the area of *scalability*, the high volume of Multimedia objects and their increasing level-of-detail needs to be reflected by application architectures for the distribution of MMIR processing steps. Scalability becomes more important for modern, cloud-based architectures.
- the increase in interoperability and integration, as well as the improved scalability also need to be reflected in the area of *explainability and expressiveness*. Here, further User Interface (UI) components explaining certain MMIR processing steps are required as well as further techniques for content based validation and optimization.

We use the term "Smart MMIR" to describe systems, algorithms, software, or user-interfaces that provide solutions for these three problem areas. "Smart MMIR" thus describes expressive, scalable, interoperable, explainable and human understandable MMIR solutions. In previous work [2][3][4], we already introduced, defined, and evaluated the core components, which contribute to Smart MMIR. However, the interoperability of these components and a corresponding formal model is a foundation for further improvements in the problem areas, which were mentioned above. In this paper, we describe formally these improvements, align them with, or base them on existing algorithms and methodologies, discuss implementation details, and give evaluation results, which finally leads to a platform and model for Smart Multimedia Information Retrieval applications.

The structure of this paper follows the problem-solving methodology of Nunamaker et al [5] and describes the current state of the art in section 2, the theory building, i.e., modeling and design of the proposed solution in section 3, implementation examples in section 4, and the results of the evaluation in section 5. In each section, the problem areas mentioned above are addressed in corresponding subsections. Finally, section 6 summarizes the results.

2. State of the art and related work

In this section, the state of the art and related work for Smart MMIR is summarized. An overall framework and corresponding research is discussed in subsection 2.1. The area of scalability and distributed MMIR processing is outlined in subsection 2.2, and the introduction of human understandable semantic annotations is given in subsection 2.3.

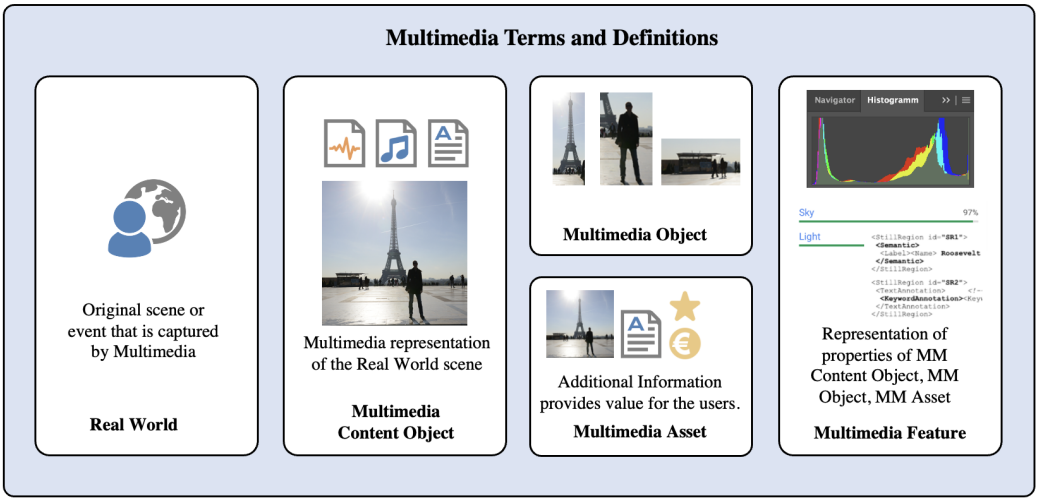


Figure 1. Multimedia Terms and Definitions

In the remainder of this paper we use the following terms to describe various MM relationships and objects (see also Figure 1):

- *Real World Object*: the objects that are captured by some MM recording.
- *MM Content Object*: a MM representation, typically as a MM file of the *Real World* scene or event.

- *MM Object*: an object within a MM Content object, e.g. a detected person or an audio track within a video.
- *MM Asset*: some MM Objects might have a *value* for users or applications, e.g. when a license is attached or when users mark MM Objects as "favourites".
- *MM Feature*: represents the features of *MM Objects*, *MM Assets*, or *MM Content Objects*.

Of course, for each MM Content Object various digital formats exist. A comprehensive overview is given in [6] by the U.S. Library of Congress. For images, these are formats, like PNG, GIF, JPEG, TIFF, RAW, or BMP. For videos, standards like MOV, MPG, MP4, or MXF, exist. Audio objects can be represented by digital formats like, MP3, WAV, AIFF, or MIDI, and textual information can be stored in, e.g., DOCX, TXT, RTF, XML, HTML, or JSON files. All these formats have different purposes, prerequisites, properties, and digital representations of MM features, and many of these formats can be combined to represent multi-media objects, literally. Working with and integrating all these different MM Content Objects is a challenge for MMIR applications.

2.1. Integration area

In our previous and related work [7], we introduced a *Generic Multimedia Analysis Framework (GMAF)*, which provides a flexible plugin architecture for the integration of plugins for the extraction of MM Features of different MM Content Objects (see Figure 2).

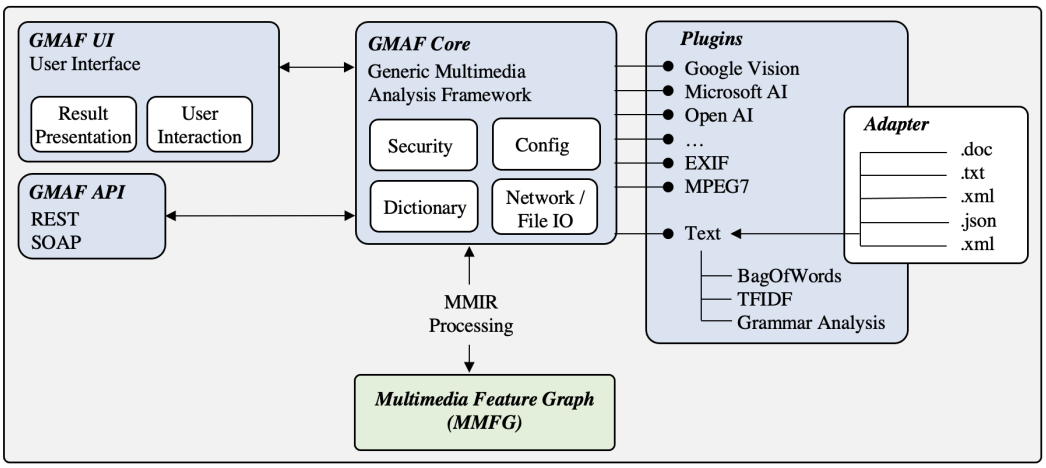


Figure 2. Overview of the Generic Multimedia Analysis Framework (GMAF).

The GMAF provides a flexible, extendable API for the integration of *Plugins*, which encapsulate the extraction of MM Features of a certain MM Content Object type. All *Plugins* contribute the detected MM Features to a generic datastructure, the *Multimedia Feature Graph (MMFG)* [2].

However, there are two remaining challenges: 1) currently, many different plugins are available for the extraction of MM Features. This can lead to contradictions, refinements, or confirmations of detection results. Hence, a mechanism is required for the integration or fusion of MM Features detected by different GMAF plugins. 2) the GMAF is currently based on a static configuration. This means, that all MM Content Objects are processed in a similar way according their content type. However, many applications need a flexible definition of processing instructions. Therefore, a flexible and configurable structure is required to support application based processing flows.

Another important related work is IVIS4BigData [14], where an architecture for the visualization of information is presented, which can also serve as an architectural model to process raw data into structured data, and apply analytic algorithms to it. The corresponding information model in the area of multimedia can be represented by the *stratification model* [8], which forms a (optionally time-based) set of different layers, that segments the contextual data contained in, e.g., a video, into multiple layers called strata. By employing

this, feature information of various Multimedia layers can also be identified for a certain point of time, e.g. within a video. Such a model can contribute to the modeling of processing flows, which is outlined in section 3.

2.2. Scalability area

Due to the increasing level-of-detail of many MM Content Objects, the number of nodes and edges in the corresponding MMFGs increases rapidly. To mitigate this resource constraint, as a first step, the GMAF has been designed to be horizontally scalable, i.e., multiple GMAF nodes can be arranged for distributed processing (see Figure 3). However, many graph based operations have polynomial or even exponential time complexity [9]. As horizontal scaling does not reduce the complexity as such, further optimizations in terms of scalability must be made. Hence, in [7], we introduced the concept of *Graph Codes*.

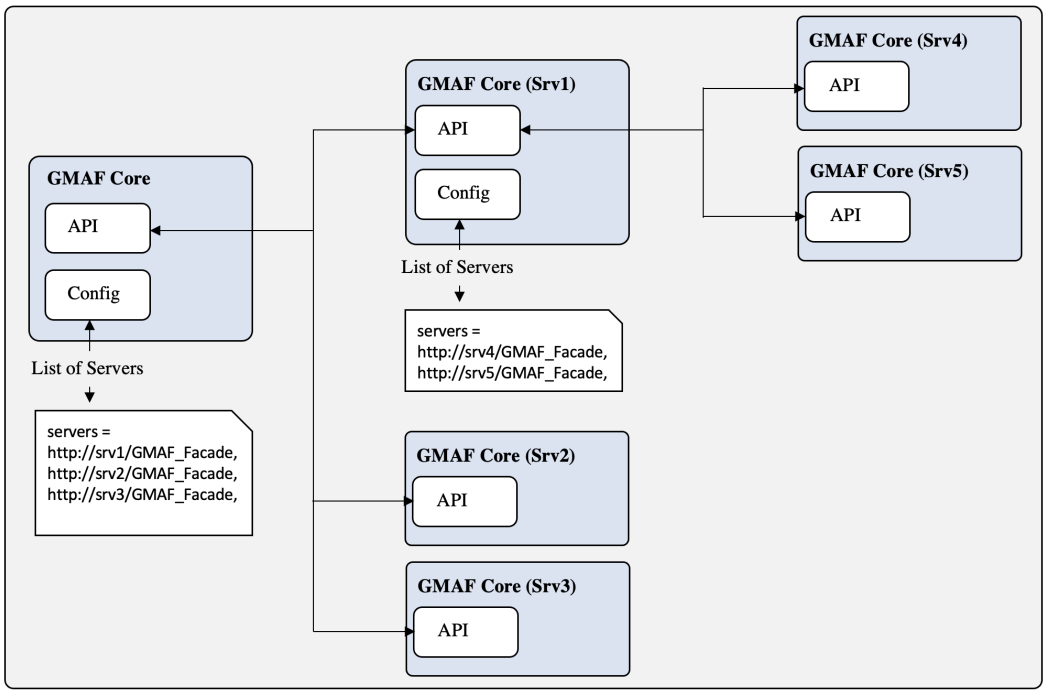


Figure 3. Distributed processing in the GMAF.

Graph Codes [10] are a 2D projection of a multimedia feature graph on which a set of metrics can be applied. The mathematical background has been outlined in [2] and it has been shown, that *Graph Codes* are very efficient for the calculation of similarity and other MMIR tasks. Figure 4 summarizes the most important concepts and shows a feature graph (4a and 4b), the corresponding adjacency matrix (4c) and the *Graph Code* (4d). Furthermore, a screenshot of the GMAF application showing a *Graph Code* is given in Figure 4e and 4f.

In the area of *Graph Codes* several definitions have been made [2], which are relevant for the modeling presented here. Therefore, in the following section, a short summary is given providing the formal background.

- matrix fields of the *Graph Codes* are denoted by $m_{i,j}$.
- the row and column descriptions are called feature vocabulary terms fvt and represented by the set FVT and also called the dictionary $dict_{GC}$ of a *Graph Code*
- the metric $M_{GC} = (M_F, M_{FR}, M_{RT})$ is a metric triple representing the similarity of *Graph Codes* on various levels
- M_F is the *feature-metric* and is based on fvt and defined as $M_F(GC_i, GC_j) = \frac{|dict_\cap|}{|dict_i|}$
- M_{FR} is the *feature-relationship-metric* and represents all possible relationships. It is defined as $M_{FR}(GC_i, GC_j) = \frac{\sum AM(M_{\cap i,j}) - n}{|AM(M_{\cap i})| - n}$, where AM is the adjacency matrix of the corresponding graph. M_{FR} represents the ratio between the number of non-zero

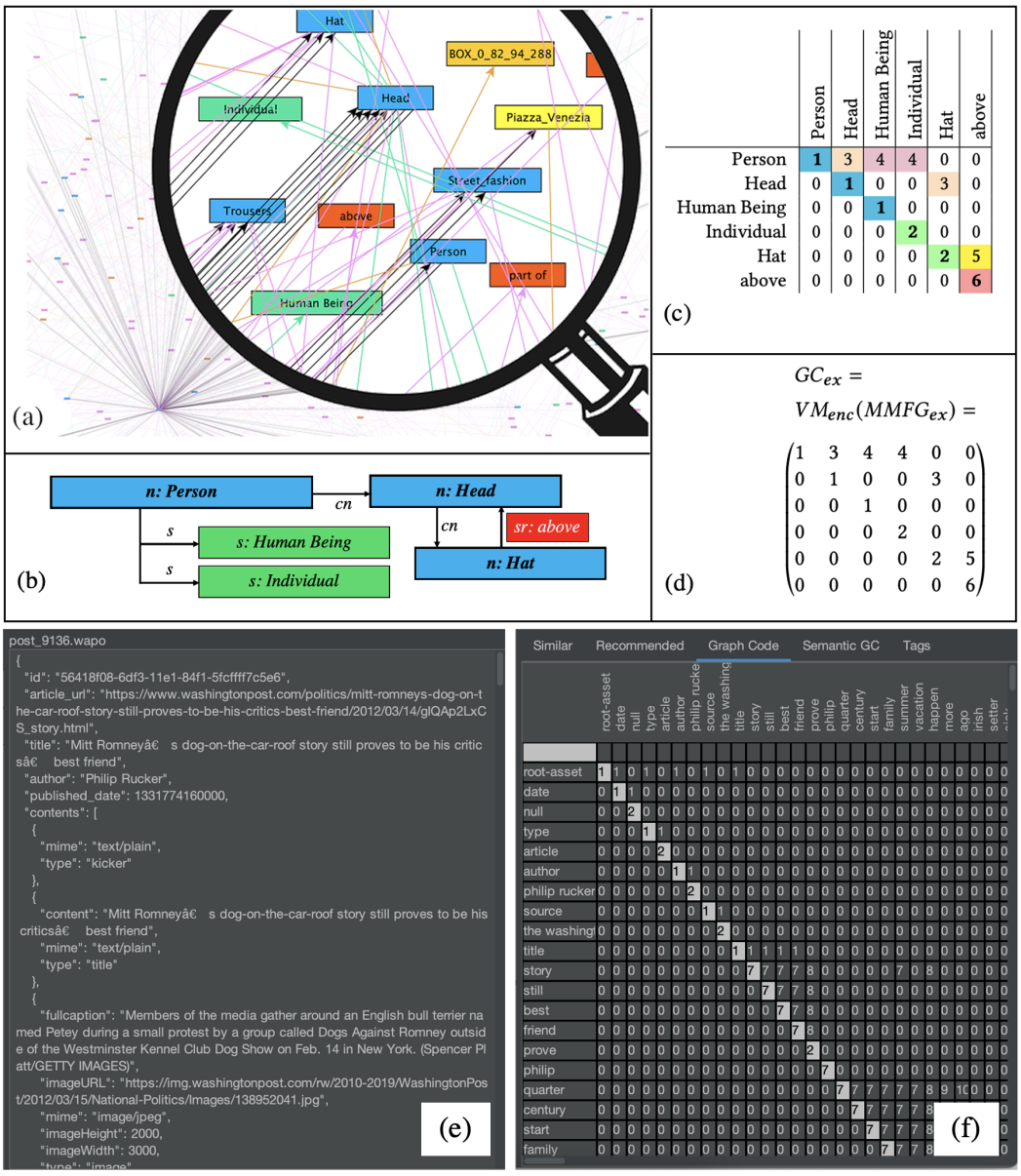


Figure 4. Mutimedia Features represented as a *Graph Code* index (a-d), example of a *Graph Code* index and its matrix visualisation for a text document (e, f).

edge-representing matrix fields and the overall number of equivalent and intersecting edge-representing matrix fields of, e.g., two *Graph Codes*.

- M_{RT} is the *relationship-type-metric* calculating similar (and not just possible) relationships as $M_{RT}(GC_i, GC_j) = \frac{\sum_{i,j}^{n,i\neq j}(|M_{\cap i} - M_{\cap j}|)}{|M_{\cap i}| - n}$

In [2] we outlined an algorithm based on these metrics for the parallel processing of *Graph Code* operations. This algorithm has been implemented in Java, Objective-C (for Apple devices), and CUDA (for NVIDIA devices) and proves, that the parallelization of *Graph Code* operations scales linear instead of polynomial or exponential time for the corresponding graph-based operations on MMFGs. Experiments [2] show, that the theoretical speedup of these operations only depends on the number of available parallel processing units and also prove linear time complexity. For the exemplary collections employed in [2], a speedup of factor 4.000 was measured. Combined with the already presented solution for horizontal scaling, this is an unseen opportunity for MMIR processing of high volume and high level-of-detail collections.

In [3], *Semantic Graph Codes (SGC)* have been defined containing semantic annotations with systems like RDF, RDFS, ontologies, or Knowledge Organisation Systems [12][13][15] and thus bridges the gap between the technical representation of MMIR features and its human understandable meaning.

The distributed processing of GMAF instances can be regarded as horizontal scaling, and the GPU-based optimizations in parallel *Graph Code* processing can be regarded as vertical scaling. However, the current architecture of the GMAF is based on a static configuration for either vertical or horizontal scaling. As shown in related work [18][19][20], various algorithms are in place to support automated and/or application based scaling of processes or processing steps. However, to support the integration of both vertical and horizontal scaling, the corresponding configuration, and also the employment of autoscaling algorithms, several prerequisites must be met, which are currently not part of the GMAF. Hence, further modeling and extensions of the framework are required. This will also affect several *Graph Code* based optimizations for further compression and relevance calculations.

2.3. Explainability area

To explain representation or indexing structures in MMIR, extensions the both MMFGs and *Graph Codes* have been made [3], which employ a formal *PS-Grammar* [17], which takes annotations of the MMFG or *Graph Code* to create sentences in a human understandable way. According to [16], a grammar $G = (V, T, P, S)$ for a language L is defined by the tuple of vocabulary terms V, the list of terminal symbols T, which terminate valid sentences of L, production rules P, which describe valid combinations of non-terminal symbols and a set of starting symbols S for sentences of L. In [17], *PS-Grammars* are employed as a specialized form to generate language terms by production rules, in which the left side of the rule is replaced by the right side. If, e.g., $\alpha \rightarrow \beta$ is a production rule in P, and ϕ, ρ are literals in V, then $\phi\alpha\rho \rightarrow \phi\beta\rho$ is a direct replacement.

Particularly, when defining grammars, the set V will contain additional classes to structure the possible production rules (typically defined as Chomsky rules [16]), e.g. classes to describe *Nominal Phrases (NP), Verbal Phrases (VP), Prepositional Phrases (PP)*, or other word types like *Adjectives (ADJ)*, and their location in validly produced sentences [17]. In many cases, grammars are designed, that $V \cap T = \emptyset$. As an example, the sentence, "The hat is above the head", can be represented by the context-free grammar $G_{en} = (V_{en}, T_{en}, P_{en}, S_{en})$ for simple english sentences:

- $V_{en} = \{S_{en}, NP, VP, V, N, DET, PR\}$ represents the variables (or non-terminal symbols) of the grammar
- $T_{en} = \{the, hat, is, above, head\}$ is the set of terminal symbols
- P_{en} is the set of production rules for this grammar and can be defined as follows:

$$P_{en} = \{S_{en} \rightarrow NP\ VP, VP \rightarrow V\ PP, NP \rightarrow DET\ N, PP \rightarrow PR\ NP\} \quad (1)$$

In [3] we further showed, that not only feature graphs but also the indexing structures like, e.g. *Graph Codes*, can be automatically transformed into human understandable texts. Based on this, further metrics for *Semantic Graph Codes* have been introduced [4] as follows:

- M_{DIS} is the *feature-discrimination-metric* describing the discriminative power of a feature vocabulary term as $M_{DIS}(fvt_i, fvt_j) = \sum_{k=0}^{n} |m(i,k)| - \sum_{k=0}^{n} |m(j,k)|$
- the TFIDF measure [21] has been adapted, as well to statistically improve the relevance of MMIR features: $\forall vt_i \in SGC, \forall vt_j \in SGC_{Coll} : TFIDF(vt_i, SGC) = M_{DIS}(vt_i, vt_j) \cdot log\frac{|SGC_{Coll}|}{M_{DIS}(vt_i, vt_j)}$
- M_{REL} can be defined as the *feature-relevance-metric* representing the difference of the TFIDF-measure of two feature vocabulary terms: $M_{REL}(vt_i, vt_j) = TFIDF(vt_i, SGC_{Coll}) - TFIDF(vt_j, SGC_{Coll})$

- the introduction of collection wide stop words SGC_{STOP} leads to further refinement of the feature vocabulary terms.
- finally, and with high relevance for this paper, M_{ABT} has been defined as the *aboutness-metric* for a collection as $M_{ABT} = \bigcup SGC_{Coll} - SGC_{STOP}$

In [4] we demonstrated, that based on these metrics, *Feature Relevant Graph Codes (FRGC)* can be calculated by measuring the distance of a *SGC* employing M_{ABT}. This facilitates calculation of *Explainable SGC* for answering typical MMIR questions:

- "why is this element in the result list?": $ESGC = FRSGC_{Element} - FRSGC_{Query}$
- "why is element A before element B?": $ESGC = FRSGC_A - FRSGC_B$
- "what is element A?": $ESGC_A = ESGC(FRSGC_A)$

On this basis, already human understandable explanations of MMIR processing steps can be calculated. However, the open challenges in the area of integration and scalability also affect the area of explainability and expressiveness. As solutions for these remaining challenges might involve additional MMIR processing facilities, the resulting MMIR process steps become more difficult. Hence, the explanation of such processing steps has also to be validated and/or enhanced.

2.4. Related work

In the area of *Integration, Fusion*, or *Enrichment* of MMIR features [22] several algorithms and techniques have been proposed. Related work also aims at solutions for individual multimedia object types. Exemplary here, *An effective content-based image retrieval technique for image visuals representation based on the bag-of-visual-words model* [23], will be outlined briefly. In their paper, the authors discuss image feature fusion based on two very common feature detection algorithms: the *SURF* (Speeded-Up Robust Feature) and the *FREAK* (Fast Retina Keypoint) algorithms. They train a machine learning model for each algorithm and fuse the detected features according to the Bag-Of-Visual-Words model by applying both SURF and FREAK algorithms.

In *Learning Specific and General Realm Feature Representations for Image Fusion* [24] another approach for the fusion of multimedia features is presented. Input image are represented in various transformed formats, each image sis processed with specific feature detection algorithms, and finally the detected features are fused iinto a single model. The authors show, that the fusion of images increases the MMIR results.

Instead of fusing features from representations of the same image, the fusion of features from images and texts has also been a focus of research. Particularly in the area of social media, this combination can lead to an increase of effectiveness in retrieval. In *Object-Aware Multimodal Named Entity Recognition in Social Media Posts With Adversarial Learning* [25], the authors introduce an approach, that feeds features from text named entities [26] and detected image features into a machine learning network. This work provides strong evidence, that the fusion of various MMIR features from different sources increases the overall effectiveness by 3-8% depending on the underlying problem domain.

As a last candidate, the paper *Discovering Multirelational Structure in Social Media Streams* [27] should be mentioned. The authors of this paper highlight that information relevant for topic clusters (e.g., social, travel, project, etc.) is enriched during the time and thus more and more refines the existing information (i.e., feature) basis. This clustering also increases the semantic information of detected features, as ambiguous content can be easier aligned with topic domains.

Further articles related to this work are *Learning rich semantics from news video archives by style analysis* [28], where news-videos in particular are semantically enriched according to production elements (e.g., weather icons, tickers), or *Beyond search: Event-driven summarization for web videos* [29], which illustrates an automated shot-detection and overview for web videos, or *Semantics and feature discovery via confidence-based ensemble* [30], where machine learning approaches are also employed for the detection of features with a focus on semantics. In *Temporal Event Clustering for Digital Photo Collections* [31], another approach

similar to the timeline enrichment for images is presented, while *Content And Concept Indexing For High- Dimensional multimedia Data* [32] introduces additional dimensions (e.g., time, topic) similar to [27]. All this related work basically shows that the fusion of MMIR features provides a potentially significant benefit for retrieval. However, they provide no general or unifying solution or framework for the fusion of any multimedia object type in general and they all utilize existing algorithms and hence are stuck with the existing level-of-detail.

2.5. Summary and remaining challenges

In this section, we outlined various components that already address general MMIR topics in the problem areas. In the integration area, the GMAF framework provides facilities and existing MM Feature extraction mechanisms to integrate MM Content Objects of different types and to store their MM Features in a MMFG. In the area of scalability, *Graph Codes* as a 2D transformation of MMFGs show significant speedup due to parallelization and are employed to formally model a set of metrics, which can be also applied in the area of explainability to introduce semantics and human understandable text generation to *Graph Codes* and MMFGs. However, to fulfill the definitions of "Smart MMIR", some challenges remain open:

- in the integration area, the GMAF provides a good and flexible solution. However, this solution is currently quite static and the processing of each MM Content Object is done individually. Although collection-based metrics are available, there is no harmonizing or integrating mechanism between various MM Feature extracting plugins. A more intelligent and semantic approach is required.
- in the area of scalability, significant achievements have been made. However, for real-world applications, a flexible approach for the combination of both horizontal and vertical scaling is required to intelligently support different application types.
- the current human understandable representation of MMFGs and *Graph Codes* in the area of explainability is text based. However, in multimedia applications, other visualization techniques must be employed, particularly, as both MMFGs and *Graph Codes* can become extensive.

A solution for these open challenges is now given in the next section.

3. Modeling and design

As outlined in section 1, Smart MMIR is interoperable, scalable, expressive, human understandable and explainable. In this section, we introduce two new concepts, which contribute to Smart MMIR: the *Soundness*, which is a descrete parameter describing the consistency of an information set, and *Processing Flows*, which are a means for scaling, distributing, and integrating Smart MMIR with other applications. This section contains five subsections and a summary. First, the *Soundness* is introduced in 3.1, then the concept of *Processing Flows* is described in 3.2. The following three subsections are employed to model the effects of *Soundnes* and *Processing Flows* in the area of integration (3.2), scalability (3.3), and explainability (3.4).

The modeling here follows the *User Centered System Design* approach by Norman & Draper [33], which places the user in the center of conceptual modeling. This further means, that the presented solution directly generates a benefit for users of the application. In the context of this paper, this means, that the starting point for the modeling is an Use Case and thus a typical scenario, users may be confronted with. The approach further implies, that such a scenario can be selected as a test case within a cognitive walkthrough experiment, where users work with the application and the results are measured. Finally, this approach guarantees, that any modeling produces a benefit for the users of an application.

3.1. Soundness

In many MMIR applications, it is important to decide, if the information extracted from a certain MMIR object, is *sound*. This means, it is consistent, fits together, robust,

describes the MMIR object correctly, and thus also indicates the quality of information. Figure 5 shows two exemplary social media posts, where a description and an image are employed to provide some information to users. If the description fits to the image, such a post can be regarded as being *sound*. If not, the soundness indicates some mismatch or contradiction in the information. However, for users it may be hard to distinguish between sound and not sound MIR assets.

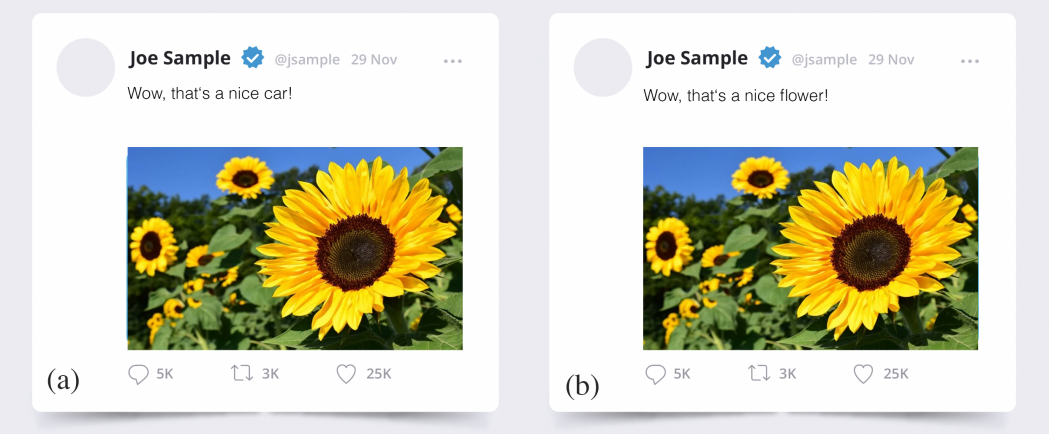


Figure 5. Exemplary social media post with information, that is not *sound* (a) and *sound* (b).

For such a scenario, we introduce the *Soundness* M_{SND} as a discrete value, that can be calculated based on *FRGCs* and the *Graph Code* metrics M_{RT} and M_{FR} as the fraction of similar relationship types and possible relationship types between given feature vocabulary terms. For the calculation of M_{SND}, Feature Relevant Semantic Graph Codes are employed, as they already represent standardized semantic identifiers, and only contain the relevant features for an application. Furthermore, for the calculation of M_{SND}, only the intersecting paramters of MMIR assets are used. If no common elements are in two *FRSGCs*, a calculation of M_{SND} is not possible.

$$M_{SND}(FRSGC_1, FRSGC_2) = \frac{|M_{RT}(FRSGC_1, FRSGC_2)|}{|M_{FR}(FRSGC_1, FRSGC_2)|}, \tag{2}$$

For the above example, this means, that the images FRSGC would contain the vocabulary term "flower", while the textual description either contains this vocabulary term, or doesn't. Therefore, M_{FR} would have the value 1 for one common relationship, while M_{RT} would have either value 1 or 0. Of course, real examples not only contain single values, and hence, in section 5 (evaluation), further examples are given. It may be noted, that a typical MMFG can contain tens of thousands of nodes and even more relationships. The calculation of *FRSGCs* compresses this information but still leaves an average of 500 vocabulary terms for a typical element of a MMIR collection. This means, that the M_{SND} will provide a fine grained classifier for a MMIR asset. It is important to highlight that *FRSGCs* are still explainable and that the grammar introduced in section 2.3 are still applicable. However, due to the compression of *FRSGCs*, now shorter and much more precise information can be presented to the users.

The introduction of *Soundness* is particulary relevant for MMIR assets, that consist of multiple individual assets, like documents with e.g., embedded pictures, social media posts with images, videos, texts, comments, likes, medical information with MRT images and doctor's letters, or various connected information of the same multimedia scene on any other application area. Because, in such a setup, the individual elements that contribute to the information of the combined MMIR asset can contradict or confirm each other and thus produce a higher value for M_{SND}. But also, when applications deal with individual MMIR assets, M_{SND} can be an important metric. In previous work, we already introduced the

Aboutness M_{ABT}, which describes a common knowledge of a MMIR collection by calculating the most relevant feature vocabulary terms and the most common relations between them. If, e.g., a medical application collects values for blood pressure, M_{ABT} would represent the typical range of such values. If a new asset is added to the collection, M_{SND} can be calculated based on M_{ABT} and such indicate the deviation of a certain value from the current state of knowledge within the collection. This leads to numerous application scenarios. Finally, if the definition of truth within an application is given, e.g., because information about laws or scientifically approved texts is fed to a MMIR system, M_{SND} indicates, if a MMIR asset complies to this set of true information.

M_{SND} can be represented as a discrete value. This means, that based on this value, thresholds and pre-defined decisions can be introduced. For example, if M_{SND} of a social media post is lower than 0.5, the post can be regarded as fake news. Such decisions can lead to a more flexible way of processing MMIR information. However, to define such processing flows, some further extensions to existing MMIR solutions must be made. This is outlined in the next subsection based on the Generic Multimedia Analysis Framework (GMAF).

3.2. Processing Flows, Integration area

As shown in section 2.1, the GMAF already contains a structure to attach plugins for the extraction of MM Features. It has also been shown, that various plugins exist, that can contribute features to the same MM Content Object type. For example, if an image is processed by different object detection algorithms, each of these algorithms might detect different or similar objects. However, if, e.g., an algorithm is optimized for the detection of fruit, a tennis ball might be considered as being an orange. If an algorithm is trained for the detection of cars, the MM Feature term "Jaguar" might have a different meaning than the "Jaguar" detected by an algorithm optimized for animals. Experiments in related work [7] show, that depending on the employed MM Feature extraction algorithms, contradictions can exist.

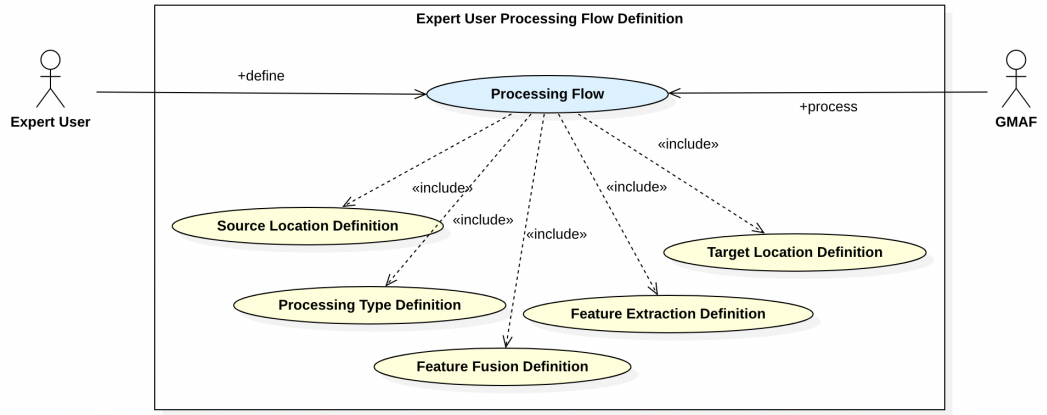


Figure 6. Expert use case for feature fusion and processing flow configuration.

This kind of integration has to be defined by an additional user type, an expert user. Hence, following the User Centered System Design approach, an additional use case is introduced (see Figure 6). This use case describes the expert tasks for the definition of *Processing Flows*. These tasks are typically performed in a preparatory step. It must be noted, that also this preparatory steps directly influcence MMIR processing steps and have also to remain explainable.

Contradictions, as well as confirmations should not occur occasionally, but in a planned and user-definable way. Users typically want to construct processing flows and define, how the results of various processing plugins should be combined (see examples in Figure 7).

In addition to already existing plugins, two components are introduced: (1) a *Feature Fusion* facility and (2) the general concept of *Processing Flows*. Feature fusion is based on

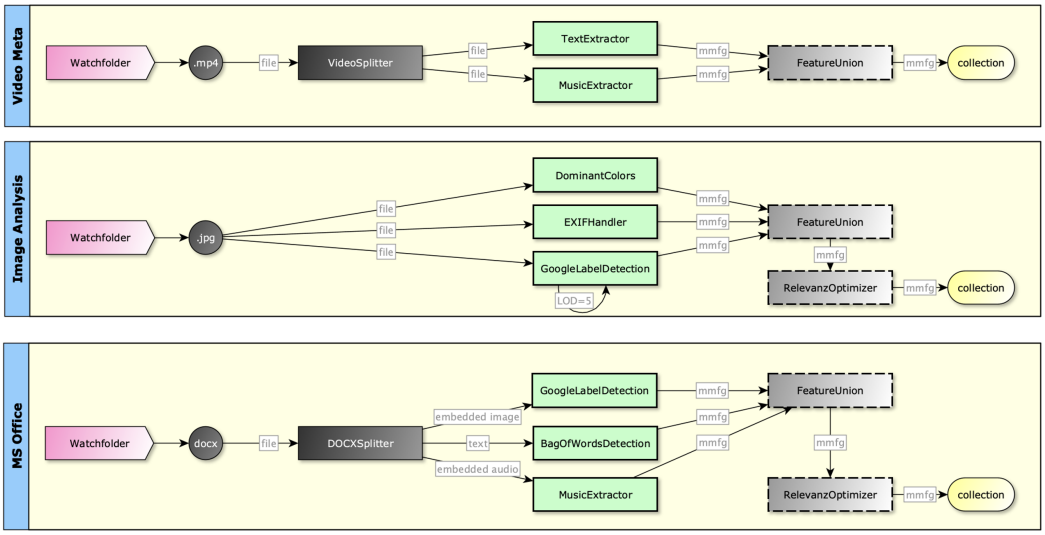


Figure 7. User definable processing flows.

MMFGs and takes one or more MMFGs as an input. The result of such a Feature fusion is a single MMFG, which contains combined or optimized elements. The decision, which elements are moved from the source MMFGs to the resulting MMFG, which elements are deleted, re-weighted, renamed, or even added, is subject to a Feature fusion strategy. According to the open design and architecture of the GMAF, also these strategies should be exchangable and interoperable. Figure 8 shows these newly introduced building blocks in the GMAF architecture.

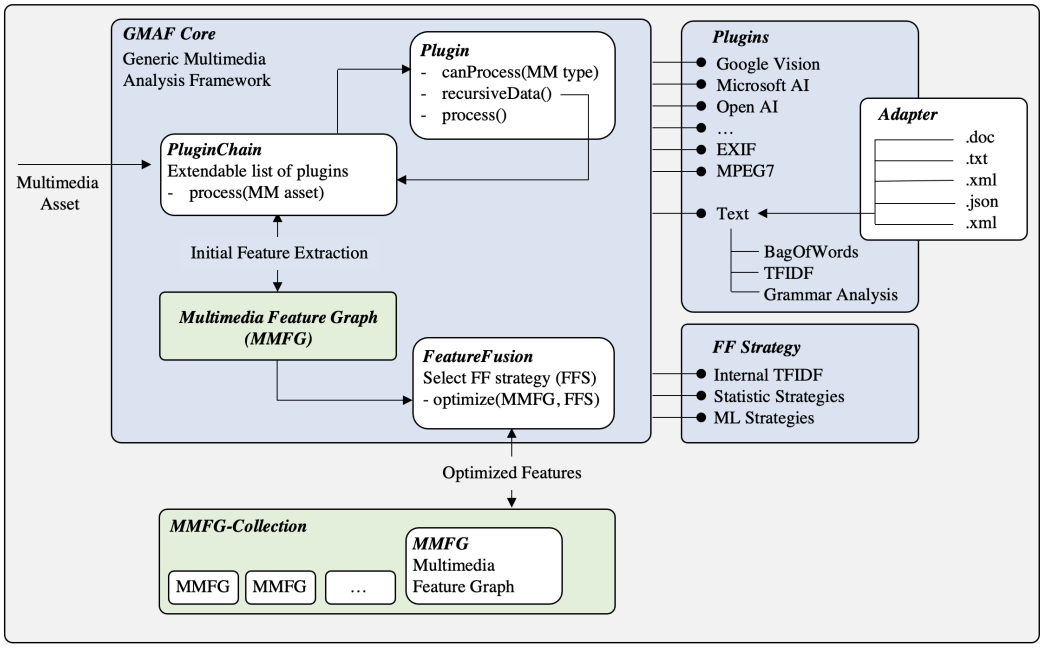


Figure 8. Feature Fusion and Plugin Chain facilities in the GMAF.

Formally speaking, a feature fusion can be denoted as a function

$$ff(MMFG_1, MMFG_2, ..., MMFG_x) \rightarrow MMFG_{Result} \quad (3)$$

which activates a node-based function f_{opt}, based on the set of nodes of all MMFGs N in a collection with x elements to calculate the resulting (i.e., fused) set of MMFG nodes M based on its node's properties.

$$N = n_j \in MMFG_i \Leftrightarrow i < x \tag{4}$$

$$f_{opt}(N) \rightarrow M \tag{5}$$

This means, that for all nodes of the input MMFGs, f_{opt} produces output nodes for the resulting MMFG. Of course, f_{opt} is the function, where algorithmic optimizations, like reasoning, inferencing, fusion, unioning, and weighting, are represented.

Furthermore, a *Plugin Chain* element is introduced (see also Figure 8), which is able to construct a list of processing plugins, feature fusion elements, and any combination of these to support GMAF processing in terms of the above mentioned *Processing Flows*.

From a design perspective, *Processing Flows* are an adaptation of the Multimedia Stratification Model [8], as each *Processing Flow* can be regarded as a representation of a particular MM Content Type. Following this model, the layering of *Processing Flows* can be particularly relevant, when content is real "multi"-media, e.g., embedded audio, video, image objects in other multimedia objects. Formally, and according to Figure 6, such a *Processing Flow PF* can be constructed by a *Source Location Definition SLD*, a *Processing Type Definition PTD*, several *Feature Extraction Definitions FED*, a number of *Feature Fusion Definitions FFD*, and a *Target Location Definition TLD*:

$$PF = \{SLD, PTD, FED*, FFD*, TLD\} \tag{6}$$

The introduced GMAF *Plugin Chain* element is designed to accept such *Processing Flow* definitions and thus provides further flexibility and interoperability, as well as smarter application profiling in the area of integration.

3.3. Scalability area

As already shown in section 2.2, *Feature Relevant Graph Codes* represent a compressed form of *Graph Codes*, based on their relevance within the overall collection. Compression is very important for *Graph Code* processing, as it leads to even better processing times due to fewer available vocabulary terms. Also, the above introduced *Feature Fusion strategies* can lead to a compression of the underlying MMFG. However, there is one important difference: Feature fusion determines, what is "right", while FRGC represent, what is "relevant" based on the collection's content. Both mechanisms require re-processing, when new content is added to a collection. Unfortunately, such re-processing of a collection may be very expensive, as any existing MMFG and any already calculated and optimized FRGC may have to be re-calculated.

A simple example illustrates this: as shown in previous work [3], the GMAF is able to detect new MM Features by comparing a new MM asset to similar assets with older timestamps. In medical applications, this can be employed to detect deviations, tumors, or general changes in a patient's medical data. This can also be employed, to detect the "new watch", a user is wearing on a photo, that has been added recently to the collection. If this is detected, the MM Feature "new watch" is added to the corresponding MMFG and *Graph Code*. However, at some point of time, this "new watch" might become an "old watch" and be replaced by another "new watch". When this happens, the whole collection (or at least the part of the collection containing the "new watch" *Graph Codes*) needs to be re-processed.

The same applies to the general calculation of FRSGCs, as the underlying TFIDF algorithm employs thresholds to determine, which features are relevant or irrelevant for a collection. If, e.g., we have a collection of thousands of football pictures, a single picture with a tennis ball might be considered as being irrelevant within the collection. However, if users upload millions of tennis pictures, the relevance of the football ones might decrease

and the irrelevant first tennis ball might gain relevance instead. Also here, re-processing is required.

Furthermore, it has to be considered, that the GMAF processing is typically distributed both horizontally and vertically. Vertical distribution is responsible for parallelization employing GPU processing, horizontal distribution can be employed to distribute the collection based on MM Content Object type or processing facilities. For example, all videos could be stored at a GMAF node, where specialized video decoding hardware is located. However, in any case, such distributed collections and processing needs to be reflected also in the *Feature Relevance Metric* M_{REL} as each individual node needs the information of the overall collection's M_{REL} to calculate *FRGCs* and thus, to process MMIR including explainability.

Hence, in the following, the calculation of M_{REL} is modified. Assuming, that the overall collection of *Semantic Graph Codes* SGC_{Coll} is distributed among n GMAF nodes, each of these nodes has its own, individual subset of SGC_{Coll}:

$$\forall k \in n : SGC_{Coll} = \bigcup_{k}^{n} SGC_{Coll_k} \tag{7}$$

To indicate, on which nodes a re-processing is required, M_{REL} is calculated both on the feature vocabulary terms of SGC_{Coll} and SGC_{Coll_k}. This means, that a node's individual collection's relevance is compared to the overall collection's relevance. If MM Assets are added to the collection that are similar to the existing ones, neither the individual, nor the overall M_{REL} is going to change. If different MM Assets are added to a distinct GMAF node, this might - of course - affect this single node, but not automatically all other nodes of the collection. The reprocessing indicator RI for a particular GMAF node k can thus be defined as:

$$\begin{aligned} &\forall vt_i, vt_j \in SGC_{Coll}, \\ &\forall vt_m, vt_n \in SGC_{Coll_k} : \end{aligned}$$

$$vt_i = vt_m \wedge vt_j = vt_n \Rightarrow RI_k = M_{REL}(vt_i, vt_j) - M_{REL}(vt_m, vt_n) \tag{8}$$

If RI_k is greater than zero (or a certain threshold), the GMAF node k needs re-processing. Otherwise, its relevance values are still valid. A further result of these modification, re-processing will also affect M_{ABT}, which is based on M_{REL}. This means, that the topic area of a collection can automatically change from time to time. As Explainable Graph Codes are based on FRGCs, the results of the calculation of human understandable texts will also change, when M_{ABT}, and M_{REL} change automatically.

Furthermore, it must be noted, that on this basis, the calculation of M_{SND} can also be completed in an efficient manner, as all prerequisites for this calculation can be fulfilled in advance. Once, e.g., M_{ABT} is calculated for a collection, for each further element M_{SND} can be calculated in a single step. Based on the introduced processing flow, also specialized hardware can be employed for the calculation of, e.g., *Graph Codes* by parallel processing, and hence improve the overall application performance. Hence, this modification leads to smarter MMIR processing and scalability.

3.4. Explainability area

Until now, human understandable texts are calculated for the explanation of MMIR processing steps and results. However, written text in many cases lacks expressiveness. The adage "a picture is worth a thousand words", is a good example that visual expression is regarded to be more appropriate in particular areas and, for sure, in the area of multimedia. Hence, further visualizations of ESGCs, ESMMFGs, and the corresponding

calculations of typical MMIR questions (see section 2.3) are required. As an example for such a visualization, a wireframe of a smart query refinement user interface is shown in Figure 9.

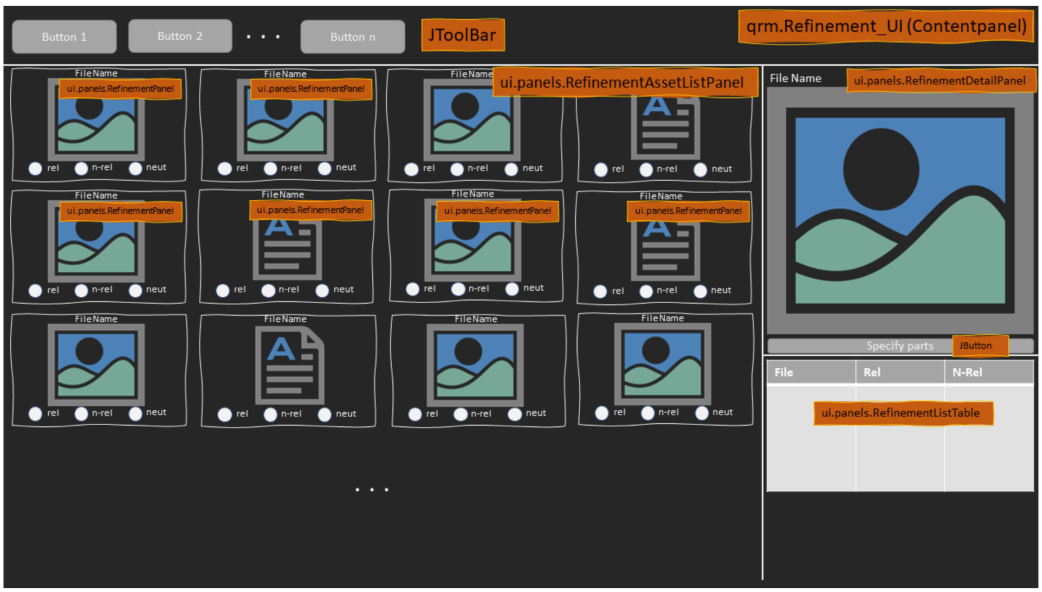


Figure 9. Visualization of Query Refinement based on Relevance Feedback.

Furthermore, the just introduced enhancements in the area of integration and scalability also affect explainability. For example, the definition of *Soundness*, *Processing Flows* and *Feature Fusion* produces important information, that potentially need to be explained to users. The *Soundness*, e.g., provides relevant information about the correctness or integrity of a certain MMIR asset. If users, e.g., upload an additional element to their MMIR collection, deviations can be detected automatically and a detailed explanation why this element deviates from another or from the rest of the collection, can be presented to the users. Depending on the definition of processing flows, the MMIR results can be completely different through applications, which may lead to confusion when the same MMIR Objects are viewed by users in different applications. Hence, these steps also have to be included in the expressiveness and explainability of Smart MMIR. However, this topic will remain the subject of future work, as in the context of this paper the important foundation for this research is introduced in the other research areas.

Basically, the introduced concepts already provide a solid foundation for the modeling of further UI elements to visualize expressiveness. However, such a refinement and feedback function should be available for any MM Content Object type. This means, that query refinement has to be available for image-based queries, text-based queries, audio-based queries, video-based queries, and mixed multimedia-based queries. Hence, in our modeling, we also employ a generic architecture here, which supports these use cases in a general way (see section 4).

3.5. Summary

In this section, we introduced a number of extensions and refinements of the existing state of the art to make existing MMIR smarter. Particularly, in the area of integration, the definition of Soundness, Feature Fusion strategies and Processing Flows empower applications to utilize smarter workflows and a semantically correct calculation of MM Features. The adaptation of the *Graph Code* metrics M_{ABT} and M_{REL} for distributed and heterogeneous collections including the calculation of a reprocessing indicator, supports highly efficient scaling of MMIR processing. Finally, we have given an example of a more expressive visualization of MMIR processes in the area of explainability. All these points contribute to Smart MMIR.

To show and prove, that the modeling here can be implemented, in the next section a brief overview of our prototypical Proof-Of-Concept (POC) implementation is given.

4. Implementation

In this section, a short overview of selected components of the POC implementation is presented. The full implementation of the GMAF and the corresponding concepts, including those presented in this paper, is available at Github [34]. In this section, for each problem area, one selected implementation example is given. Subsection 4.1 contains information about the integration area presenting a feature fusion plugin, subsection 4.2 shows the distribution of collections in the area of scalability, and subsection 4.3 demonstrates the implementation of visual query refinement and relevance feedback.

4.1. Integration area

In the implementation area, we introduce a new structure in the GMAF, the *Feature Fusion Strategy*. A corresponding Java interface has been added to the framework as shown in Listing 1:

```
public interface FeatureFusionStrategy {
        public void optimize(MMFG mmfg, Vector<MMFG> collection);
}
```

Listing 1: The introduced interface for Feature Fusion Strategies

Based on this interface, various strategies have been implemented. To outline the simplicity, whith which new strategies can be added to this structure, Listing 2 shows an example for a *UnionFeatureFusion*, which calculates the union of a given set of MMFGs according to the above mentioned structure.

```
public class UnionFeatureFusion implements FeatureFusionStrategy {
        public void optimize(MMFG mmfg, Vector<MMFG> collection) {
                for (MMFG m : collection) {
                        for (Node n : m.getNodes()) {
                                if (mmfg.getNodesByTerm(n.getName()) != null) {
                                        mmfg.addNode(n);
                                }
                        }
                }
        }
}
```

Listing 2: The union feature fusion strategy

Feature fusion is made a core component of the GMAF processing, which now also has been extended to provide *Processing Flows*. These can be represented by an XML file, which is passed to the GMAF processing of a distinct MM Content Object. An exemplary description of such a processing flow in XML is shown in Listing 3.

```
<process-flow name="ImageImport" extension="*.jpg" isGeneral="false">
        <plugin-definition name="plugin1" class="de.swa.img.google.GoogleVision"/>
        <plugin-definition name="plugin2" class="de.swa.img.yolo.FruitDetector"/>
        <plugin-definition name="plugin3" class="de.swa.img.amazon.FaceDetection"/>

        <fusion-definition name="merge1" class="de.swa.feature.UnionFeatureFusion"/>
        <fusion-definition name="merge2" class="de.swa.feature.RelevanceOptimizer"/>

        <export-definition name="mpeg7" class="de.swa.exporter.Mpeg7Converter"/>
        <export-definition name="xml" class="de.swa.exporter.XMLFlattener"/>
        <export-definition name="graphml" class="de.swa.exporter.GraphMLFlattener"/>

        <resource-definition name="upload-dir" type="folder" location="temp/upload"/>
        <resource-definition name="target-dir" type="folder" location="temp/target"/>
        <resource-definition name="export-dir" type="folder" location="temp/export"/>
        <resource-definition name="facebook" type="url" location="http://www...."/>

        <param name="plugin1.lod" value="2"/>
        <param name="plugin2.output" value="temp"/>

        <flow-source name="upload-dir"/>
        <mmfg processor="plugin1, plugin2, plugin3"/>
        <fusion processor="merge1"/>

```

```
25          <export target="export-dir" exporter="mpeg7"/>
26          <export target="collection"/>
27  </process-flow>
```

Listing 3: Definition of a processing flow

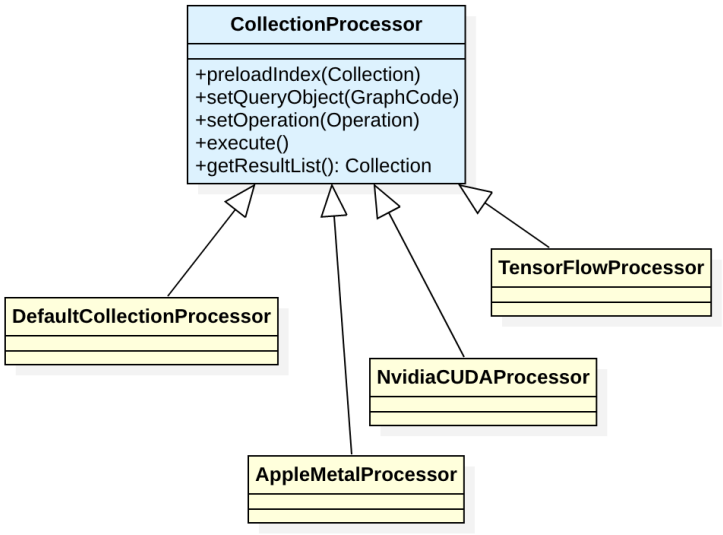


Figure 10. Collection Processor structure for the distribution of collections and processing.

In lines 2-11, the definition of the required resources for the described processing flow are given. For example, in line 2, a *GoogleVision* plugin is defined, which internally follows the GMAF plugin structure and is made accessible within the processing flow by the name *plugin1*. Resource definitions in lines 13-16 can be employed to describe infrastructure settings. Each of the processing components can receive additional parameters (see line 18, 19), which are then passed via Java Reflection to the specified component. Finally, in Lines 21-26, the actual processing flow is defined by a sequential list of actions. In this case, the flow looks for new images in the *upload-dir* folder, processes these with *plugin1, plugin2*, and *plugin3* and applies a feature fusion with *merge1* before finally exporting the result in the *mpeg7* format to the collection.

4.2. Scalability area

In the area of scalability, the structure of the GMAF has been extended to fully support distributed processing. The component responsible for this, is a *CollectionProcessor*, which represents both horizontal and vertical distribution (see Figure 10).

With these introduced structures, also the overall setup of GMAF installations has to be changed. As collections can now be distributed, each collection needs to have one (or more) master-nodes, which represent the knowledge about the distributed components. Hence, when logging on to the GMAF, users must specify which master-node they want to connect to.

4.3. Explainability area

Finally, in the area of visualization and explainability, a prototypical implementation of relevance feedback and query refinement has been added to the framework, which allows users to mark sections of MM Objects as being generically relevant or irrelevant. Each such mark internally is processed as a separate *Graph Code* and correspondingly added or subtracted from the query. Figure 11 shows a screenshot of the implemented solution.

In Figure 11 for each result element, a set of checkboxes has been added, which give the users the opportunity to mark a complete asset as being "relevant", "irrelevant" or "neutral" according to the current query. Furthermore, event the subsections of the content of a selected query can be marked by drawing bounding boxes (for images) or highlighting

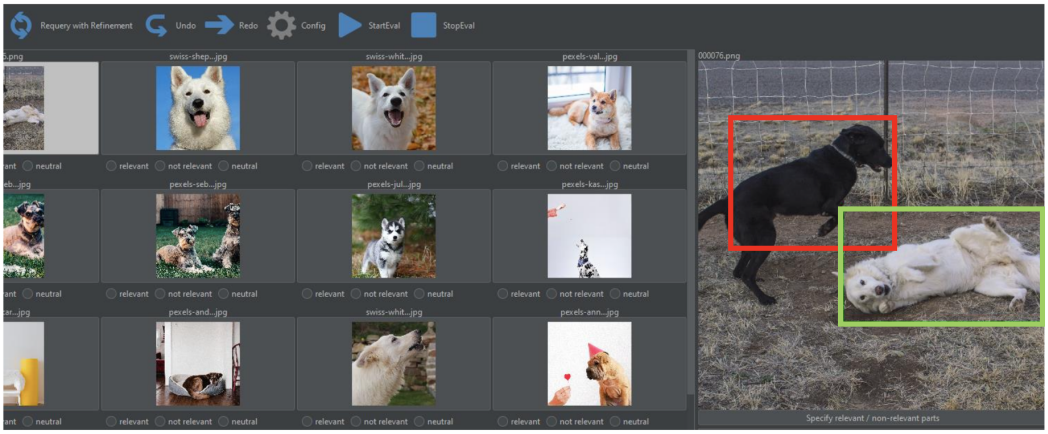


Figure 11. UI for query refinement and relevance feedback.

text with different colors to indicate, which passages or sections are relevant or irrelevant. This highly improves the overall effectiveness of the MMIR process, as users are now able to interactively and visually refine their queries. Further details of this approach are given in section 5.

Furthermore, the expressiveness of the GMAF has been improved by adding complex comparison functions, which explain, why an MM Asset is in a result list, what the difference is between two selected MM Assets, and what MM Features are contained in an MM Object from a MMIR perspective. An example of this is shown in section 5.

For the processing of reasoning and inferencing, the Apache Jena project [35] has been integrated with the GMAF, which comes with various APIs to define rules and to calculate inferences. As the Jena project is able to import RDFS and RDF files, the integration of the MMFG-RDFS-datastructure is implemented employing the RDF and RDFS export formats of the GMAF. The result of this integration is, that the GMAF framework can now calculate inferences and conflicts based on its own semantic model by passing RDF to Jena, letting Jena calculate the consistency and inferences of the model and thus define the Default Logics and the corresponding set of facts $\mathcal{F}$ and hypotheses $\mathcal{D}$. The code snippet in Listing 4 shows the exemplary steps to validate a model and to show conflicts.

```
1  // Generate the relevant GraphCode
2  Vector<GraphCode> gcs =
3  MMFGCollection.getInstance().getAllGC();
4  GraphCode relevantGC = TFIDF.calculateRelevantGC(gcs);
5  RDFExporter.export(relevantGC, "mmfgDataExport.rdf");
6
7  // Initialize Apache Jena
8  Model schema = RDFDataMgr.loadModel("mmfgSchema.rdf");
9  Model data = RDFDataMgr.loadModel("mmfgDataExport.rdf");
10 InfModel infmodel = ModelFactory.createRDFSModel(schema, data);
11
12 // Validate Collection
13 ValidityReport validity = infmodel.validate();
14 if (validity.isValid()) {
15         // everything fine
16 }
17 else {
18         // Conflicts
19         for (ValidityReport.Report r : validity.getReports()) {
20                 System.out.println(r);
21                 // process the conflict
22         }
23 }
```

Listing 4: Use of the explainability-feature of the GMAF.

The example in Listing 4 shows, that Jena is employed as a calculation engine for inferencing and reasoning based on the GMAF and MMFG representations (lines 8-10). All relevant MMFG information is exported to a *mmfgDataExport.rdf* file in RDF format (line 5), which is then loaded into Jena (line 9). Then, the inferencing model can be calculated (line 10) and a validity report can be generated (lines 13-23).

4.4. Summary

The exemplary implementations of the POC presented in this section show, that the proposed approach can actually be implemented and that both integration, and scalability, as well as explainability of the GMAF can be extended to become smarter. In the next section, an evaluation of this POC is presented.

5. Evaluation

In this section, details of the evaluation of the POC are discussed. Also following the structure of the previous sections, for each of the problem areas, selected experiments are presented, which outline the overall improvement of MMIR by employing Smart MMIR approaches. First, an evaluation of the integrability of the Smart MMIR components is given in 5.2. Then, experiments in the area of scalability are presented in 5.3, and finally, in 5.4 results in the area of explainability are presented.

5.1. Soundness

The introduction of *Soundness* provides additional insight and further expressiveness to users, which can be regarded as a major improvement of explainability in MMIR applications. Hence, in the following discussion, further experiments and the corresponding results are shown, which demonstrate the benefits of M_{SND} in various application areas.

The detetection of security relevant traffic scenes is one major task in the area of *Automotive and Autonomous Driving*. The introduction of *Soundness* can contribute to this task by comparing the actual traffic scene to expected or uncritical and secure traffic scenes. One major advantage of this is, that the calculation of *Soundness* falls down to simple matrix operations, which can be performed extremely fast, even in realtime, which is highly important in the area of autonomous driving. In the following experiment, we investigated, if and how *Soundness* can be employed to approve, if the behaviour of cyclists can be regarded as safe or if a higher risk for injuries has to be expected in case of an accident. Therefore, we took legal texts as *sound* input, which define the recommendations for safe cycling (like wearing a helmet) and created a *Graph Code* GC_{Safe} of this text. Then, a set of images has been processed with the GMAF to also calculated the corresponding *Graph Codes* GC_i. The images were taken from Adobe Stock [36] (see Figure 12).

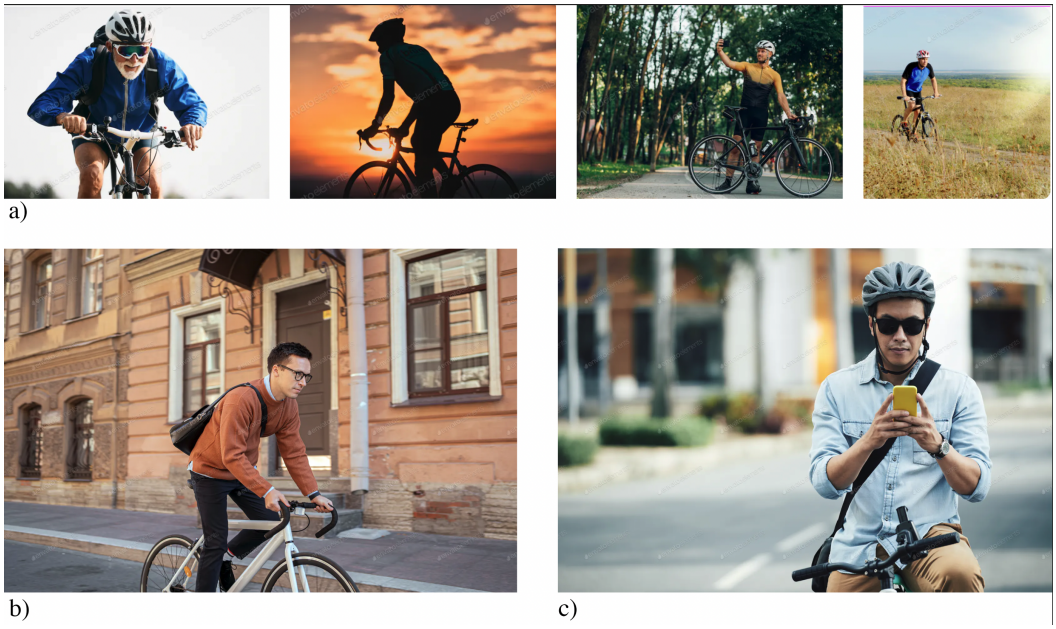


Figure 12. Calculation of Soundness in the area of traffic security.

GC_{Safe} contained vocabulary terms and relationships, that, e.g., described that wearing a helmet is safe, handling of smartphones during drivin is not safe, etc. In total, GC_{Safe}

had 132 vocabulary terms and the corrsponding relationships. For this experiment we did not use the intersection of GC_{Safe} and GC_i as this would lead to a loss of relevant safeness parameters. Instead, we decided to leave all 132 vorabulary terms and relationships as input for the calculation of *Soundness*. In total 250 images have been processed in this way. The results show, that no image fully complies to all vocabulary terms and relationships and thus provide a perfectly sound result. This was, of course, expected, as legal texts and the corresponding transformation into *Graph Codes*, as well as the object detection algorithms employed within the GMAF produce slightly different levels of features. Even after a semantic analysis based on *SGCs*, there was no perfectly sound result. However, the experiment shows, that most images of the chosen dataset produce a *Soundness* of $M_{SND} = 0.7 - 0.8$ (see example images shown in Figure 12a.). Some images show a significantly lower value as shown in Figure 12b with $M_{SND} = 0.53$ and Figure 12c with $M_{SND} = 0.62$. A visual examination shows, that images with lower M_{SND} values contain indicators for safety violations, like not wearing a helmet or dealing with a smartphone during cycling.

Another area, where *Soundness* can support MMIR processes, is the area of *News and Fake News*. As a underlying dataset, we selected the text archive of the Washington Post [38], which is also part of the reference datasets of the TREC conference [39] and contains about 750.000 articles in machine readable JSON-format (see Figure 13a.). These articles have been processed into *Graph Codes* (see Figure 13b).

a)

b)

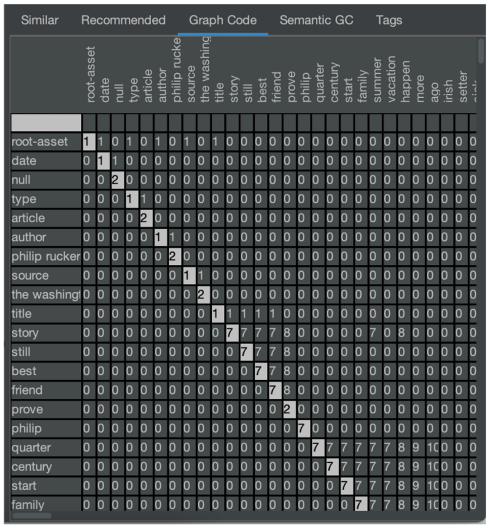


Figure 13. Washington Post article and the corresponding Graph Code.

Based on these prerequisites, we conducted two experiments. First, the *Soundness* between to articles in the same topic area is calculated. Second, the *Soundness* parameter is employed to determine contradicting documents within the same topic area. In both cases it is required to work on articles within a similar topic. It doesn't make sense to compare sports articles with international politics. As a starting point, we selected an article, that has also been employed during the TREC 2021 conference about "Coyotes in Maryland" (see Figure 14).

```
<top>
    <num> Number: 957 </num>
    <docid> e0b684ae-20d3-11e5-bf41-c23f5d3face1 </docid>
    <url>
https://www.washingtonpost.com/national/health-science/cats-may-not-be-as-much-of-a-threat-to-wildlife-as-previously-thought/2015/
07/06/e0b684ae-20d3-11e5-bf41-c23f5d3face1_story.html </url>
    <title> Coyotes in suburban Maryland </title>
    <desc> Find information about increasing numbers of coyotes in suburban Maryland and any impacts on other species. </desc>
    <narr>
    As coyotes have moved into the area other animals such as feral cats have been driven out. This can lead to the downturn of
the number of birds killed by the cats. While coyotes are natural predators, which get rid of rodents, they also have an impact by
attacking people and their pets.  Find information on the growing coyote population in Maryland and its impact on other species.
    </narr>
    <subtopics>
        <sub num="0">Find instances of coyotes attacking people and their pets in suburban Maryland.</sub>
        <sub num="1">How does the increased coyote population affect other wildlife?</sub>
        <sub num="2">Are coyotes becoming more common in the area?</sub>
    </subtopics>
</top>
```

Figure 14. Sample Article chosen as a topic for the calculation of Soundness.

Based on this starting point, different datasets have been selected for both experiments and M_{SND} has been calculated for the base article and the elements in the datasets. For the first experiment, a similarity search (based on M_F)has been performed to define the dataset. For the second experiment, a search for recommendations (i.e., somehow related articles) based on M_{FR} has been performed to define the dataset. The expectation is, that similar articles would mostly be *sound*, while in the recommendations also contradicting elements can be found. In this manner, we selected 25 documents for each experiment, the results are shown in Table 1.

Doc-Id	M_F	M_{SND}	**Doc-Id**	M_{FR}	M_{SND}
c23f5d3face1	1.0	1.0	c23f5d3face1	1.0	1.0
9736d04fc8e4	0.9987393	0.93	e7278db80d86	0.9987393	0.82
a83e627dc120	0.99747854	0.88	a83e627dc120	0.99747854	0.82
7f2f110c6265	0.99621785	0.91	7f2f110c6265	0.99621785	0.79
e7eb4319b8bc	0.99495715	0.89	7b9eba0f87d6	0.9924357	0.81
0034bb576eee	0.9936964	0.85	14b64f3d453f	0.991175	0.86
...	...	...	...	...	...
0047d15a24e0	0.96974283	0.94	d43a3ca733b4	0.9621785	0.91
a3ce76ec4751	0.9684821	0.88	d068924b49	0.96091783	0.88
fake news	0.9623122	0.64	fake news	0.93221342	0.59

Table 1. Soundness calculation based on the Washnington Post dataset.

In the first row of Table 1, the input document (see Figure 14) with Doc-Id "c23f5d3face1" is processed and - of course - achieves the highest possible value for similarity, recommendation and soundness. In the remainder of Table 1, the other documents of the 25 selected items and the corresponding processing values are shown. The last row in the table with Doc-Id "fake news" contains an article, that has been re-written based on the original text (see Figure 14) with the narrative "As birds have moved into the area other animals such as coyotes have been driven out. This can lead to the downturn of the number of other animals killed by the birds. While birds are natural predators, which get rid of coyotes, they also have an impact by attacking people and their pets." So basically, the terms "coyote, bird, other animals" have been switched to produce a fake news article.

The results for *Soundness* in this experiment show, that *Soundness* is independent from similarity or recommendations. Furthermore, it shows, that it can be employed for fake news detection, as the value for manually produced fake articles is significantly lower thant the values for the other articles. We assume, that the combination of all *Graph Code* metrics and M_{SND} will deliver best fake detection results. This will be further elaborated as part of future work. However, even this experiment shows, that M_{SND} can provide a highly relevant measure. Furthermore, it is important to highlight, that the calculation of M_{SND} falls down to simple matrix operations, which can be processed easily, efficiently, and even in parallel. This will be shown in the experiments in section 5.3. Also, a further compression in terms of *Feature Fusion* can be an additional means to compress the *Graph Codes* for processing. This is now shown in the next section.

5.2. Integration area

In the area of integration, relevance calculations can be performed by employing *Feature Fusion* strategies. To show the improvement of *Feature Fusion*, a qualitative experiment has been conducted, where resulting *Graph Codes* of images are compared. Figure 15 shows two *Graph Code* for the same image. In Figure 15a) the normal *Graph Code* is shown, while in Figure 15b) a Feature Fusion plugin has been applied, which removes irrelevant features according to the collection's content. In this experiment, the collection contained 200 photos of a photo shooting with the same person, same background, same clothing, etc. However only in few photos, the person on the picture was presenting a coffee cup.

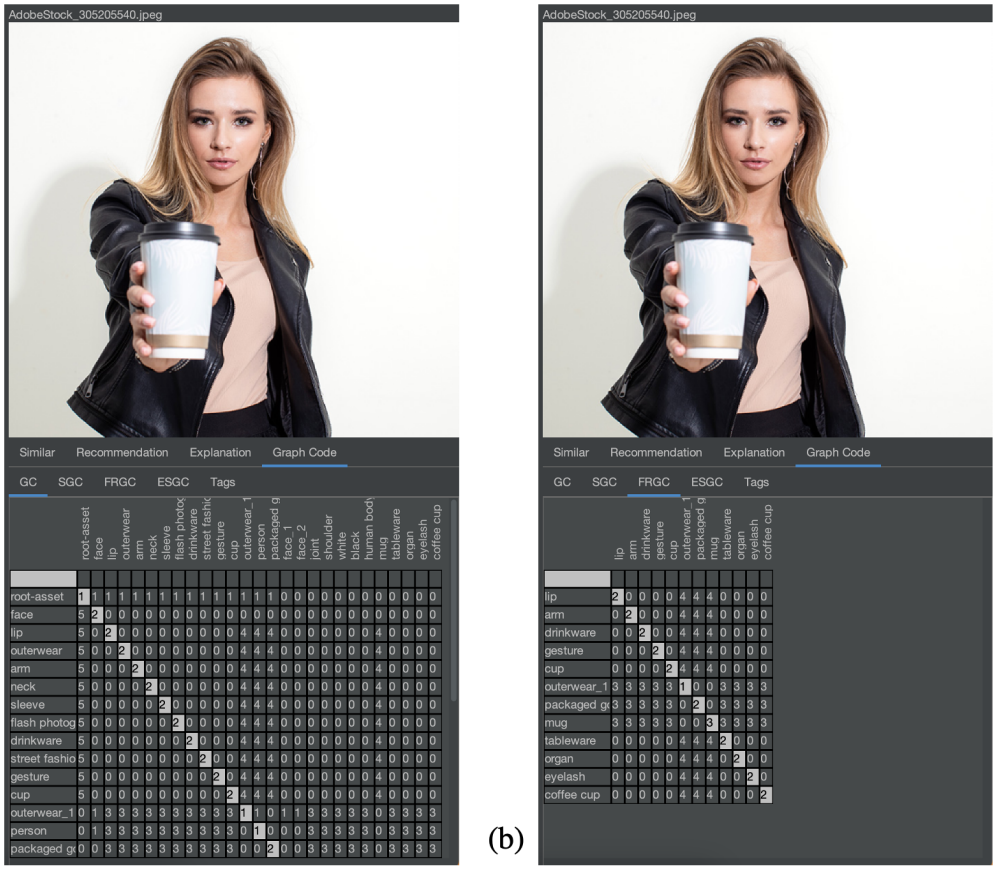


Figure 15. Feature Fusion for relevance calculation.

This experiment clearly shows the improvement of Feature Fusion and relevance calculations. And when considering, that the *Graph Code* in Figure 15b) now contains exactly the subset of MM Features, that is actually relevant for the collection, this becomes a very Smart MMIR solution. Of course, when looking for a "coffee cup", the image would have been found also without Smart MMIR. However, when asking questions like, "why is this image relevant?" or "what's the most important information on this image?", Smart MMIR can produce answers immediately. This is also further evaluated in the area of scalability and presented in the next subsection.

5.3. Scalability area

In the area of scalability, several quantitative experiments have been conducted to further refine and detail the set of experiments already shown in [2]. Figure 16 shows the corresponding results. The details of this extended evaluation are given in Tables 2 and 3 based on the number of input images c, the number of calculated MMFG nodes n, the corresponding edge number e, the Neo4J runtime with $p = 3$ (i.e., that Neo4J compares up to three links between nodes for similarity). The *Java* and *iPad* column shows the runtime

of the corresponding GMAF implementation. The evaluation of scalability is shown in Table 4 based on n nodes, i GMAF instances, the number a of multimedia objects per instance, and the runtime t for the execution of the experiment. Furthermore, in Table 5, the overall runtime based on the number of physical servers for horizontal scaling n_{HSC} and the number of instances per physical sever i_{HSC} is evaluated. Finally, Table 6 shows the parallelization (i.e., vertical scaling) based on CPU and GPU implementations of the *Graph Code* algorithms.

c	n	e	$N(p=3)$	*Java*
10	326	1591	8 ms	9 ms
20	634	3218	33 ms	18 ms
30	885	4843	62 ms	40 ms
40	1100	5140	196 ms	42 ms
50	1384	7512	272 ms	48 ms
60	1521	9979	380 ms	51 ms
70	1792	1231	533 ms	54 ms
80	1986	1482	786 ms	54 ms
90	2208	1705	1044 ms	58 ms
100	2479	1823	1262 ms	60 ms

Table 2. Scalability with the Flickr30K dataset. c

c	n	e	$N(p=4)$	$N(p=5)$	*Java*	*iPad*
10	558	3273	65 ms	1027 ms	10 ms	10 ms
20	870	5420	430 ms	4688 ms	18 ms	12 ms
30	1119	7799	1686 ms	44217 ms	26 ms	14 ms
40	1415	10501	3303 ms	63705 ms	35 ms	15 ms
50	1692	12994	3495 ms	75845 ms	39 ms	15 ms
60	2023	16078	4643 ms	-	39 ms	18 ms
70	2427	19776	-	-	39 ms	17 ms

Table 3. Scalability with the DIV2K dataset

n	i	a	t
1	1	720,000	635
1	2	360,000	320
1	3	240,000	214
1	4	180,000	164
1	5	144,000	129
1	6	120,000	110
1	7	102,000	96
1	8	90,000	81
1	9	80,000	75
1	10	72,000	73
1	11	65.000	71
1	12	60,000	68
1	13	55,000	67
1	14	51,000	66
1	15	48,000	65
1	16	45,000	65

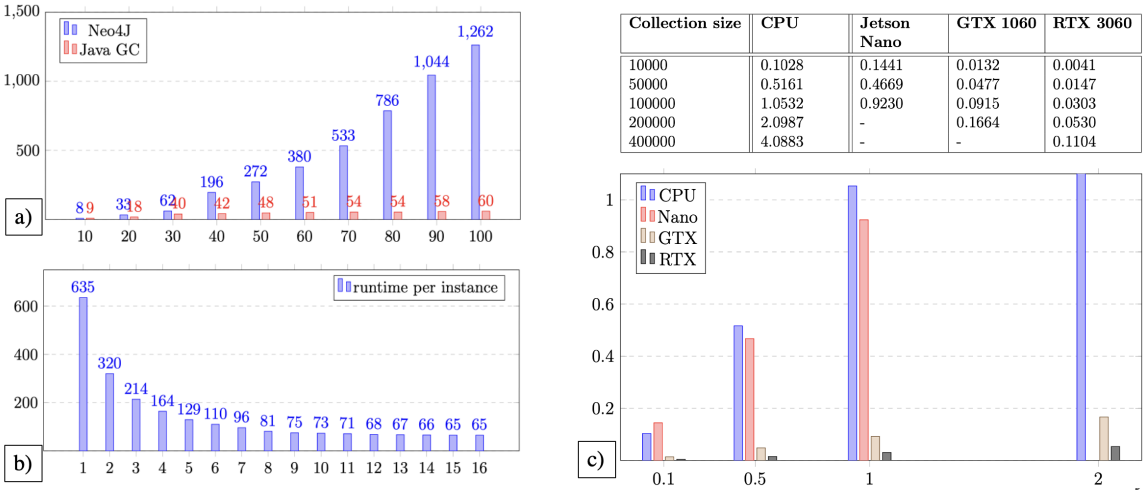
Table 4. Scalability, initial run on a single server with n GMAF-instances

n_{HSC}	i_{HSC}	a	t
1	8	90,000	81
2	8	45,000	41
3	8	30,000	29
4	8	22,500	22
5	8	18,000	17
6	8	15,000	14
7	8	12,850	12
8	8	11,250	11

Table 5. Scalability of nodes with 8 GMAF-instances each.

Processing Step	**CPU (Single thread)**	**Apple Metal (M1)**	**Nvidia Cuda GTX**	**Nvidia Cuda 2x RTX**
Ramp Up	27	2.430	3.015	2.130
Search 1	2.327	103	327	98
Search 2	2.406	107	342	102
Search 3	2.388	98	339	98
Ramp Down	625	792	1.210	723
Total	**7.773**	**3.530**	**5.233**	**3.151**

Table 6. Scalability, runtime measures (milliseconds) of vertical scaling on GPUs including ramp up and ramp down phases



Collection size	CPU	Jetson Nano	GTX 1060	RTX 3060
10000	0.1028	0.1441	0.0132	0.0041
50000	0.5161	0.4669	0.0477	0.0147
100000	1.0532	0.9230	0.0915	0.0303
200000	2.0987	-	0.1664	0.0530
400000	4.0883	-	-	0.1104

Figure 16. Results in the area of scalability.

In Figure 16a), a comparison of the runtime of a similarity search based on graphs (blue) and *Graph Codes* (red) is shown. For the graph calculations, a standard Neo4J database [37] has been employed and the calculated MMFGs have been inserted. On GMAF side, a standard Java implementation of the above mentioned metrics has been employed for this comparison. The experiment has been executed on the same machine. The results of this experiment clearly prove, that *Graph Codes* have a better scaling (linear vs. polynomic or exponential) than graph-based algorithms. In this experiment, a speedup of factor 20 has been achieved, however the switch to linear complexity is, of course, even more important than the numbers.

Figure 16b) shows the results of a runtime measuring of a horizontal distribution of GMAF instances, which perform *Graph Code* based operations. This also shows, that the overall runtime of a query processing can be reduced significantly by adding additional nodes to a GMAF setup. The optimal number of nodes for this particular experiment is between 8 and 10 and leads to an improvement of the overall processing time by a factor

of 8.01 (8 nodes with processing time of 81 seconds vs. 1 node with processing time of 635 seconds). For this experiment, huge collections containing 750,000 elements have been employed to get reliable results of the possible speedup.

Figure 16c) shows both the result values and a diagram of an experiment for vertical scaling on different hardware. In particular, here the CUDA implementation for NVIDIA GPUs has been evaluated. This experiment showed, that significant improvement can be achieved also within a single GMAF instance by enabling parallel processing. In this example, a speedup of factor 40 has been measured, which is only limited by the number of parallel processing units on the GPU. If, theoretically, the whole collection fits into the GPU memory, any MMIR processing can be performed in a single step producing results immediately.

Depending on the application, these three scaling methods can be flexibly combined and integrated with each other. If, e.g., these experiments are combined, the overall processing time can be reduced by factor $20 \times 8 \times 40 = 6.400$! This means, when the previous processing of a MMIR request took, e.g., 6.400 seconds (i.e., one hour and 45 minutes), the same request can be resolved with Smart MMIR in a single second.

5.4. Explainability area

For the area of explainability, various cognitive-walkthrough-based experiments have been conducted to evaluate, how Smart MMIR can improve the overall MMIR experience for users. As stated in the modeling section, further research is planned in this area. Therefore, the following experiments are mostly designed to confirm, that the changes in the areas of integration and scalability do not affect the existing solution. Therefore, in this subsection, two examples of these experiments are shown.

Figure 11 already showed the user interface for query refinement. On the right side, sections of a specific image have been marked as "relevant" (green bounding box) and "irrelevant" (red bounding box). The results of this refined query are shown in the center of this screenshot and demonstrate, that due to this refinement, now only white (or at least white-ish) dogs remain in the result list and black dogs have been removed automatically. In a second experiment, the textual visualization of MMIR processing steps has been evaluated. Figure 17 shows, how results of a GMAF search can now be explained automatically by comparing them to the query and applying the introduced metrics.

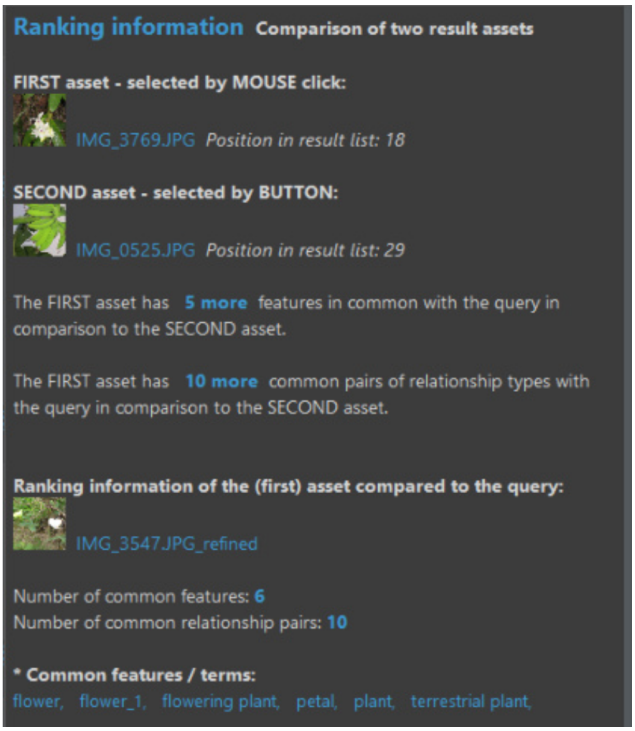


Figure 17. Visualization of ranking and comparison information.

5.5. Summary

Summarizing this evaluation section, it can be stated, that Smart MMIR improves existing MMIR solutions in all problem areas. The experiments show an increase in integrability, significant performance optimizations, and also UI components, that provide more expressiveness and explainability for the users. Particularly, the introduction of *Soundness* and the corresponding capabilities of SMART MMIR in various application areas, can improve existing solutions and applications. Therefore, the results of these experiments support the overall assumption, that Smart MMIR can provide benefits in all areas of MMIR.

6. Summary and conclusion

In this paper we introduced, defined, and evaluated our definition of the term "Smart MMIR" and showed, how Smart MMIR differs from standard MMIR. Based on previous work, Smart MMIR can be achieved by adding further modeling, formal calculations, and functional extensions to standard MMIR processes, components and processing steps. Smart MMIR improves MMIR in the following areas:

- interoperability and integration: the integration of processing flows and feature fusion provides significant benefit for the interoperability with other applications, the adaptation of solutions for distinct application areas, and the exchangability of algorithms for further refinements of MMFGs and *Graph Codes*.
- scalability: the improvements in the area of scalability are enormous. Both vertical and horizontal scaling provide a significant speedup of the overall processing time and their combination offers opportunities to increase the Smart MMIR experience for users.
- explainability and expressiveness: in addition to the already existing generation of human understandable texts based on ESMMFG and ESGC, further MMIR expressiveness is introduced to provide and visualize insight into MMIR processing steps.

All these areas are important for any modern MMIR application, algorithm, component, user interface, or framework. The Smart MMIR improvements can either be adapted

for other solutions, or integrated via the GMAF API to enrich applications with Smart MMIR mechanisms and algorithms.

Furthermore, Smart MMIR offers great opportunities for further research in the area of feature fusion, reasoning and inferencing, feature extraction, and feature detection. Therefore, Smart MMIR can be regarded as an important and relevant base technology in the area of Multimedia Information Retrieval.

Author Contributions: Conceptualization and methodology: Stefan Wagenpfeil and Matthias Hemmje. Software, validation, formal analysis, investigation, resources, data curation, writing: Stefan Wagenpfeil. Review, editing and supervision: Paul Mc Kevitt and Matthias Hemmje. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data presented in this study are openly available in [34].

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Statista Ltd. **(2020)**, *Social media - Statistics and Facts*. Available online: https://www.statista.com/topics/1164/social-networks/ (accessed 10.11.2022)
2. Wagenpfeil, S. and McKevitt P. and Hemmje, M. **(2021)**, *Fast and Effective Retrieval for Large Multimedia Collections*. Big Data Cogn. Comput. 2021, 5(3), 33; DOI: https://doi.org/10.3390/bdcc5030033
3. Wagenpfeil, S. and McKevitt P. and Hemmje, M. **(2021)**, *Towards Automated Semantic Explainability of Multimedia Feature Graphs*. Information 2021, 12(12), 502; DOI: https://doi.org/10.3390/info12120502
4. Wagenpfeil, S. and McKevitt P. and Cheddad, A. and Hemmje, M. **(2022)**, *Explainable Multimedia Feature Fusion for Medical Applications*. Information 2021, 12(12), 502; DOI: https://doi.org/10.3390/info12120502
5. Nunamaker, J. and Chen M. and Purdin, T.D.M. **1990**, *Systems Development in Information Systems Research*, Journal of Management Information System, DOI: 10.1080/07421222.1990.11517898
6. Library of Congress (U.S.) **(2020)**, *Text » Quality and Functionality Factors*. Available online: https://www.loc.gov/preservation/digital/formats/content/text_quality.shtml (accessed 10.11.2022)
7. Wagenpfeil, S. and Engel, F. and McKevitt P. and Hemmje, M. **(2021)**, *AI-Based Semantic Multimedia Indexing and Retrieval for Social Media on Smartphones*. Information 2021, 12(1), 43; DOI: https://doi.org/10.3390/info12010043
8. Kankanhalli, M. and Chua T. **(2000)**, *Video modeling using strata-based annotation*; Journal: IEEE MultiMedia, Volume 7, pp 68-74; DOI: 10.1109/93.839313
9. Needham M. **(2019)**, *Graph Algorithms*; Publisher: O'Reilly Media, Inc., ISBN: 978-1-492-05781-9
10. Wagenpfeil, S. and Engel, F. and McKevitt P. and Hemmje, M. **(2021)**, *Graph Codes-2D Projections of Multimedia Feature Graphs for Fast and Effective Retrieval*. Available online: https://publications.waset.org/vol/180 (accessed 10.11.2022)
11. Sciencedirect.com **(2020)**, *Adjacency Matrix*. Available online: https://www.sciencedirect.com/topics/mathematics/adjacency-matrix (accessed 2020)
12. Asim, M. and Wasim, M. and Ghani Khan, M. and Mahmood, M. and Mahmood, W. **(2019)**, *The Use of Ontology in Retrieval: A Study on Textual*; Volume 7, pp 21662-21686;
13. Domingue, J. and Fensel, D. and Hendler, J. **(2011)**, *Introduction to the Semantic Web Technologies*. Publisher: Springer, DOI: https://doi.org/10.1007/978-3-540-92913-0, ISBN: 978-3-540-92913-0
14. Bornschlegl, F. and Nawroth C. and Hemmje M. **(2016)**, *IVIS4BigData: A Reference Model for Advanced Visual Interfaces Supporting Big Data Analysis in Virtual Research Environments*, DOI: https://doi.org/10.1007/978-3-319-50070-6
15. W3C.org **(2021)**, *SKOS Simple Knowledge Organisation System*. Available online: https://www.w3.org/2004/02/skos/ (accessed 10.11.2022)
16. Aho, A. *Compilerbau*; ISBN: 9783486252941
17. Hausser R. **(2000)**, *Principles of Computer Linguistics*; Publisher: Springer, Berlin Heidelberg New York Barcelona Hongkong London Mailand Paris Singapur Tokio; ISBN: 3-540-67187-0

18. Chunlin, L. and Jianhang, T. and Youlong, L. **(2020)**. *Elastic edge cloud resource management based on horizontal and vertical scaling*. The Journal of Supercomputing. 76. DOI: https://doi.org/10.1007/s11227-020-03192-3.
19. Chien-Yu, L. and Meng-Ru, S. and Yi-Fang, L and Yu-Chun, L. and Kuan-Chou, L. **(2014)**. *Vertical/Horizontal Resource Scaling Mechanism for Federated Clouds.* ICISA 2014 - 2014 5th International Conference on Information Science and Applications. 1-4. 10.1109/ICISA.2014.6847479.
20. Shamsuddeen, R. and Chan, Y. and Sharifah, M. **(2022)**. *A Cloud-Based Container Microservices: A Review on Load-Balancing and Auto-Scaling Issues.* International Journal of Data Science. 3. 80-92., DOI: https://doi.org/10.18517/ijods.3.2.80-92.2022.
21. Silge, J. and Robinson, D. **(2022)**, *Text Mining with R - a tidy approach*. Available online: https://www.tidytextmining.com/tfidf.html (accessed 2020-09-03)
22. Krig, S. **(2016)**, *Interest Point Detector and Feature Descriptor Survey*; Publisher: Springer, pp 187-246; ISBN: 978-3-319-33761-6
23. Jabeen, S. and Mehmood, Z. and Mahmood T. and Saba, T. and Rehmann A. and Mahmood, M. **(2018)**, *An effective content-based image retrieval technique for image visuals representation based on the bag-of-visual-words model*; Journal: PLoS ONE, Volume 13
24. Zhao, W. and Zhao, F. **(2020)**, *Learning Specific and General Realm Feature Representations for Image Fusion*; Journal: IEEE Transactions on Multimedia, Volume PP, pp 1-1
25. Zheng, C. and Wu, Z. and Wang, T. and Cai, Y. and Li, Q. **(2021)**, *Object-Aware Multimodal Named Entity Recognition in Social Media Posts With Adversarial Learning*; Volume 23, pp 2520-2532
26. Nawroth, C. and Engel, F. and Eljasik-Swoboda, T. and Hemmje, M. **(2018)**, *Towards Enabling Named Entity Recognition as a Clinical Information and Argumentation Support*; Journal: Proceedings of the 7th International Conference on Data Science, Technology and Applications, pp 47–55; DOI: https://doi.org/10.5220/0006853200470055
27. Lin, Y. and Sundaram, H. and De Choudhury M. and Kelliher, A. **(2012)**, *Discovering Multirelational Structure in Social Media Streams*; Journal: TOMCCAP, Volume 8, pp 4; DOI: https://doi.org/10.1145/2071396.2071400
28. Snoek, M. and Worring, M. and Hauptmann, A. **(2006)**, *Learning rich semantics from news video archives by style analysis*; Journal: ACM Transactions on Multimedia Computing, Volume 2, DOI: https://doi.org/10.1145/1142020.1142021
29. Hong, R. and Tang, J. and Tan, H. and Ngo, C. and Yan, S. **(2011)**, *Beyond Search: Event-Driven Summarization for Web Videos*. ACM Transactions on Multimedia Computing, Communications, and ApplicationsVolume 7, Issue 4, DOI: https://doi.org/10.1145/2043612.2043613
30. Goh, K. and Li, B. and Chang, E. **(2005)**, *Semantics and Feature Discovery via Confidence-Based Ensemble*. ACM Transactions on Multimedia Computing, Communications, and ApplicationsVolume 1, Issue 2, DOI: https://doi.org/10.1145/1062253.1062257
31. Cooper, M. and Foote, J. and Girgensohn, A. and Wilcox, L. **(2004)**, *Temporal Event Clustering for Digital Photo Collections*; ACM Transactions on Multimedia Computing Communications and Applications, Volume 1, DOI: https://doi.org/10.1145/957013.957093
32. Arslan, S. and Yazici, A. **(2019)**, *Content And Concept Indexing For High-Dimensional Multimedia Data*; 2019 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE), 2019, pp. 1-6, doi: https://doi.org/10.1109/FUZZ-IEEE.2019.8858870.
33. Norman, D. and Draper, S. **(1986)**, *User Centered System Design - New Perspectives on Human-computer Interaction*; Publisher: Taylor & Francis, Justus-Liebig-University, ISBN: 978-0-898-59872-8
34. Wagenpfeil, S. **(2021)**, *Github Repository of GMAF and MMFVG*. Available online: https://github.com/stefanwagenpfeil/GMAF/ (accessed 10.11.2022)
35. Apache Software Foundation **(2020)**, *Reasoners and rule engines: Jena inference support*. Available online: https://jena.apache.org/documentation/inference/ (accessed 10.11.2022)
36. Adobe Inc. **(2020)**, *Adobe Stock*. Available online: https://stock.adobe.com (accessed 10.11.2022)
37. Neo4J Inc. **(2021)**, *Neo4J Graph Database*, Available online: https://neo4j.com (accessed 15.12.2021)
38. The Washington Post **(2021)**, *Washington Post Archives*. Available online: https://www. washingtonpost.com (accessed 10.11.2022)
39. The Text Retrieval Conference (TREC) **(2021)**, *TREC Datasets*. Available online: https://trec.nist.gov/data.html (accessed 10.11.2022)