Impact of Interaction Strategies on User Relevance Feedback

Omar Shahbaz Khan
IT University of Copenhagen
Copenhagen, Denmark
omsh@itu.dk

Björn Þór Jónsson
IT University of Copenhagen
Copenhagen, Denmark
bjth@itu.dk

Jan Zahálka
Czech Technical University in Prague
Prague, Czech Republic
jan.zahalka@cvut.cz

Stevan Rudinac
University of Amsterdam
Amsterdam, Netherlands
s.rudinac@uva.nl

Marcel Worring
University of Amsterdam
Amsterdam, Netherlands
m.worring@uva.nl

ABSTRACT

User Relevance Feedback (URF) is a class of interactive learning methods that rely on the interaction between a human user and a system to analyze a media collection. To improve URF system evaluation and design better systems, it is important to understand the impact that different interaction strategies can have. Based on the literature and observations from real user sessions from the Lifelog Search Challenge and Video Browser Showdown, we analyze interaction strategies related to (a) labeling positive and negative examples, and (b) applying filters based on users' domain knowledge. Experiments show that there is no single optimal labeling strategy, as the best strategy depends on both the collection and the task. In particular, our results refute the common assumption that providing more training examples is always beneficial: strategies with a smaller number of prototypical examples lead to better results in some cases. We further observe that while expert filtering is unsurprisingly beneficial, aggressive filtering, especially by novice users, can hinder the completion of tasks. Finally, we observe that combining URF with filters leads to better results than using filters alone.

CCS CONCEPTS

• **Information systems** → **Multimedia and multimodal retrieval**; **Evaluation of retrieval results**.

KEYWORDS

Interactive Learning, User Relevance Feedback, Multimedia Retrieval, Interaction Strategies

ACM Reference Format:
Omar Shahbaz Khan, Björn Þór Jónsson, Jan Zahálka, Stevan Rudinac, and Marcel Worring. 2021. Impact of Interaction Strategies on User Relevance Feedback. In *Proceedings of the 2021 International Conference on Multimedia Retrieval (ICMR '21), August 21–24, 2021, Taipei, Taiwan.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3460426.3463663

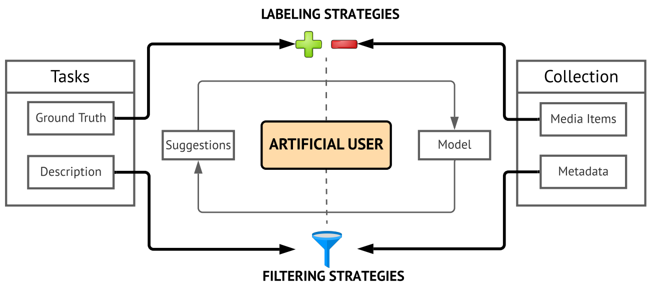


Figure 1: Our proposal for an automated evaluation process to understand the impact of user interaction strategies: We create artificial users that label suggestions from the collections using different strategies and apply metadata filters based on different levels of domain knowledge.

1 INTRODUCTION

Interactive Learning (IL) is a multimedia retrieval approach, where a user and system work together to build a model in order to satisfy a user information need [8, 9, 16, 19, 36, 37]. The system presents users with media items from a collection that they label as positive or negative, only positive [25], or only negative [31]. The labeled items are then used to train a classifier that is deployed to retrieve new suggestions. Some IL systems provide additional features, such as: filters to focus on subcollections; text search to find positive examples; or advanced browsing features, such as an event timeline. IL is a continuous process that stops when the user considers the task to be complete. User Relevance Feedback (URF) is a variation of IL that focuses on quick convergence of the user's information need by providing the most relevant items from the system's model in each interaction round. Active Learning, another variation of IL, suggests items from the collection that are most valuable to improve the model, rather than the most relevant ones [12]. Since we are focusing on scenarios where the user is seeking relevant items, we focus on URF.

Traditionally, evaluation of URF focuses on measuring the classification performance of the model used to retrieve items [11, 29, 34, 35]. Recently, automated evaluation protocols have been used to evaluate URF systems with a focus on artificial users [16, 23, 36] for large multimedia collections. However, these are based on *one* interaction strategy where the artificial user labels relevant items as positives and the system labels everything else as negatives. Intuitively, when using URF systems, a common assumption is that more positive examples build a better model.

We conjecture that for many tasks it may be beneficial to have a smaller set of stronger positive examples. Understanding when to prefer one interaction strategy over another can greatly reduce the time of convergence for an information need. To garner this understanding, the first thought might be to conduct user studies. However, user studies are expensive and allow users too much freedom which makes them impractical for fine-tuning models and analyzing the impact of specific interaction strategy. To evaluate URF systems with an emphasis on understanding the impact of interaction strategies, an automated approach is necessary.

This paper uses such an automated approach to analyze the impact of different interaction strategies for URF systems. First, we define several strategies for labeling positive and negative examples and study their impact on result quality. Second, we explore the impact of 4 classes of artificial users applying filters based on their level of domain knowledge. To evaluate the interaction strategies, we define automated evaluation protocols based on three multimedia collections, two from the interactive search challenges Lifelog Search Challenge [10] and Video Browser Showdown [28], and one for VOPE-8hr [39], a domain-specific forensic research collection. Figure 1 outlines the proposed evaluation process, where artificial users use the strategies to guide the relevance feedback process.

The knowledge gained from analyzing these interaction strategies can benefit the training of new users of URF systems, as well as help experienced users improve their performance on specific tasks. It could also be incorporated into systems as a feature to automatically suggest suitable interaction strategies for different tasks. Specifically, this paper contributes three best practice guidelines:

1. Labeling strategies impact results significantly, in particular, strategies with more examples are not always better.
2. Adding URF is always as good or better than just using filters.
3. While filtering is often beneficial, overly aggressive filtering can adversely affect the ability to complete tasks.

2 RELATED WORK

User Relevance Feedback has been used since the 1960s to improve queries for information retrieval [24] and saw a boom in the 1990s and 2000s for multimedia retrieval [11, 12, 29, 33–35]. Later, it started fading as hash based approaches [20], product quantization [13], and deep learning models [7, 32, 40] were proven more efficient for retrieval on large-scale collections. However, in recent years the issue of scalability has largely been resolved and the state-of-the-art URF systems for large scale multimedia retrieval are competitive with other approaches and require fewer examples to train their models than the supervised approaches [16, 18, 36].

Since users are central to URF systems, it is important that the evaluation methods of these systems account for their behavior. Early evaluation efforts for relevance feedback utilized collections that had relevance judgement mappings between queries and associated documents [1, 2]. This allows for automating the evaluation process with the simulated "user" judging items based on the relevance judgement mappings. This form of evaluation with optimal users that have knowledge about the ground truth has remained the most common form for URF systems to date. Some evaluation protocols use this for labeling the suggestions as positive or negative [6, 29]. Other evaluation protocols, especially those that work with large-scale collections, also add additional arbitrary negatives [16, 23, 25, 36]. Analytic Quality uses artificial actors which solve an analytic task derived from an existing benchmark/user task, measuring precision and recall over time and estimating the user's insight gain [38].

While the plethora of work on automated evaluations contributes to show the effectiveness of URF systems in various fields, the evaluation methodology only captures the behavior of a specific interaction strategy which may not be a strategy a real user will resort to. Aside from this there has also been work that has focused on evaluating systems with real users [23, 26, 29, 39]. URF systems typically showcase up to 30 images in each round and depending on the restrictions they can label as many items as they want [29], be limited to label a few examples [26], or only label examples as positive [25]. These evaluations give greater insight towards user behaviour, but they rarely generalize the interaction strategies due to the inherently limited set of users (tens at most).

With real users, it is also important to study the impact of users with different levels of knowledge. Dividing the users into users with relevant or no domain knowledge, it is possible to show that the performance of labeling examples or applying filters can be greatly affected [10, 27, 28].

From the related work, we identify a gap between artificial users and real users and to the best of our knowledge no work has focused on the impact this can have when evaluating URF systems. Hence there is a need for considering various labeling strategies that are inspired by real users, as well as establishing different levels of domain knowledge when applying filters.

3 USER INTERACTION STRATEGIES

To evaluate URF systems in an automated way, we need *artificial users*, software agents that simulate user behaviour. Their task is to find one or more relevant item(s) from a collection C, based on a textual description of an information need. To achieve this, they follow a certain strategy for labeling examples. Additionally the users apply filters based on different levels of domain knowledge. In the remainder of this section we propose a variety of labeling and filtering strategies to better understand the impact the different strategies can have on the performance of a URF system.

3.1 Labeling Strategies

The common labelling strategy of marking ground truth items as positives and everything else as negatives [4, 14, 16, 36] may result in a near empty set of positives and a vast set of negatives, where some negatives might feature relevant content. Furthermore, the task objective can involve finding all items in the relevant set, finding as many relevant items in r rounds as possible, or stopping once the first relevant item is encountered. To support evaluation for the different task objectives we must define strategies that use the ground truth items to rank suggestions and select the best suggestions as positives.

All strategies in this paper are based on observations of URF systems used in live interactive search challenges, in particular the

Lifelog Search Challenge and the Video Browser Showdown [10, 15, 17, 19, 21, 28]. We assume that the collections are comprised of images or videos represented with semantic feature vectors that can be compared using a distance metric. Each strategy uses a distance function with two feature vectors with semantic concepts extracted using neural networks as input; $d(\mathbf{v}_x, \mathbf{v}_{max})$. The first vector is the item from the suggestion set $\mathcal{S}_r$ of the current interaction round r. The second is the max-pooled feature vector of the relevant items. We use the Euclidean distance: it is simple, efficient, well-researched, and works well with using a compressed representation [36].[1]

We have identified three major categories of labeling strategies, *Accumulative Sets*, *Fixed Positive Sets* and *Arbitrary Negative Sets*, that we now describe in detail.

3.1.1 Accumulative Sets. Continuously adding items to the positive and negative set is a typical behavior of users that are attempting to gradually improve the model. This leads to the first strategy.

±*AccAdd* — Accumulative Sets with Additions: Label the p nearest items to $\mathbf{v}_{max}$ in $\mathcal{S}_r$ as positive, adding them to $\mathcal{P}_r$, the set of positives for round r, and label n furthest items from $\mathbf{v}_{max}$ in $\mathcal{S}_r$ as negative, adding them to $\mathcal{N}_r$, the set of negatives for round r.

$$\mathcal{P}_r = \mathcal{P}_{r-1} \cup \arg\min{}^{p}_{x \in \mathcal{S}_r} (d(\mathbf{v}_x, \mathbf{v}_{max})) \tag{1}$$

$$\mathcal{N}_r = \mathcal{N}_{r-1} \cup \arg\max{}^{n}_{x \in \mathcal{S}_r} (d(\mathbf{v}_x, \mathbf{v}_{max})) \tag{2}$$

As users keep adding to the sets, examples from earlier interaction rounds may become less important or even bad for the model. Therefore, users may start replacing items to improve the model; this behavior is typically observed from more experienced users.

±*AccRep* — Accumulative Set with Replacements: The user is allowed to replace items from the positive or negative sets if better representatives exist in the suggestion set $\mathcal{S}_r$, or in the labeled sets $\mathcal{P}_{r-1}$ and $\mathcal{N}_{r-1}$. The positive and negative sets at round r have the size pr and nr respectively.

$$\mathcal{P}_r = \arg\min{}^{pr}_{x \in \mathcal{S}_r \cup \mathcal{P}_{r-1} \cup \mathcal{N}_{r-1}} (d(\mathbf{v}_x, \mathbf{v}_{max})) \tag{3}$$

$$\mathcal{N}_r = \arg\max{}^{nr}_{x \in \mathcal{S}_r \cup \mathcal{P}_{r-1} \cup \mathcal{N}_{r-1}} (d(\mathbf{v}_x, \mathbf{v}_{max})) \tag{4}$$

An example of a positive example moving to the negative set is when an early positive example becomes a negative example as the model evolves. By replacing items, the chances of building a stronger model is enhanced. ±*AccRep* is a strategy that a user may utilize early on in a session but as the size of the positive and negative sets increases, the task of replacing items will become too time consuming. Therefore, even if this strategy works well, it may not be an optimal strategy during long sessions.

Enforcing an accumulative strategy increases the chances for overfitting the model towards certain features, which can be especially bad if erroneous items, e.g., incorrectly labeled, are added.

3.1.2 Fixed Positive Sets. Limiting the positive and negative sets to a fixed size, where the user can only replace items after the first round, avoids overwhelming the user with trying to replace from large sets. Instead, such strategies solely rely on the user's ability to replace bad examples when better ones are encountered. These strategies tend to be more dynamic, and are suitable for tasks where the user looks for strong archetypes to model the categories of relevance. However, limiting the sets can hurt the classification model for tasks where more items are required.

±*FixRep* — Fixed Set with Replacements: Restricts the size of both sets to p and n respectively.

$$\mathcal{P}_r = \arg\min{}^{p}_{x \in \mathcal{S}_r \cup \mathcal{P}_{r-1} \cup \mathcal{N}_{r-1}} (d(\mathbf{v}_x, \mathbf{v}_{max})) \tag{5}$$

$$\mathcal{N}_r = \arg\max{}^{n}_{x \in \mathcal{S}_r \cup \mathcal{P}_{r-1} \cup \mathcal{N}_{r-1}} (d(\mathbf{v}_x, \mathbf{v}_{max})) \tag{6}$$

Next is a hybrid strategy that users might use for extremely descriptive tasks, where good positive examples are rare, and where the limitation on negatives cannot train the model well enough to find the good positive examples. By restricting the positive set to only the strongest positives, but continuously adding to the negative set, the model can potentially be steered towards the relevant items. This strategy can be linked closely to negative relevance feedback as it is mainly guided by its negative set in the initial rounds [31].

+*FixRep-AccAdd* — Fixed Positive Set, Accumulative Neg. Set: Fixed positive set (Eq. 5) and accumulative negative set (Eq. 2).

3.1.3 Arbitrary Negative Sets. There is work that suggests that spending time on labeling negatives may not be as important as labeling positives [16, 38]. It is therefore crucial to investigate the impact of arbitrarily choosing negatives from either the suggested item set or the overall collection. Both use $arb(\mathcal{X}, n)$, a function which selects n arbitrary elements from a media item set $\mathcal{X}$. We define two strategies, whose positive strategies follow Eq. 1.

+*AccAdd-ArbLoc* — Arbitrary Negative Set (Local): Label n negatives arbitrarily from $\mathcal{S}_r$ and add them to $\mathcal{N}_{r-1}$.

$$\mathcal{N}_r = \mathcal{N}_{r-1} \cup arb(\mathcal{S}_r \setminus \mathcal{P}_r, n) \tag{7}$$

+*AccAdd-ArbGlo* — Arbitrary Negative Set (Global): Label n negatives from the entire collection C and add them to $\mathcal{N}_{r-1}$.

$$\mathcal{N}_r = \mathcal{N}_{r-1} \cup arb(C \setminus \mathcal{P}_r \setminus \mathcal{N}_{r-1}, n) \tag{8}$$

3.2 Filtering Strategies

Another aspect of interactive retrieval sessions is applying filters to reduce the scope of the analysis. Typically, filters are set based on metadata or features extracted from the items, and they can be added or removed at any point during a session. The reasoning behind the chosen filters can vary between users and the quality of filters can often depend on the level of *domain knowledge* they have. We define 4 types of users based on their expertise; *No Filter*, *Novice*, *Expert* and *Data Author*.

No Filters: To act as a baseline, this user type only utilizes the interactive learning system by labeling the retrieved suggestions, without applying any filters.

Novice: Users that tend to read a query and try to match the text with matching filters. This reflects the behavior of new users starting to work with a collection or system. However, it can also lead to misinterpretation of the query and exclusion of relevant items. This has indeed been observed during the novice sessions of interactive search challenges [10, 28]. The reasons for misinterpretations include time pressure, misusing the system, lack of domain knowledge, and language barriers. For example, consider the following

[1]We also experimented with Mahalanobis distance but found it to be less effective.

query: “Walking on a green footpath, to my car. I remember I had come off a flight and it was around lunch-time. I got into my car and drove to have a meal. No, I drove to work where I had lunch” [10]. A *Novice* user might attempt a filter such as “work”, excluding the relevant image in which a person was walking on a green footpath.

Expert: Represents analysts with expert domain knowledge. They know the system, have adequate knowledge of the collection and are able to interpret tasks beyond their description. They can connect query information with metadata from previous acquired knowledge, such as setting a location filter from a reference to a person who was seen at that location in a prior session. Semantic filters, such as "morning" or "evening" for hours, also fall under this.

Data Author: This type represents users with detailed knowledge, gained from having either created the collection or maintained it by updating or adding metadata. They are thus able to go beyond the query and apply filters due to actual recollection of creating the desired item in the collection. They may also use external information for clarification, as they may recall a detail which can be found via personal files or the internet, e.g., using a query regarding a place they visited but forgot the name of. Note that this user type is only applicable on collections that have a handful of contributors.

3.3 Summary

We conjecture that the labeling and filtering strategies presented in this section have a strong impact on the performance of URF systems. This has to date not been sufficiently captured by existing evaluation methods. In the remainder of the paper we therefore conduct experiments that analyze the impact of the different labeling and filtering strategies with a variety of collections and tasks.

4 EXPERIMENTAL SETUP

The experiments are conducted on tasks from 3 collections with varying objectives, query details and metadata quality. The key metric is completion time, or how many interaction rounds it takes on average to finish a task. While observing the behavior of labeling and filtering strategies for a handful of rounds is interesting, the direct impact of a strategy will ultimately be reflected in the time to finish the task. In addition, recall is also a relevant metric for tasks with time limits, or tasks that require finding all relevant items.

An actor [38] is an artificial user that uses a particular labeling and filtering strategy, and has a unique arbitrary starting point for each relevant task. Each actor communicates with a URF server using a script to perform the relevance feedback. To conduct the experiments we use a URF system where 25 items are suggested each round. The underlying classification model is linear SVM, which has a good accuracy/speed ratio and is consistent with the state of the art [16, 37]. During each interaction round the actor has to label p positives and n negatives from the 25 suggestions and apply filters depending on their labeling and filtering strategy respectively. All the results reported in this paper are an average of 50 different actors for each labeling and filtering strategy combination.

4.1 LifeLog Search Challenge 2019

Lifelogging is the idea of recording everything one does digitally, such as taking 2-3 images via a body camera every minute and logging daily routines manually and by using smart gadgets. Lifeloggers tend to end up with a large collection of images and metadata. The Lifelog Search Challenge (LSC) is an interactive live search challenge featuring a small curated lifelog collection [10]. The collection used in LSC2019 contains 41,666 images represented as 1000 dimensional feature vectors extracted with a deep neural network using concepts from ImageNet [5]. The collection contains metadata such as location, day and time that are useful as filters. Additional data, such as eating logs, fitness information, personal notes, are excluded as they are only available for a subset of the collection.

LSC2019 featured 24 interactive tasks with corresponding ground truths, where each task aimed to find images relevant to a textual query describing events from the lifelogger. The descriptions are extended through iterations, where each iteration adds some new information. Every iteration lasts 30 seconds with a total of six iterations. The nature of the task description is that of a memory, where one iteration may contradict a statement made in a previous one. The descriptions also typically contain information that can be correlated to metadata. The objective of a task is to find any of the relevant items; for some tasks the relevant set contains only a handful of images, while for a few it contains 50+ items. Due to the quality and transparency of the metadata, all types of users defined in Section 3.2 are applicable to this collection.

4.2 Video Browser Showdown 2020

The Video Browser Showdown [28] (VBS) is a live interactive search challenge similar to LSC. The latest edition of VBS was in 2020 and used the V3C1 collection that consists of 1000 hours of video segments or approximately 1M keyframes, from the online video site Vimeo [3]. We refer to this collection as VBS2020.

Unlike LSC2019, which consists of images from a single user, these videos are from different users all over the world. The users have free reign over the metadata, such as video categories and tags related to the video. While categories have a fixed number of options to select from, it is up to the user to determine which fits the video, making it highly subjective. The tags have no real restrictions, allowing the user to define their own tags. The categories and tags metadata are considered video level filters as they only refer to entire videos. The keyframes have also been processed for number of faces visible, making it possible to set this as a keyframe level filter. Note that our system uses keyframes from the videos as representatives of the segments. The representation for the keyframes is a more detailed feature vector with 12,988 dimensions [22].

VBS2020 featured 13 known item search tasks. The tasks describe visual events from a specific video segment. The descriptions focus more on visual features compared to metadata information. These tasks are presented through iterations as well, with time intervals of 60 seconds and a total of three iterations. The task objective is to find any relevant video segment of the described event. The collection also has more vaguely described tasks called Ad-hoc video search, where the goal is to find as many segments as possible that match the description. However, these tasks have been omitted since they lack a ground truth.

Due to the metadata having no central curator it is difficult to define a user for *Data Author* user strategy from Section 3.2. We therefore focus on *Novice* and *Expert* user strategies instead.

4.3 VOPE-8hr

The VOPE-8hr [30, 39] collection is inherently different from the previous two, both in terms of scale and objective. It consists of 8 hours of video broken into shots of 3 seconds, making it the smallest collection with ~9600 items. VOPE-8hr is a domain-specific collection for forensic research and the associated tasks are to find extremist propaganda content of three types; Neo-Nazis (task 1), Islamic terrorists (task 2) and Scottish ultra-nationalism (task 3). These tasks differ from the other two collections' tasks as the objective is to find *all* relevant examples of which some are easily identifiable and others being needles in a haystack. In addition to this, the collection is intentionally curated to have a portion of "red-herring" data that shares visual similarity to the relevant items but contextually is completely irrelevant [39]. The number of items to find for each task also differs, with tasks 1 and 3 having roughly 50 items and task 2 having 684 items. As there is no metadata that a real user could use as filters, no metadata filtering strategy experiments have been run for this collection.

5 EXPERIMENT 1: LABELING STRATEGIES

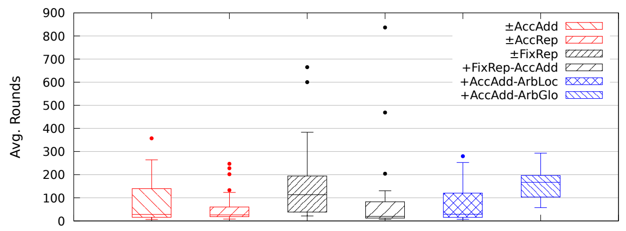
A baseline experiment is run with all the labeling strategies from Section 3.1 where $p = 5$ and $n = 15$ with no filter options.[2] Typically in a URF setting the starting point is arbitrary. Therefore, selecting more negatives than positives in each round can be beneficial for directing the classifier quicker to the relevant item, as the initial items may not contain any good positive examples.

5.1 LSC2019

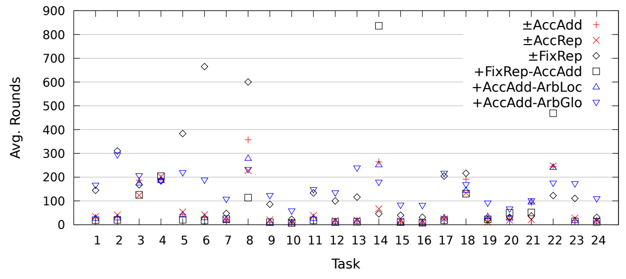
Figure 2(a) shows the number of rounds required to complete tasks for each labeling strategy on LSC2019. The two leftmost boxes show the distribution for the two *Accumulative* strategies. While their median is the same, ±*AccRep* is far more consistent than ±*AccAdd*, indicating that replacing items from the positive and negative set while adding items is better than just adding items. The two boxes in the middle show the *Fixed Positive* strategies. The hybrid strategy +*FixRep-AccAdd* is far better than the ±*FixRep* strategy. However, it does have some tasks where it does extremely poorly, as indicated by its outliers. We explore this in more detail with Figure 2(b) below. Lastly we have the two boxes on the right for the *Arbitrary Negative* results. Of the two, +*AccAdd-ArbLoc* is the better, meaning that labeling arbitrary negatives from the suggestion set is better than labeling them from the whole collection. If we compare this strategy with its *Accumulative* counterpart ±*AccAdd*, it is nearly identical in performance if not slightly better. Note that similar effect is observed from using -*ArbLoc* with +*FixRep* (not shown).

Figure 2(b) shows the average rounds per run for each task from LSC2019. The majority of tasks follow the pattern of tasks 1 and 2, which have ground truths with many near-duplicate images in the collection: "...looking at an old *clock*, with *flowers* visible. There was a *lamp* also..." (task 1), "A red *car* beside a white *house*..." (task 2). While most strategies fare well with these tasks, ±*FixRep* and +*AccAdd-ArbGlo* are consistently bad, but for different reasons. For these tasks, the ±*FixRep* strategy can end up in a state where it cannot improve the model as no stronger positive or negative

[2]We have run experiments with $p \in \{1, 3, 5, 7\}$ and $n \in \{5, 10, 15, 20\}$. Since the results support the conclusions in this section, we do not discuss them further.



(a) Avg. rounds per run for each strategy



(b) Avg. rounds per run for each task

Figure 2: Baseline results for LSC2019. (a) shows the average rounds it takes to complete the tasks, emphasizing the distribution for each strategy. (b) shows the average rounds per run for each task separately.

examples can be found, and the strategy simply browses through the model's ranked list of suggestions. +*AccAdd-ArbGlo*, on the other hand, explores the search space more sporadically as it labels negatives from the whole collection. Since the actor must select some positives in every round, this sporadic exploration leads to many bad positives, resulting in a poor model for the tasks.

There are a few tasks that show a different pattern. First, tasks 14 and 22 are cases where the description leads to many positive examples: "I was in my office taking a skype call... large image of a man's face on the *screen*..." (task 14). There is an abundance of computer, laptop, tablet, smartphone and tv related screens in this collection, and for this task the screen relates to a laptop/notebook screen which is seen in roughly one-third of the images in the collection. This can be bad for +*FixRep-AccAdd*, as the -*AccAdd* part will add many of these screens to its negative set. The other strategies counter this by adding screens also to the positive set. ±*FixRep* is a good option here, because of the exact same reason it is bad in the other tasks: few or none of these near-duplicates end up in the negative set, resulting in a better model. This scenario refutes the assumption that more examples are always better.

Second, task 8 has ground truth with visual features that are a mix of common and distinctive features, where the common features can lead the model away from the distinctive features: "Walking on a green footpath, to my *car*..." (task 8). Here, the ground truth item consists of a parking lot with several cars. While the collection consists of many different types of cars, some are abundant in the collection, while others are rarer. For this particular task, one of the abundant types is "minivan" which has approximately 3,000 related items, while one of the rarer types is "sports car" with roughly

50 items. *+FixRep-AccAdd* is the best strategy in this case, as it limits the positive set to the strongest examples. While suggestions with the common "minivan" feature will continuously appear, some of them will be labelled negative, which in effect will allow the distinctive "sports car" feature to dominate the model.

5.2 VBS2020

Figure 3(a) shows the average rounds per run to complete the tasks. As this collection is roughly 20 times larger than LSC2019, we restrict the number of rounds a session can take to 500. Since some tasks cannot be completed within this limit, we also report the average recall distribution for each strategy in Figure 3(b).

Again, the two leftmost boxes in both figures represent the results for the *Accumulative* strategies. Here, *±AccAdd* fares better than *±AccRep*. With *±AccRep* replacing items in growing positive and negative sets, the model can jump to many different directions, including back to areas already explored. This can occur when weak examples are repeatedly replaced with different weak examples. As the variety of content is much larger in the VBS2020 collection than LSC2019, this is more likely to happen.

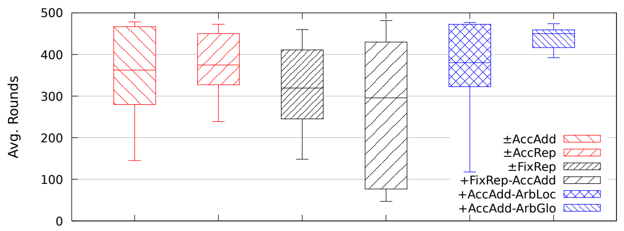
The middle two boxes show the results for the *Fixed Positive* strategies. For VBS2020, *+FixRep-AccAdd* is the best strategy for majority of the tasks in terms of rounds and recall. *±FixRep* also shows great improvement over the other strategies, and has a more consistent distribution than *+FixRep-AccAdd* in terms of recall. The reason why these strategies do not fall into the same trap as *±AccRep*, is because the positive set is limited, which allows for better control over the model's direction. Even with *±FixRep* replacing negatives, eventually only strong negative examples will remain.

The two rightmost boxes show the *Arbitrary Negative* strategies where the performance resembles that of LSC2019. *+AccAdd-ArbLoc* is again close to the performance of *±AccAdd* as shown in the figure, and when tested with the *+FixRep* strategy (not shown) it resembles the performance of *+FixRep-AccAdd*. Based on these finding there is definitely some merit to letting the system choose arbitrary negatives from the suggestion set.

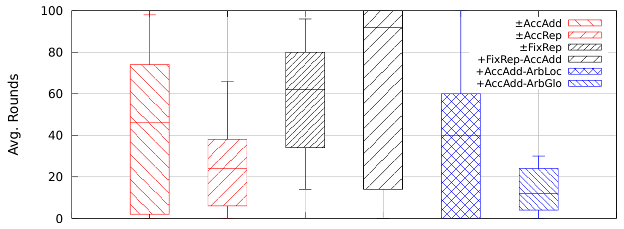
The objective of each task is to find any segment related to an event in a video, and as mentioned in Section 4.2 the task descriptions emphasize visual features of the ground truth items. The majority of the VBS2020 tasks have ground truth with a mixture of common and distinct visual features, similar to the outlier task 8 for LSC2019. As described above, many positives hurt the classification model with such tasks, as the positive set includes many items with strong common features, which drive the model away from the ground truth items with distinctive features. Policies with few positive examples are better candidates in this case. The prevalence of this type of tasks is the main reason for the strong performance of the *±FixRep* and *+FixRep-AccAdd* strategies for VBS2020.

5.3 VOPE-8hr

The results are significantly different for the VOPE-8hr collection, as the focus of tasks is to find *all* relevant items. Figure 4(a) shows the number of average rounds with each labeling strategy for each task. The bottom points of each line depict the average round when the first relevant item was encountered and the upper point is the average rounds it took find all relevant items and complete the task.



(a) Avg. rounds per run



(b) Avg. recall per run

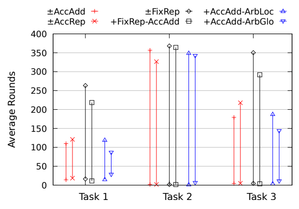
Figure 3: Baseline results for VBS2020. (a) shows the number of avg. rounds it takes to complete the tasks for each labeling strategy and (b) shows the average recall of them.

The red lines show the *Accumulative* strategies, where there are no significant differences between them. The black lines depict the *Fixed Positive* strategies, which for these tasks perform the worst. As the intention of the tasks is to find all relevant items, the number of examples in the positive set may be too small to define a satisfactory model. The blue lines show the *Arbitrary Negative* strategies. Here, we observe that *+AccAdd-ArbGlo* does surprisingly well for all 3 tasks. This is due to the presence of "red-herring" data along with noise in the collection, making it difficult to select good negatives from the local suggestion set. Task 2 takes the longest, as it has 14 times as many relevant items as the other two tasks.

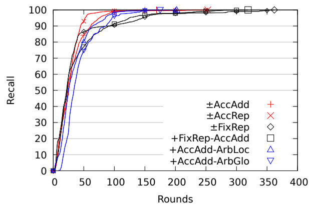
If the objective was to find the first relevant item, all strategies are efficient. This is more clear when looking at the recall over rounds for each task, depicted in Figures 4(b)-(d) for each task respectively. For task 1, all strategies find more than 80% of the relevant items in fewer than 50 rounds but struggle with the remaining 20%. While most strategies follow a similar pattern for tasks 2 and 3, the *Fixed Positive* strategies finds the majority at a slower rate but complete the task at the same time as the others. This behavior can be related to the "red herring" data as the other strategies add far more of those into their positive set which leads their models quicker to the relevant search space. Ultimately, it is worth considering that when such a scenario occurs where the user goes many rounds without discovering a relevant item, it could indicate to the system that it should guide the user to switching strategies.

5.4 Analysis of Replacements

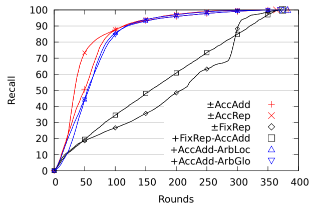
Since some strategies allow replacements, it is interesting to know when those replacements occur. Figure 5(a) shows the replacement occurrences for *±AccRep*, *±FixRep* and *+FixRep-AccAdd* for LSC2019.



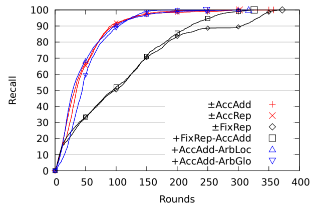
(a)



(b)

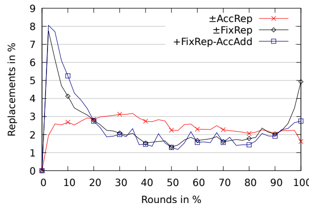


(c)

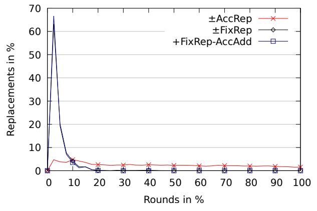


(d)

Figure 4: Results from baseline labeling strategy experiments for VOPE-8hr. (a) shows the number of avg. rounds for each labeling strategies and task. The bottom point indicates the avg. round the first relevant item was discovered, while the top depicts the avg. rounds it took to complete the task. (b), (c) and (d) shows the average recall over rounds for each task respectively.



(a) LSC2019



(b) VOPE-8hr

Figure 5: Occurence of replacements in the positive set.

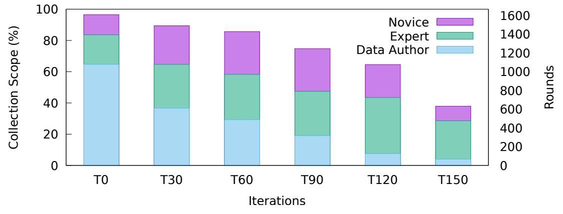


Figure 6: Average scope of the tasks from LSC2019 when their filters are applied, shown with percentage (left axis) and maximum number of interaction rounds (right axis) when the number of suggestions per round is 25.

The y-axis shows the average replacement occurrences as percentage and the x-axis shows the rounds as percentage. For the *Fixed Positive* strategies, many replacements occur early in the session. This is expected, as the model is starting to form and every round will have different suggestions. As the model becomes better, however, replacements become rare as many of the suggestions are similar and not necessarily better. For ±*AccRep* the occurrence of replacements is more balanced; as the positive set keeps increasing, the chance of replacements occurring remains similar.

The replacement patterns are similar across all collections. The replacement pattern for VBS2020 (not shown) is nearly identical to LSC2019. Figure 5(b) shows the replacement occurrences for VOPE-8hr which has a similar pattern but the trend of the fixed strategies is far more apparent, where both strategies stop replacing items after roughly 20% into the session. This is because the “red-herring” data in the collection is visually similar to the relevant items, helping the system to rapidly find optimal positive examples.

5.5 Summary

From the labeling strategy experiments we learn that different strategies can be beneficial depending on the collections content and size, and the nature of the tasks. This is a clear indication that the current evaluation methods that use strategies resembling +*AccAdd-ArbGlo* are not good enough to indicate the quality of URF systems. In addition to this revelation, the results contradict the assumption that more positive/negative labeled examples each round always lead to faster convergence.

6 EXPERIMENT 2: FILTERING STRATEGIES

We have observed how different labeling strategies can impact the number of rounds it takes to solve tasks for the different datasets. On average the VBS2020 and VOPE-8hr take roughly 230 rounds to complete a task, which tranlates to 75 minutes assuming the user spends an average of 20 seconds judging examples per interaction round. LSC2019 tasks fare better with average tasks for the strongest strategies taking fewer than 75 rounds (25 minutes). However, considering the actual time to complete the tasks in the live search challenges is 5-7 minutes, this is too long. In this experiment we run the best labeling strategies with the different filtering strategies described in Section 3.2 on LSC2019 and VBS2020. As previously mentioned VOPE-8hr does not contain metadata that can act as filters and therefore experiments for it have been omitted.

6.1 LSC2019

We start by analyzing the potential impact of filters. Figure 6 shows the percentage of search space when filters are applied by the 3 different user types for each iteration of the tasks in LSC2019. Furthermore, it depicts the worst case number of rounds it will take to find the relevant items on the right axis.

Overall this indicates the possibility of faster retrieval with the scope being reduced by more than 60% when all filters are applied for the *Novice* user. Additionally the relation between type of user and scope is clear and shows that in the worst case *Data Author* need much fewer rounds than the *Expert*, while the difference between *Expert* and *Novice* is slightly smaller. Note that the *Novice* users apply filters that exclude the relevant items for four tasks, meaning

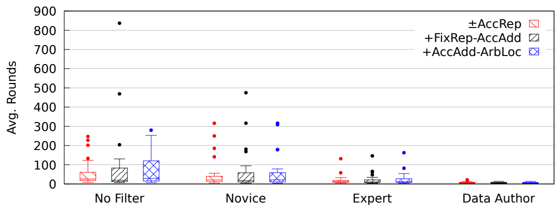


Figure 7: Avg. rounds to complete tasks with filters using the best labeling strategies for LSC2019.

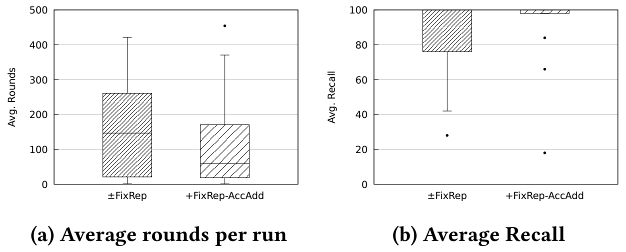


Figure 8: VBS2020 results for the *Fixed* strategies using *Expert* filters.

that the retrieval process either stops by finding the relevant item in an iteration prior to the one where the excluding filters are applied or they run out of items once they are applied.

Turning to the actual impact of filters, Figure 7 shows the results for actors using the best strategy from each category of the baseline experiments, *±AccRep*, *+FixRep-AccAdd* and *+AccAdd-ArbLoc*, with regards to average number of rounds per run to complete the task when the different filtering strategies are applied. The leftmost group of boxes represents the same results from Figure 2(a) where no filters were applied. The next group is the results from running the filters applied by the *Novice* which sees each strategy taking fewer than 70 rounds in average across tasks. The results include the four failure tasks which are shown as outliers.[3] The third group of boxes represent the results where the filters are applied by an *Expert*. Again, this shows great improvement for all strategies with average rounds being between 20-30 for all strategies. The final group of boxes show the result for the *Data Author* filters, which brings all strategies below 10 rounds. Overall the trend is expected, as users with better domain knowledge apply better filters and avoid exclusion issues. As a final note, we observe no change in the relative performance between labeling strategies when filters are applied: *±AccRep* and *+FixRep-AccAdd* remain the strongest.

6.2 VBS2020

For VBS2020, the *Novice* selects most filters from tags and categories which end up excluding the correct video segment for most tasks. In fact there are only two tasks where it manages to not exclude them and manages to find the desired segment in fewer than 20 rounds. For the remaining 11 tasks, however, it fails to complete any of them with the filters applied. We therefore do not consider the *Novice* user further.

Next we study the impact of actors using the filtering strategy of an *Expert* user that has worked with the collection and understands when and how to set frame level filters and video level filters. Figure 8(a) highlights the results of these actors using the *Fixed Positive* strategies, which were the best overall from the labeling strategy experiments. None of the filters set by the actor exclude any relevant items and we see a great improvement in terms of average rounds per run for both strategies with *+FixRep-AccAdd* still being the best. Figure 8(b) shows the avg. recall per run for the actors. While both labeling strategies improve in this area as well, the *+FixRep-AccAdd* is far more consistent with the average recall close to 100% for the majority of the tasks, which further solidifies it as a preferred strategy. However, there are still tasks where some of the runs fail to complete. This means that the model either got derailed and exceeded the number of rounds or that even with the filters applied the search scope is still too large.

[3]Note that *±AccRep* does complete up to 11 of its 50 runs for 1 of those tasks. This is due to the excluding filter being set later for this task than the other 3, making it possible for the task to be completed.

6.3 Summary

Overall we have shown that applying filters is beneficial for all types of users if the collection is well curated and the task descriptions reflect the metadata used as filters. It can have negative consequences if the user has little domain knowledge, especially when they set extreme filters that exclude the relevant items. However, as filters are set over time and the excluding feature is not set at the beginning, URF is occasionally fast enough to bypass the excluding filter by finding a relevant item before that filter is set. Furthermore, our results firmly indicate that URF with filters applied by users with high domain knowledge is always better than just applying filters.

7 CONCLUSION

In this paper, we have analyzed the impact of interaction strategies for labeling positives and negatives as well as applying filters based on user's domain knowledge for user relevance feedback systems. By conducting experiments on three different collections of various sizes and tasks using artificial users, we observe that the choice of labeling strategy can have a major impact on number of interaction rounds it takes to finish a task. There is no single optimal labeling strategy, as the best strategy depends on both the collection and the task. Furthermore, our results refute the common assumption of providing more training examples is always beneficial, as strategies with smaller set of examples lead to better results in some cases.

We observe that users with expert level or higher domain knowledge unsurprisingly apply filters that are beneficial. However, aggressive filtering, especially by novice users, can hinder the completion of tasks. Furthermore, URF is a powerful tool in conjunction with filters that leads to better results than using filters alone.

These findings should be considered in future URF evaluation efforts as more refined artificial users will lead to better benchmarks, making it easier to quantify the performance of URF systems.

ACKNOWLEDGMENTS

This work was supported by a PhD grant from the IT University of Copenhagen and by the European Regional Development Fund (project Robotics for Industry 4.0, CZ.02.1.01/0.0/0.0/15 003/0000470).

REFERENCES

[1] IJsbrand Jan Aalbersberg. 1992. Incremental Relevance Feedback. In *Proceedings of the 15th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval* (Copenhagen, Denmark) *(SIGIR '92)*. Association for Computing Machinery, New York, NY, USA, 11–22.

[2] James Allan. 1996. Incremental Relevance Feedback for Information Filtering. In *Proceedings of the 19th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval* (Zurich, Switzerland) *(SIGIR '96)*. Association for Computing Machinery, New York, NY, USA, 270–278.

[3] Fabian Berns, Luca Rossetto, Klaus Schoeffmann, Christian Beecks, and George Awad. 2019. V3C1 Dataset: An Evaluation of Content Characteristics. In *Proceedings of the 2019 on International Conference on Multimedia Retrieval* (Ottawa ON, Canada) *(ICMR '19)*. Association for Computing Machinery, New York, NY, USA, 334–338.

[4] Bogdan Boteanu, Ionuţ Mironică, and Bogdan Ionescu. 2017. Pseudo-relevance feedback diversification of social image retrieval results. *Multimedia Tools and Applications* 76, 9 (2017), 11889–11916.

[5] François Chollet. 2017. Xception: Deep Learning with Depthwise Separable Convolutions. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, Honolulu, HI, USA, 1800–1807.

[6] Duc-Tien Dang-Nguyen, Luca Piras, Giorgio Giacinto, Giulia Boato, and Francesco GB DE Natale. 2017. Multimodal retrieval with diversification and relevance feedback for tourist attraction images. *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)* 13, 4 (2017), 1–24.

[7] Lianli Gao, Jingkuan Song, Fuhao Zou, Dongxiang Zhang, and Jie Shao. 2015. Scalable Multimedia Retrieval by Deep Learning Hashing with Relative Similarity Learning. In *Proceedings of the 23rd ACM International Conference on Multimedia* (Brisbane, Australia) *(MM'15)*. Association for Computing Machinery, New York, NY, USA, 903–906.

[8] Ralph Gasser, Luca Rossetto, and Heiko Schuldt. 2019. Multimodal Multimedia Retrieval with Vitrivr. In *Proceedings of the 2019 on International Conference on Multimedia Retrieval* (Ottawa ON, Canada) *(ICMR '19)*. Association for Computing Machinery, New York, NY, USA, 391–394.

[9] Paula Gómez Duran, Eva Mohedano, Kevin McGuinness, Xavier Giró-i Nieto, and Noel E. O'Connor. 2018. Demonstration of an Open Source Framework for Qualitative Evaluation of CBIR Systems. In *Proceedings of the 26th ACM International Conference on Multimedia* (Seoul, Republic of Korea) *(MM'18)*. Association for Computing Machinery, New York, NY, USA, 1256–1257.

[10] Cathal Gurrin, Klaus Schoeffmann, Hideo Joho, Andreas Leibetseder, Liting Zhou, Aaron Duane, Dang Nguyen, Duc Tien, Michael Riegler, Luca Piras, et al. 2019. Comparing approaches to interactive lifelog search at the lifelog search challenge (LSC2018). *ITE Transactions on Media Technology and Applications* 7, 2 (2019), 46–59.

[11] Xiaofei He, Oliver King, Wei-Ying Ma, Mingjing Li, and Hong-Jiang Zhang. 2003. Learning a semantic space from user's relevance feedback for image retrieval. *IEEE transactions on Circuits and Systems for Video technology* 13, 1 (2003), 39–48.

[12] Thomas S Huang, Charlie K Dagli, Shyamsundar Rajaram, Edward Y Chang, Michael I Mandel, Graham E Poliner, and Daniel PW Ellis. 2008. Active learning for interactive multimedia retrieval. *Proc. IEEE* 96, 4 (2008), 648–667.

[13] Herve Jegou, Matthijs Douze, and Cordelia Schmid. 2010. Product quantization for nearest neighbor search. *IEEE transactions on pattern analysis and machine intelligence* 33, 1 (2010), 117–128.

[14] Lu Jiang, Teruko Mitamura, Shoou-I Yu, and Alexander G. Hauptmann. 2014. Zero-Example Event Search Using MultiModal Pseudo Relevance Feedback. In *Proceedings of International Conference on Multimedia Retrieval* (Glasgow, United Kingdom) *(ICMR '14)*. Association for Computing Machinery, New York, NY, USA, 297–304.

[15] Björn Þór Jónsson, Omar Shahbaz Khan, Dennis C. Koelma, Stevan Rudinac, Marcel Worring, and Jan Zahálka. 2020. Exquisitor at the Video Browser Showdown 2020. In *MultiMedia Modeling*, Yong Man Ro, Wen-Huang Cheng, Junmo Kim, Wei-Ta Chu, Peng Cui, Jung-Woo Choi, Min-Chun Hu, and Wesley De Neve (Eds.). Springer International Publishing, Cham, 796–802.

[16] Omar Shahbaz Khan, Björn Þór Jónsson, Stevan Rudinac, Jan Zahálka, Hanna Ragnarsdóttir, Þórhildur Þorleiksdóttir, Gylfi Þór Guðmundsson, Laurent Amsaleg, and Marcel Worring. 2020. Interactive Learning for Multimedia at Large. In *European Conference on Information Retrieval*. Springer, Cham, 495–510.

[17] Omar Shahbaz Khan, Mathias Dybkjær Larsen, Liam Alex Sonto Poulsen, Björn Þór Jónsson, Jan Zahálka, Stevan Rudinac, Dennis Koelma, and Marcel Worring. 2020. Exquisitor at the Lifelog Search Challenge 2020. In *Proceedings of the Third Annual Workshop on Lifelog Search Challenge* (Dublin, Ireland) *(LSC '20)*. Association for Computing Machinery, New York, NY, USA, 19–22.

[18] Miroslav Kratochvil, František Mejzlík, Patrik Veselý, Tomáš Soućek, and Jakub Lokoć. 2020. *SOMHunter: Lightweight Video Search System with SOM-Guided Relevance Feedback*. Association for Computing Machinery, New York, NY, USA, 4481–4484.

[19] Miroslav Kratochvíl, Patrik Veselý, František Mejzlík, and Jakub Lokoč. 2020. SOM-Hunter: Video Browsing with Relevance-to-SOM Feedback Loop. In *MultiMedia Modeling*. Springer International Publishing, Cham, 790–795.

[20] Brian Kulis and Kristen Grauman. 2009. Kernelized Locality-Sensitive Hashing for Scalable Image Search. In *2009 IEEE 12th International Conference on Computer Vision*. IEEE Computer Society, Los Alamitos, CA, USA, 2130 – 2137.

[21] František Mejzlík, Patrik Veselý, Miroslav Kratochvíl, Tomáš Souček, and Jakub Lokoč. 2020. SOMHunter for Lifelog Search. In *Proceedings of the Third Annual Workshop on Lifelog Search Challenge* (Dublin, Ireland) *(LSC '20)*. Association for Computing Machinery, New York, NY, USA, 73–75.

[22] Pascal Mettes, Dennis C. Koelma, and Cees G.M. Snoek. 2016. The ImageNet Shuffle: Reorganized Pre-Training for Video Event Detection. In *Proceedings of the 2016 ACM on International Conference on Multimedia Retrieval (ICMR '16)*. Association for Computing Machinery, New York, NY, USA, 175–182.

[23] G. L. J. Pingen, M. H. T. de Boer, and R. B. N. Aly. 2017. Rocchio-Based Relevance Feedback in Video Event Retrieval. In *MultiMedia Modeling*. Springer International Publishing, Cham, 318–330.

[24] J. J. Rocchio. 1965. *Relevance feedback in information retrieval.* Technical Report. University of Harvard, Computer Laboratory.

[25] Ork De Rooij and Marcel Worring. 2012. Efficient Targeted Search Using a Focus and Context Video Browser. *ACM Trans. Multimedia Comput. Commun. Appl.* 8, 4, Article 51 (Nov. 2012), 19 pages.

[26] Yong Rui, Thomas S Huang, and Sharad Mehrotra. 1997. Content-based image retrieval with relevance feedback in MARS. In *Proceedings of International Conference on Image Processing*, Vol. 2. IEEE, IEEE, Santa Barbara, CA, USA, 815–818.

[27] Sheikh Muhammad Sarwar, John Foley, and James Allan. 2018. Term Relevance Feedback for Contextual Named Entity Retrieval. In *Proceedings of the 2018 Conference on Human Information Interaction & Retrieval* (New Brunswick, NJ, USA) *(CHIIR '18)*. Association for Computing Machinery, New York, NY, USA, 301–304.

[28] Klaus Schoeffmann. 2014. A User-Centric Media Retrieval Competition: The Video Browser Showdown 2012-2014. *IEEE MM* 21, 4 (2014), 8–13.

[29] Roberto Tronci, Gabriele Murgia, Maurizio Pili, Luca Piras, and Giorgio Giacinto. 2013. *ImageHunter: A Novel Tool for Relevance Feedback in Content Based Image Retrieval.* Springer Berlin Heidelberg, Berlin, Heidelberg, 53–70.

[30] VOX-Pol. 2021. About Us. https://www.voxpol.eu/about-us

[31] Xuanhui Wang, Hui Fang, and ChengXiang Zhai. 2008. A Study of Methods for Negative Relevance Feedback. In *Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval* (Singapore, Singapore) *(SIGIR '08)*. Association for Computing Machinery, New York, NY, USA, 219–226.

[32] Yunchao Wei, Yao Zhao, Canyi Lu, Shikui Wei, Luoqi Liu, Zhenfeng Zhu, and Shuicheng Yan. 2016. Cross-modal retrieval with CNN visual features: A new baseline. *IEEE transactions on cybernetics* 47, 2 (2016), 449–460.

[33] Rong Yan, Alexander Hauptmann, and Rong Jin. 2003. Multimedia Search with Pseudo-relevance Feedback. In *Image and Video Retrieval*. Springer Berlin Heidelberg, Berlin, Heidelberg, 238–247.

[34] Rong Yan, Alexander G. Hauptmann, and Rong Jin. 2003. Negative Pseudo-Relevance Feedback in Content-Based Video Retrieval. In *Proceedings of the Eleventh ACM International Conference on Multimedia* (Berkeley, CA, USA) *(MULTIMEDIA '03)*. Association for Computing Machinery, New York, NY, USA, 343–346.

[35] Yi Yang, Feiping Nie, Dong Xu, Jiebo Luo, Yueting Zhuang, and Yunhe Pan. 2011. A multimedia retrieval framework based on semi-supervised ranking and relevance feedback. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 34, 4 (2011), 723–742.

[36] Jan Zahálka, Stevan Rudinac, Björn Þór Jónsson, Dennis C Koelma, and Marcel Worring. 2018. Blackthorn: Large-Scale Interactive Multimodal Learning. *IEEE Transactions on Multimedia* 20, 3 (2018), 687–698.

[37] Jan Zahálka, Stevan Rudinac, Björn Þór Jónsson, Dennis C. Koelma, and Marcel Worring. 2016. Interactive Multimodal Learning on 100 Million Images. In *Proceedings of the 2016 ACM on International Conference on Multimedia Retrieval* (New York, New York, USA) *(ICMR '16)*. Association for Computing Machinery, New York, NY, USA, 333–337.

[38] Jan Zahálka, Stevan Rudinac, and Marcel Worring. 2015. Analytic Quality: Evaluation of Performance and Insight in Multimedia Collection Analysis. In *Proceedings of the 23rd ACM International Conference on Multimedia* (Brisbane, Australia) *(MM '15)*. Association for Computing Machinery, New York, NY, USA, 231–240.

[39] Jan Zahálka, Marcel Worring, and Jarke J. Van Wijk. 2021. II-20: Intelligent and pragmatic analytic categorization of image collections. *IEEE Transactions on Visualization and Computer Graphics* 27, 2 (2021), 422–431.

[40] Shifeng Zhang, Jianmin Li, Mengqing Jiang, Peijiang Yuan, and Bo Zhang. 2017. Scalable discrete supervised multimedia hash learning with clustering. *IEEE Transactions on Circuits and Systems for Video Technology* 28, 10 (2017), 2716–2729.