Technische Universität Chemnitz at TRECVID Instance Search 2016

Stefan Kahl[1], Christian Roschke[1], Markus Rickert[1], Daniel Richter[2], Anna Zywietz[2], Hussein Hussein[1], Robert Manthey[1], Manuel Heinzig[1], Danny Kowerko[1], Maximilian Eibl[2], and Marc Ritter[1,3]

[1]Junior Professorship Media Computing, Technische Universität Chemnitz, D-09107 Chemnitz, Germany
[2]Chair Media Informatics, Technische Universität Chemnitz, D-09107 Chemnitz, Germany
[3]Professorship Media Informatics, University of Applied Sciences Mittweida, D-09648 Mittweida, Germany

Abstract. This contribution presents our third appearance at the TRECVID *Instance Search* (INS) task (Awad et al., 2016; Smeaton et al., 2006). We participated in the evaluation campaign with four runs (two interactive and two automatic) using video-based visual concepts. A combination of different methods is used in every run. Our basic approach is based on probabilistic assumptions about appearance-based methods in combination with semantic context analysis. A deep learning convolutional neural network (CNN) combines the recognition of persons and filming locations. In addition, we extended the sequence clustering method from our previous contribution that incorporates visual similarity calculations between all corresponding shots in the omnibus episodes provided. Throughout all approaches, again we make use of our adaptable and easy-to-use keyframe extraction scheme (Ritter et al., 2014, 2015). Furthermore, we created a web-based interactive platform in order to optimize our workflow and enhance our speed in the interactive part of the search task.

1 Structured Abstract

1. *Briefly, list all the different sources of training data used in the creation of your system and its components.*

- For training issues, we solely used the given master shot reference, and the video only tracks of the first video with (ID 0 also denoted as Dev0, D0 in this contribution) from the provided *BBC EastEnders* video footage as well as the location video examples.

2. *Briefly, what approach or combination of approaches did you test in each of your submitted runs?*

- Our first automatic run F_A_TUC_4 builds the baseline for our deep learning networks for person and filming location recognition on around 1.1 million extracted keyframes.
- The second automatic run F_A_TUC_3 combines our approach to partially semantic sequence clustering (SC) with deep learning for person and location recognition.
- Within the first interactive run I_A_TUC_2, we use our web-based interactive platform for evaluation as a post-processing step on our deep learning technology for person and location recognition.
- Our second interactive run I_A_TUC_1 combines our semantic sequence clustering (SC) and person and location recognition on deep learning together with interactive post-processing.

3. *What if any significant differences (in terms of what measures) did you find among the runs?*

- As expected, and in terms of MAP, there is a significant difference of 8% from the worst interactive over the best fully automatic run.
- According to Precision at rank 15 (P15), our two runs with sequence clustering (SC) are performing about 10% better then the other two runs without it.

4. *Based on the results, can you estimate the relative contribution of each component of your system/approach to its effectiveness?*

- Our convolutional neural networks (CNNs) perform the main work.
- Sequence clustering (SC) is able to improve the performance, when the CNNs do well at the beginning, but miss some corresponding shots in a sequence. When the CNNs fail within the highest ranked results, SC emphasizes this negative effect. According to this years

Correspondence to: Marc Ritter
marc.ritter@hs-mittweida.de

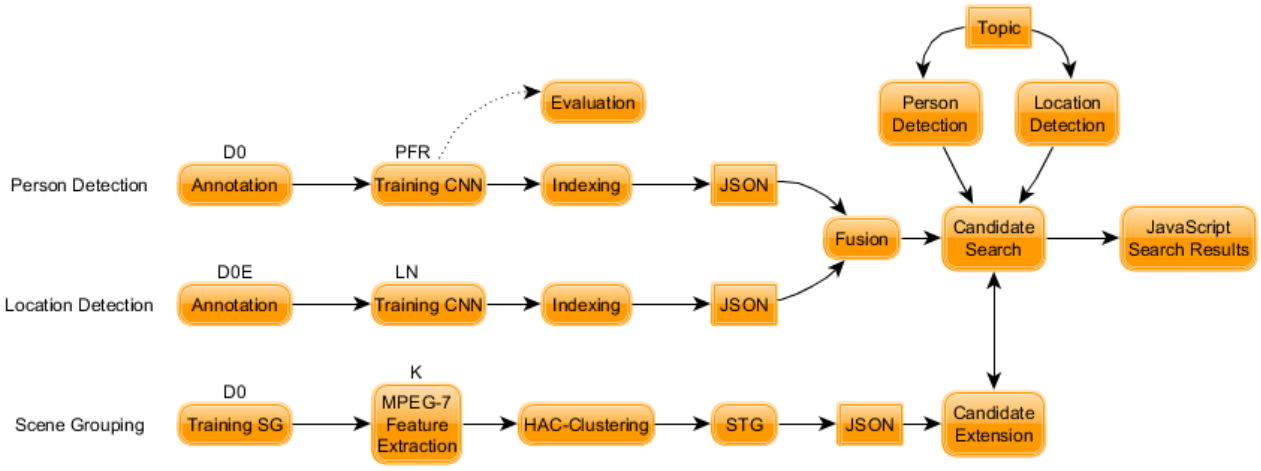

Figure 1. Architecture of our approach to person and location recognition. A detailed explanation is given in subsection 2.2.

evaluation, SC improves the results of about two third of all topics and leads to a better MAP in general.

- The web-based interactive platform for evaluation is able to improve most results in a significant way. Again, if the CNNs fail, no improvement can be recognized.

5. *Overall, what did you learn about runs/approaches and the research question(s) that motivated them?*

- Machine learning trivially can recognize people and locations.
- Even on consumer-hardware, one can get a sufficient performance with open source tools and frameworks in the machine learning domain that have already been made available for the community.
- Sequence clustering seems to be an usable heuristic for finding missed instances in the direct or indirect neighborhood of already detected samples.

The remainder of the paper is organized as follows: Section 2 provides a general view about the basic concepts and more common components of our system architecture and the underlying workflow for both run types. The specific algorithms being used by the system are described in section 3. Remarks regarding the official evaluation results are given in section 4 while being followed by some conclusions in section 5.

2 System Architecture

The architecture of our approach to this year's Instance Search Task consists of a number of relevant components that build the base for the automated methods and interactive components shown in the next sections. In order to decrease the amount of data that is to be processed, we rely on our established concept of data reduction by dynamic shot-length-based keyframe extraction in section 2.1. Section 2.2 discusses the infrastructure of our deep learning methods that enable us to recognize persons and locations while also taking higher-level semantic similarities of temporally co-located scenes into account. Finally in section 2.3, we introduce a web-based platform that allows for a comfortable intellectual annotation and evaluation of the interactive parts of the evaluation campaign.

2.1 Preprocessing and Keyframe Extraction

As the corpus for this task didn't change since several years, we mostly used our already built collection, described in our reports from the previous TRECVID evaluation campaigns (Ritter et al., 2014, 2015). The collection consists of 1.15 million keyframes in a resolution of 928×512 pixels, that were extracted from the video collection from the *BBC EastEnders* series consisting of about 42 million frames. The same preprocessing steps of squared pixel stretching and border cropping were applied to this year's topic examples, i.e. the pictures and corresponding masks, in order to fit our collection and for simplified and compliant processing by our system.

2.2 Deep Learning & Sequence Clustering

The workflow of our approach to deep learning is shown in Figure 1. Therein, each appearance of the 28 main characters in the Dev0 episode (D0) is annotated. Locations are annotated based on the Dev0 episode and additional given example locations (D0E). Both data sets are used to train convolutional neural networks with *Py-Faster-RCNN* (PFR) and *Lasagne/Nolearn* (LN) to create indexes of all episodes. The quality of the training can be monitored with a web-based interface where training parameters can be modified and evaluation can be achieved. After the creation of the indexes, the results are being transformed into plain JSON-files for further fusion.

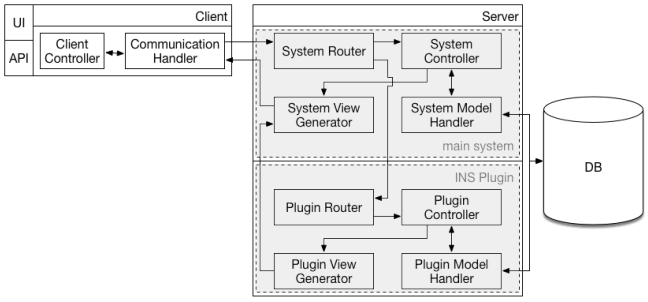

Figure 2. Schematically representation of system architecture of WIPE

This years approach to sequence clustering algorithm is an advancement of our last year's solution (Ritter et al., 2015). The scene grouping is trained on D0, too. After the extraction of MPEG-7-features via the LIRE library (Lux and Chatzichristofis, 2008) from the keyframes of the test set, an hierarchical agglomerative clustering (HAC-Clustering) is performed and a scene transition graph (STG) is computed. Resulting similarity group mappings are extracted as JSON-files and combined with previous results.

The computation of the queries is performed automatically by the analysis of persons and locations in the given topic examples. Those queries are used to find a set of candidate frames with a simple Java Script search engine.

2.3 Web-based interactive platform for evaluation

As part of the TRECVid INS task, we developed a web-based interactive platform for evaluation (WIPE). We focused on the design of a stable, scalable and reusable platform to integrate services which we need for but which are not limited to INS. Our platform is based on the Laravel framework and languages like PHP, JavaScript, HTML and CSS. Laravel supports a standardised development workflow by automatically processing some logic relationships. Hence, the developer gains the opportunity to focus on the implementation of business logic.

2.3.1 Main system

The architecture of the main system displayed in Figure 2 is based on the client-server principle. An UI or an API allows interactions with the *Client Controller*. The controller forwards inquiries to the *Communication Handler*, which is the interface of the server application. This handler allows to send and receive messages through AJAX, direct connections, server-sent events and WebSockets.

The server consists of a main system and several addable plug-ins, like the INS Plugin. The main system contains the complete management logic, encompassing methods to administrate users, to protect application ranges, to generate views and to integrate plug-ins. The use of plug-ins allows to extend the functional core without the risk of destroying

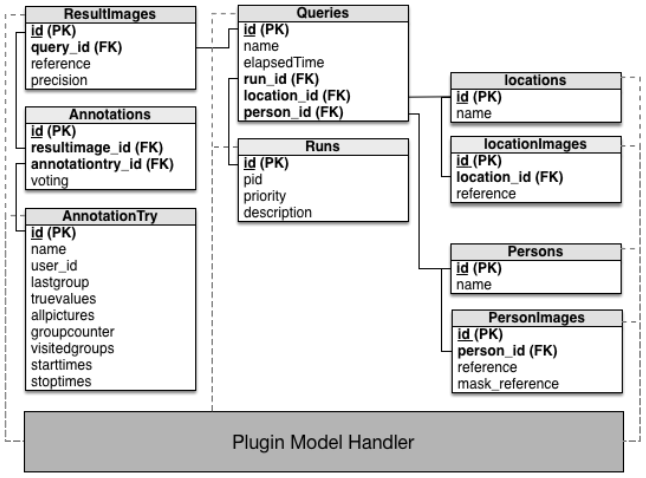

Figure 3. INS Plugin database structure.

the core logic. The *System Router* of the main system accepts all inquiries which were sent by the client and passes them onto the *System Controller* or to instances of *Plugin Router*. The *System Controller* contains the application logic and generates data structures by using models of the *System Model Handler*. This data structures are made available to the *System View Generator*. The *System Model Handler* is the interface between the application logic and the database. A model represents each table and provides an access object to interact with the database for the application logic. The *System View Generator* uses the data structures created in the controller and transforms them into a standardised exchange format like HTML, JSON, or XML. The generated formats will be transmitted as a response to the *Communication Handler* of the HTTP client.

Instead of passing the requests to the *System Controller* of the main system, they can also be relayed into a plugin. Like in the main system, every plugin consists of a *Router*, a *Model Handler*, a *View Generator*, and a *Controller*. Each request will be forwarded by the *Plugin Router* to the *Plugin Controller*. The *Plugin View Generator* converts data structures, which were created by the *Plugin Controller* by using the *Plugin Model Handler* and integrates them into the view of the *System View Generator*.

2.3.2 INS Plugin

To post-process the automatically generated result candidates for the INS task on behalf of the interactive runs, we have expanded the main system with additional functionalities by means of the INS Plugin. In order to conduct a solid study in the interactive part, we store every decision in a database and make use of the created scheme shown in Figure 3. Since we decided to evaluate the two automatic runs, we transferred the associated data into the tables ResultImages, Queries, Runs, Locations, LocationImages, Persons and PersonImages.

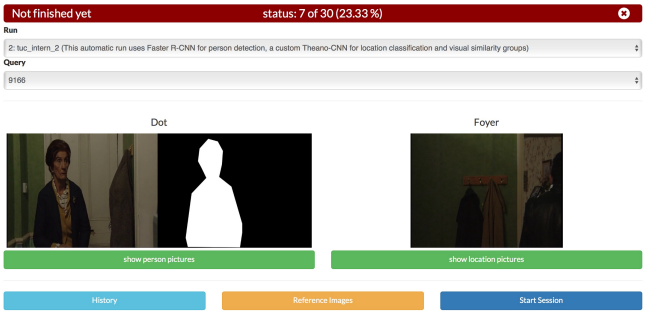

Figure 4. Screenshot of the INS Plugin OverviewUI

Figure 5. Screenshot of the INS Plugin AnnotationUI

The Run table consists of the properties ProcessID, Priority and Description. The associated queries for each run contain the properties Name and ElapsedTime. A query is further characterized by a location and a person, while appropriate tables are mapped directly. The LocationImages table and the PersonImages table store the references of the locations and the persons. Result images generated from the automatic runs are represented in the table ResultImages. In addition to the reference of a result image the precision will be stored aswell. The data and decisions of the annotation process are saved in the tables Annotations and AnnotationTry. The table Annotations only contains the voting as a boolean and the id of the result image. AnnotationTry consists of all metadata of the annotation process. In order to evaluate an image multiple times, the table Annotation is mapped to the table AnnotationTry. Each database table is coupled to the plugin model handler and can be approached by plugin controllers.

To realize the required business logic, we developed two types of plugin controllers, the Frontendcontroller and the Backendcontroller. By means of the Backendcontroller, it is possible to import metadata into the database and to export annotation results as XML. For the integration of the data, we developed two XML schemas, which were parsed in the system and transferred to the database. A new Plugin View Generator creates multiple UIs, to ensure that users can interact. The Frontendcontroller transfers relevant data for the annotation process to the Plugin View Generator. We created two different graphical user interfaces in order to realize a user interaction with the Frontendcontroller.

The OverviewUI (see Figure 4) displays information about the current annotation status of a user including how many queries of a run are finished. Furthermore, a user is able to select a run and an associated query. Here, reference pictures of persons and locations can be reviewed. A history comprises information about the date of the last annotations and the number of already seen and voted images. Based on this selection, the interactive annotation/evaluation process can be started. The architecture of the system allows in principle a repetition of the annotations where each attempt is stored in the history. In compliance to the rules of the instance search task, we did not make use of it.

The AnnotationUI (see Figure 5) is displayed after the start of an annotation process. Initially, a timer is created and set to 5 minutes. The start time will be transmitted to the server and stored there, to prevent cheating or a manipulation of the timer. Multiple pages are generated for the user containing each 2×3 result images of the associated query. It is generally assumed that all the images are denoted as incorrect answer to the requested query therefore being marked with a red bounding box. A user can change the status of an image by pressing the hot keys 1 to 6. A change of the status of an image alters the color to green and vice versa. We also included some functionality to visually recall the reference images of the query or to enlarge single images. After the timer expires, the UI is disabled while the transfer is being uploaded to the server.

3 Methods

We apply a variety of different methods in this year evaluation campaign. One approach is based on deep learning with convolutional neural networks for the recognition of characters from the East Enders series at specific filming locations illustrated in section 3.1. In addition, our second approach adds semantic information by introducing sequence clustering with weighted similarity measures on video segmentation presented in section 3.2.

3.1 Person & location recognition

We split this years INS task into two sub-tasks, one being the detection and recognition of specific characters from the BBC East Enders series whereas the other includes the classification of different filming locations of the show. Later on, we combined the results for each of the sub-tasks to one overall score. We made extensive use of web-based interfaces for almost our entire workflow, consisting of manual instance annotation, supervised neural network training and result evaluation.

3.1.1 Person Detection & Recognition

We decided to annotate every occurrence of all main characters from the first episode (shot0). Therefore, we extracted 20,944 frames (3 fps) from this episode and used a custom web-based bounding box annotation tool to localize and label 25,804 instances of 28 main characters. This is an average of 921 instances of each character with a minimum of 91 for "Patrick Trueman" and a maximum of 2,714 for "Stella Crawford". Although we were not able to train and detect characters other from the ones present in shot0 (which is the case for "Fatboy"), this approach proved to be sufficient.

In recent years artificial neural networks emerged as the leading technique in computer vision. Nowadays, especially convolutional neural networks (CNN) appear as a major choice for extensive image classification tasks. Ren et al. (2015) introduced *Faster R-CNN*, a region proposal network which can detect salient regions in images and being able to classify their contents in one forward pass through the net. We used a modified version of *Py-Faster-R-CNN*, built into a custom web-interface for training and person recognition. We configured the training process to require an overlap ratio of 0.6 for region proposals to be considered as part of the ground truth and ran training for 120,000 iterations with a batch size of 128 and a learning rate decay after every 50,000 iterations, image scales of 600 pixel and horizontal flip as data set augmentation. The training process took about 25 hours on a slightly overclocked Nvidia GeForce GTX 980 graphics card with 4 GB RAM.

3.1.2 Location Classification

Our approach for the recognition of filming locations is also based on a CNN and implemented in *Python* using *Theano* (Theano Development Team, 2016), *Lasagne* (Dieleman et al., 2015) and *Nolearn* (Nouri, 2014). We categorised episode shot0 into 32 different classes according to filming locations. We learned that the locations Pub and Supermarket are not a subset of shot0. Therefore, we used the given training set and merged it with our annotations automatically. The locations *Cafe 1* and *Cafe 2* are considered as one location (Cafe).

We reduced the number of training classes to 14 consisting of: 10 locations, an outdoor class containing every outdoor shot of episode shot0, samples of the introduction at the beginning of every episode, frames of commercials at the end of every episode, and other locations (every other frame, not part of the previous classes). The resulting number of frames in our training set is 21,973. We used a modified version of the *VGG-16* network layout (Simonyan and Zisserman, 2014). Additionally, we utilized dropouts after every pooling layer and fewer filters (starting with 32 in the first convolution) due to hardware limitations. A linear learning rate decay is performed after every epoch. The increase of the Nesterov momentum per epoch is also linear (ranging from

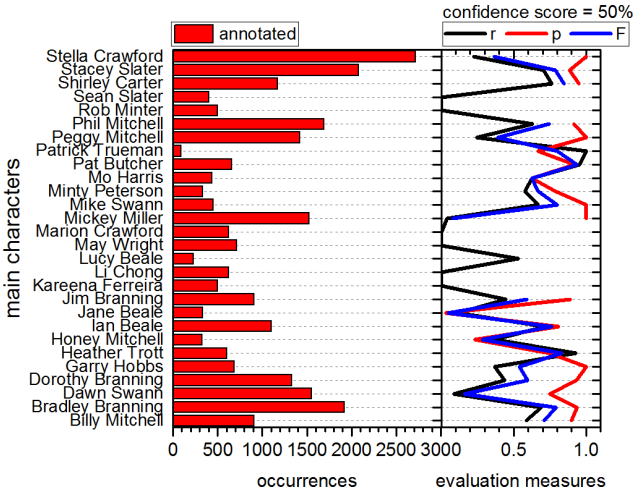

Figure 6. Person recognition at shot0 and evaluation results at shot1. Some characters had no appearance in this episode.

0.9 to 0.999). We subtracted the pixel mean and standard deviation values from every batch and performed horizontal flip and random crop for data set augmentation.

Early stopping of the training after 117 epochs prevents the net from overfitting. Our validation split is 15% of the data set, the test split consists of 5% of the data set. We achieved a best error rate of 8.4% by using the test set. The experiments were conducted on a simple workstation PC with Core i7 processor and Nvidia GeForce Titan X Maxwell graphics card with 12 GB RAM. The training time was around 77 seconds per epoch.

3.1.3 Scoring and Search

Overall, 471,562 keyframes have been analysed in our system. The calculation of a likelihood score for classifying those into locations took around 17 ms per frame or approximately 134 minutes in total. For the person detection, however, we used *Py-Faster-RCNNs* to extract 500 object proposals per frame, which took an average of 161 ms for each frame. We saved all suggestions with a confidence value greater than 0.5 as plain text and arranged them in descending order (score-summary set). Both computations were performed offline, before retrieval time and therefore are not added to the crucial search time. As a drawback of this approach, we were not able to retrieve any detections (not even similar ones) for "Fatboy", due to his absence in shot0.

Prior to the search, we automatically analysed the topic samples with respect to the person (image) or location (XML) contained for query reformulation. Scores are calculated for each of the four example images where the best scoring detection is picked to retrieve images from the previously calculated score-summary set. We performed a manual evaluation of our person detection workflow for convenience by using episode shot1 (we did this after our final training

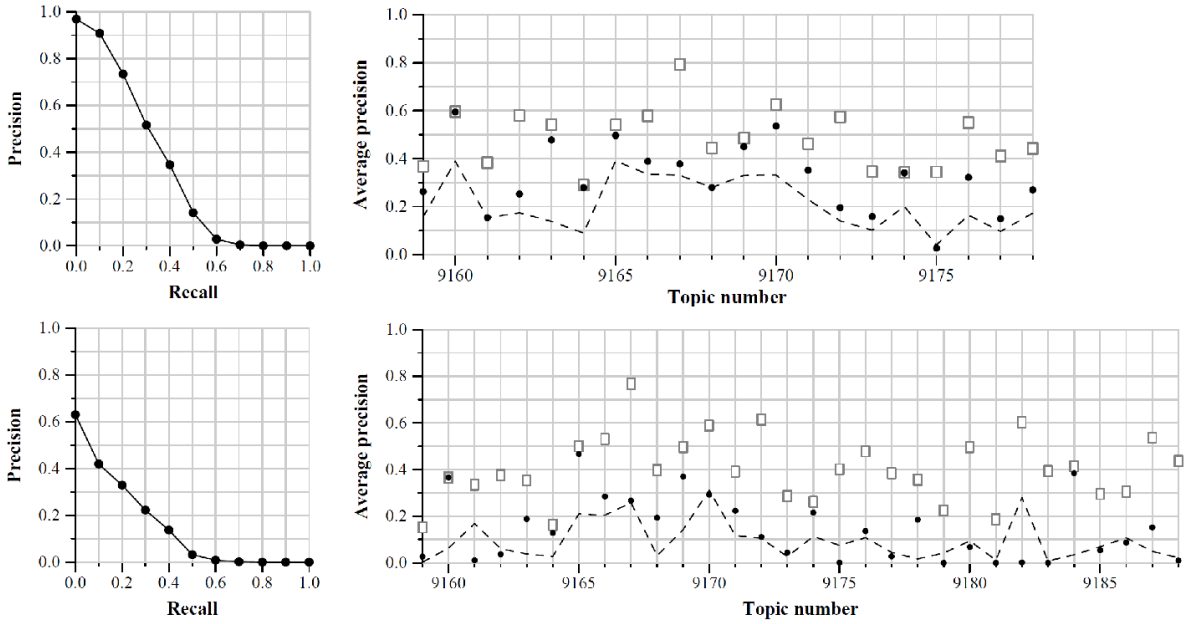

Figure 7. Evaluation result of the first and the 3rd run ("I_A_TUC_1" and "F_A_TUC_3").

optimization without any backward driven injection into the development system; therefore, this complies with the rules of the evaluation campaign). We accomplished a MAP of 0.53 for this episode, overall results of detection frequencies are shown in Figure 6.

The merged score of a single image is calculated by:

$$\frac{1}{3}\left(2 \cdot PersonConfidence + 1 \cdot LocationScore\right) \qquad (1)$$

We then combine the results of person recognition for each frame with the concept of visual similarity groups to achieve a score for the whole group, which is the average score of all frames in the respective conglomerate. Our chosen way of presenting the results is an illustrative web interface which makes it easy for the viewer to assort the outcome. We furthermore export the needed XML file. The whole process of searching takes about 1.3 seconds on a single core low-standard machine with no further optimization in Java Script.

3.2 Sequence Clustering

For the competition in 2015, we invented a sequence clustering algorithm to make use of probabilistic reasoning in our solution. It is an approach based on the methods Time-Constrained-Clustering and Scene-Transition-Graph developed by (Yeung et al., 1998). In 2016, we enhance this approach in a more sophisticated manner. Basically, sequence clustering is a technique for the segmentation of narrative audiovisual media. It uses visual feature descriptors to measure the similarity between shots. Based on the similarity the shots are aggregated into larger structures that are compatible to scenes. As a result of the algorithm, every shot of every video can be assigned to a group of visually similar shots and to a scene of continuous action.

We use this semantic data in combination with the results of the previous steps of person and location recognition to create inferences about when a certain person is present at a certain location. As the first step, we use the visual similarity groups. All shots belonging to a similarity group are supposed to depict the same camera recording in terms of time, location, and persons therein.

3.2.1 Inferences by Similarity Groups

If in a task all instances are searched where person X is present at location A, the results of person recognition and location recognition are combined to a result list. For each item of the list, we conclude by the results of the sequence clustering, that all other members of the items similarity group depict person X at Location A as well. Even if in these shots the person or the location has not been recognized. Therefore, we extend the result list by adding all members of the shots similarity group.

The Sequence Clustering (SC) approach enables us to extend our search results based on probabilistic reasoning and semantic coherence. For the calculations of the sequence clustering algorithm we use the information of the master shot reference data and the keyframes that have been extracted from the video file. The algorithm is described in detail in (Ritter et al., 2015, section 3.5) with an adjustment of the following advancements.

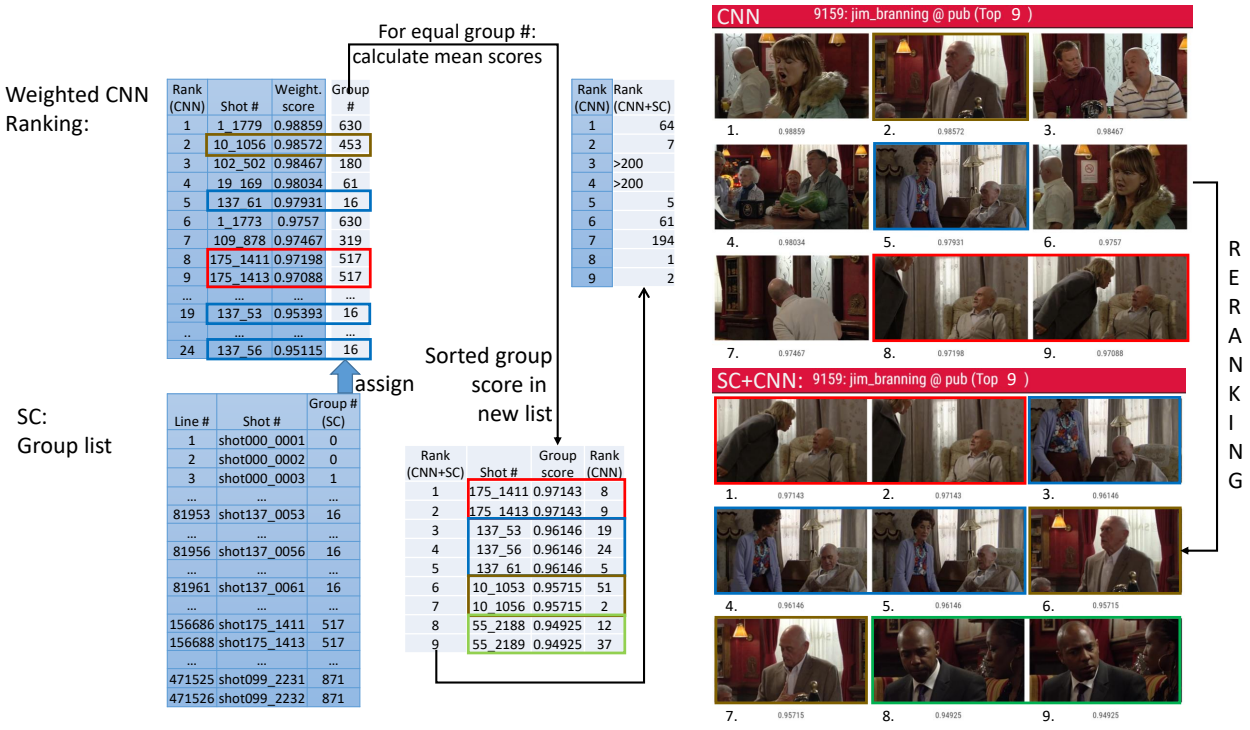

Figure 8. Example of image re-ranking after the fusion of CNN and SC for topic 9159 (finding Jim Branning at the pub).

3.2.2 Improvements to the Sequence Clustering

The algorithm of 2015 uses MPEG-7 visual descriptors: Color-Layout-Descriptor (CLD), Scalable-Color-Descriptor (SCD) and Edge-Histogram-Descriptor (EHD) and a Transition-Resistance (TR). We added the MPEG-7 Dominant-Color-Descriptor to the linear combination.

The distance metric has been wrapped into a logistic function (cf. to equation 3). This involves several advantages. The logistic function scales the result of the distance metric to [0,1] (see equation 4). Hence, the complex exit criterion becomes obsolete. The hierarchical agglomerative clustering exits its loop when a distance of 0.5 is reached. But most importantly, the weighting coefficients $(\alpha_0,\alpha_c,\alpha_s,\alpha_e,\alpha_d,\alpha_t)$ of the linear combination (see equation 2) can now be calculated with an approach to logistic regression. Therefor, a set of training data from EastEnders shot0 has been manually annotated in order to support the logistic regression computation.

$$x = \alpha_0 + \alpha_c CLD + \alpha_s SCD + \alpha_e EHD + \alpha_d DCD + \alpha_t TRD \quad (2)$$

$$D = \frac{1}{1+e^{-x}} \quad (3)$$

$$D = \frac{1}{1+e^{-(\alpha_0+\alpha_c CLD+\alpha_s SCD+\alpha_e EHD+\alpha_d DCD+\alpha_t TRD))}} \quad (4)$$

The linkage criterion of the hierarchical agglomerative clustering algorithm changes from complete linkage to the Wards criterion (Ward Jr, 2012) in order to get a more compact and homogeneous result.

3.2.3 Evaluation of the Sequence Clustering Algorithm

To evaluate the performance of the sequence clustering method, we did a full intellectual annotation of the sequences for ten videos of the EastEnders data set. We compared this ground truth data with the results of the algorithm. The parameters of the sequence clustering were trained only by using the development video shot0. The evaluation set consisted of 20,946 shots in total with a duration of 1,129 minutes. We use the Differential-Edit-Distance (DED) metric by (Sidiropoulos et al., 2012) for the evaluation while comparing the number of correctly labeled shots to the ground truth data. Our last years version of the algorithm had a DED result of 0.497 (0.0—optimal segmentation, 1.0—worst possible solution). The new advanced algorithm results in a mean DED of 0.272 (Rickert, 2016). The ground truth segmentation provided by manual annotation was replicated by our algorithm with an accuracy of 72.8 percent. This is a raise of 22.5 percent, compared to created baseline of SC in 2015.

4 Results

We participated with four different runs: Two interactive and two automatic ones. As they are building up on each other, we describe them in an inverse order in correlation to the results from lowest to highest rank (i.e. Run 4 to Run 1). The results of our best automatic and interactive runs are shown in Figure 7.

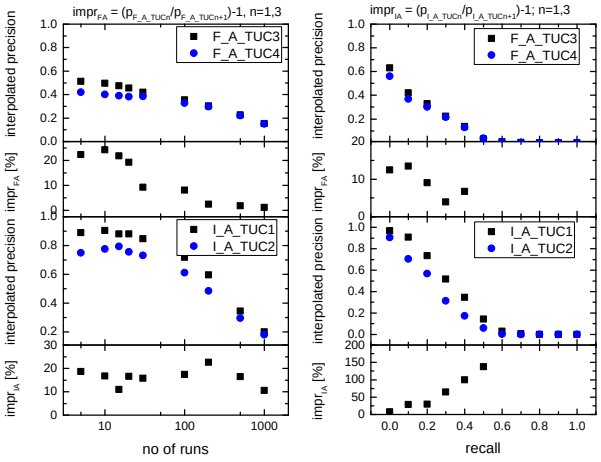

Figure 9. Improvement of fusion combining CNNs and Sequence Clustering

4.1 Run 4: CNN

Our lowest ranked fully automatic run ("F_A_TUC_4") is our internal baseline. We just used the result set of our CNN classifiers. The Mean Average Precision is 0.133, which already is a huge performance increase compared to our last years approach with CNNs scoring worse at 0.004 MAP. Compared to this years participants, we rank in the middle field of all evaluated runs.

One reason that prohibited a better performance is inherit by specific topic number like 9181, 9182, and 9183, which we only hit by accident because the searched person "Fatboy" could not recognised by our system. As mentioned before, we were not able to provide enough samples for this person at the learning phase of our CNN due to a no-show in the episode called "shot0", which was the primary data source for our training. Other topics with weak performance include those where people are searched at places they rarely visit. This appears to happen for example at topics 9161 and 9179. We also had problems with topics 9173 through 9177 as there are many false positive hits for the searched person "Stacey". One reason might be that she looks quite similar to some other females from the series what is recognised best with our CNN.

In contrast to that, our system worked remarkably well compared to the average at some topics. With an Average Precision of 0.443, topic 9165 is our best, because our CNNs are able to identify the searched person as well as the searched location; and since there are many occurrences of the person at this place. The same assumption applies to topic 9184 where we got an Average Precision of 0.368.

4.2 Run 3: CNN & SC

In our second fully automatic run ("F_A_TUC_3"), we used the result sets of our CNN classifiers, that were re-ranked by the results of Sequence Clustering, exemplarily shown in Figure 8. Comparing the results of all topics of Run 4 to Run 3, we get varying results. According to the Average Precision we are performing better at 19 topics and worse at 11 topics. The Mean Average Precision of the whole run gets a boost of 0.011 to 0.144.

Re-ranking by the means of sequence clustering can improve the results when the highest ranked results of the CNN classifiers are true positive hits and there are more occurrences of the topic in the same sequence of shots than those being missed by the CNN classifiers. For example, this can be seen on one side at topic 9160, where Average Precision improved from 0.274 to 0.366 and P15 scores increased from 0.8 to 1.0, and on the other side also at topic 9163, where P20 scores increased from 0.3 to 0.6 while the Mean Average Precision improved from 0.12 to 0.188. However, in the case that the first results occur to be false positive hits, the performance tends likely to decrease. This observation can be verified at topic 9186 where P30 scores decreased from 0.233 to 0 and Average Precision reduced from 0.107 to 0.087, or topic 9162, where P15 scores decreased from 0.133 to 0 being accompanied by an Average Precision drop from 0.069 to 0.037. A more detailed view on the change in precision by combining the results from CNN classifiers and Sequence Clustering is shown in Figure 9.

4.3 Run 2: Interactive Run CNN

Our first interactive run ("I_A_TUC_2") used the same result set of our CNN classifiers as Run 4. This was evaluated with our graphical evaluation tool which presented up to 2,000 instance candidates per topic out of a list of the top 3,000 results. Within the 5 minutes period per topic, one person was capable to intellectually examine about 600 candidates in average while grouping them into positive and negative result sets. The remaining positive candidates set of about 100 shots was filled up to 1,000 examples with the remainder of unevaluated results yielding the final results lists.

As expected, the Average Precision increased for almost all topics from Run 4 to Run 2. However, there are two exceptions, as topics 9161 and 9169 achieved a worse evaluation score. This might show a weakness of our graphical evaluation tool, as only middle-frames of all evaluated shots are shown to the reviewer while this single frames might not include the searched topic, although it is shown somewhere in the shot. Here, we might also observe a human error in the judgement of our evaluating person. By just looking at the fraction of a second at the chunks of six frames and deciding whether they show the topic or not, might lead to accidental skipping, especially when the occurrence of the topic appears to be is a very rare and distinct event like in topic 9161.

Our Mean Average Precision achieves a value of 0.224, which again is a notable performance increase compared to 0.17 at our best interactive run last year. Compared to all interactive runs this year, we are scoring a middle place, too.

4.4 Run 1: Interactive Run CNN & SC

In our second interactive run ("I_A_TUC_1"), we used the result set that was also used for run 3. The same evaluation process was applied, as described for Run 2.

Our Mean Average Precision is at 0.318, which is the best score that we achieved in all of our appearances at this evaluation campaign during the last three years. Compared to the other six interactive runs from this year, we score a second place with respect to Mean Average Precision. Our Average Precision is better for all topics compared to Run 3 as shown in Figure 7. However, there are several topics with a lower count of true positive hits, which again is a sign of human error by accidentally rejecting true positive hits by the user.

5 Future Work

This contribution introduced our approaches for instance search while employing a lot of modern open source state-of-the-art tools and frameworks. Due to the limited sample size of the query, a major drawback of the current method shows a lack of inference when requested person or location has not been trained a priori in the required training data. Hence, this approach can benefit from the introduction of transfer learning techniques that allow for an inference and integration of a very small data set of unseen examples in order to retrieve similar or at least somewhat related base classes in the feature space.

Acknowledgements. This work was partially funded by the German Federal Ministry of Education and Research in the program of Entrepreneurial Regions InnoProfile-Transfer in the project group localizeIT (funding code 03IPT608X). Programme material is copyrighted by BBC.

References

Awad, G., Fiscus, J., Michel, M., Joy, D., Kraaij, W., Smeaton, A. F., Qunot, G., Eskevich, M., Aly, R., and Ordelman, R.: TRECVID 2016: Evaluating Video Search, Video Event Detection, Localization, and Hyperlinking, in: Proceedings of TRECVID 2016, NIST, USA, 2016.

Dieleman, S., Schlter, J., Raffel, C., Olson, E., Snderby, S. K., Nouri, D., Maturana, D., Thoma, M., Battenberg, E., Kelly, J., Fauw, J. D., Heilman, M., de Almeida, D. M., McFee, B., Weideman, H., Takcs, G., de Rivaz, P., Crall, J., Sanders, G., Rasul, K., Liu, C., French, G., and Degrave, J.: Lasagne: First release., doi:10.5281/zenodo.27878, http://dx.doi.org/10.5281/zenodo.27878, 2015.

Lux, M. and Chatzichristofis, S. A.: Lire: Lucene Image Retrieval: An Extensible Java CBIR Library, in: Proceedings of the 16th ACM International Conference on Multimedia, MM '08, pp. 1085–1088, ACM, New York, NY, USA, doi:10.1145/1459359.1459577, 2008.

Nouri, D.: Nolearn: scikit-learn compatible neural network library, https://github.com/dnouri/nolearn, 2014.

Ren, S., He, K., Girshick, R., and Sun, J.: Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks, in: Advances in Neural Information Processing Systems (NIPS), 2015.

Rickert, M.: Inhaltsbasierte Analyse und Segmentierung narrativer, audiovisueller Medien, Ph.D. thesis, Technischen Universität Chemnitz, in press, 2016.

Ritter, M., Heinzig, M., Herms, R., Kahl, S., Richter, D., Manthey, R., and Eibl, M.: Technische Universität Chemnitz at TRECVID Instance Search 2014, in: Proceedings of TRECVID Workshop, Orlando, Florida, USA, 2014.

Ritter, M., Rickert, M., Juturu-Chenchu, L., Kahl, S., Herms, R., Hussein, H., Heinzig, M., Manthey, R., Richter, D., Bahr, G. S., and Eibl, M.: Technische Universität Chemnitz at TRECVID Instance Search 2015, in: Proceedings of TRECVID Workshop, Gaithersburg, Maryland, USA, 2015.

Sidiropoulos, P., Mezaris, V., and Kompatsiaris, I.: Differential edit distance as a countermeasure to video scene ambiguity, in: IEEE International Workshop on Machine Learning for Signal Processing, MLSP, pp. 1–6, 2012.

Simonyan, K. and Zisserman, A.: Very deep convolutional networks for large-scale image recognition, arXiv preprint arXiv:1409.1556, 2014.

Smeaton, A. F., Over, P., and Kraaij, W.: Evaluation campaigns and TRECVid, in: MIR '06: Proceedings of the 8th ACM International Workshop on Multimedia Information Retrieval, pp. 321–330, 2006.

Theano Development Team: Theano: A Python framework for fast computation of mathematical expressions, arXiv e-prints, abs/1605.02688, http://arxiv.org/abs/1605.02688, 2016.

Ward Jr, J. H.: Hierarchical Grouping to Optimize an Objective Function, Journal of the American Statistical Association, 58, 236–244, 2012.

Yeung, M., Yeo, B.-L., and Liu, B.: Segmentation of Video by Clustering and Graph Analysis, Computer vision and image understanding, 71, 94–109, 1998.