ITEC-UNIKLU
Ad-hoc Video Search Submission 2017

Manfred Jürgen Primus, Bernd Münzer, Klaus Schoeffmann
ITEC - Information Technology, Klagenfurt University
Klagenfurt, Austria
{juergen.primus,bernd,ks}@itec.aau.at

November 4, 2017

Abstract

This paper describes our approach used for the fully automatic and manually assisted Ad-hoc Video Search (AVS) task for TRECVID 2017. We focus on the combination of different convolutional neural network models and query optimization. Each of this model focus on a specific query part, which could be, e.g., location, objects, or the wide-ranging ImageNet classes. All classification results are collected in different combinations in Lucene indixes. For the manually assisted run we use a junk filter and different query optimization methods.

1 Introduction

The search in large video archives for certain scenes is very challenging. It is more difficult if the annotations and the metadat a belonging to the video data is not reliable and sufficient to support individual search tasks. The Ad-hoc video search (AVS) task is a challenge that models this practical problem. It has been performed for the first time in TRECVID 2016 [1].

For the challenge the IACC.3 data set is used that consists of 4593 Internet Archive videos with durations between 6.5 and 9.5 minutes. All in all these are about 600 hours of video with a total file size of 144GB. In addition to the videos a master shot boundary reference is provided that splits the videos into 335.944 single shots. Several videos have metadata as title, keywords, descriptions, etc. There is also the possibility to use metadata extracted by automatic speech recognition. This last mentioned metadata is not used by our approach.

Besides this data collection there are also thirty queries given. The target of a search query can be person(s), action(s), location(s), object(s), time specification(s), or combination of these objectives. Based on that the queries simulate ad-hoc requests in various levels of difficulty. The queries given in 2017 reach from a simple search for *"Shots of a newspaper"* to a search for *"Shots of a person talking behind a podium wearing a suit outdoors during daytime"*.

In the following, we present our approach to the AVS task and discuss the obtained results.

2 Ad-hoc Video Search (AVS) Approach

2.1 Architecture

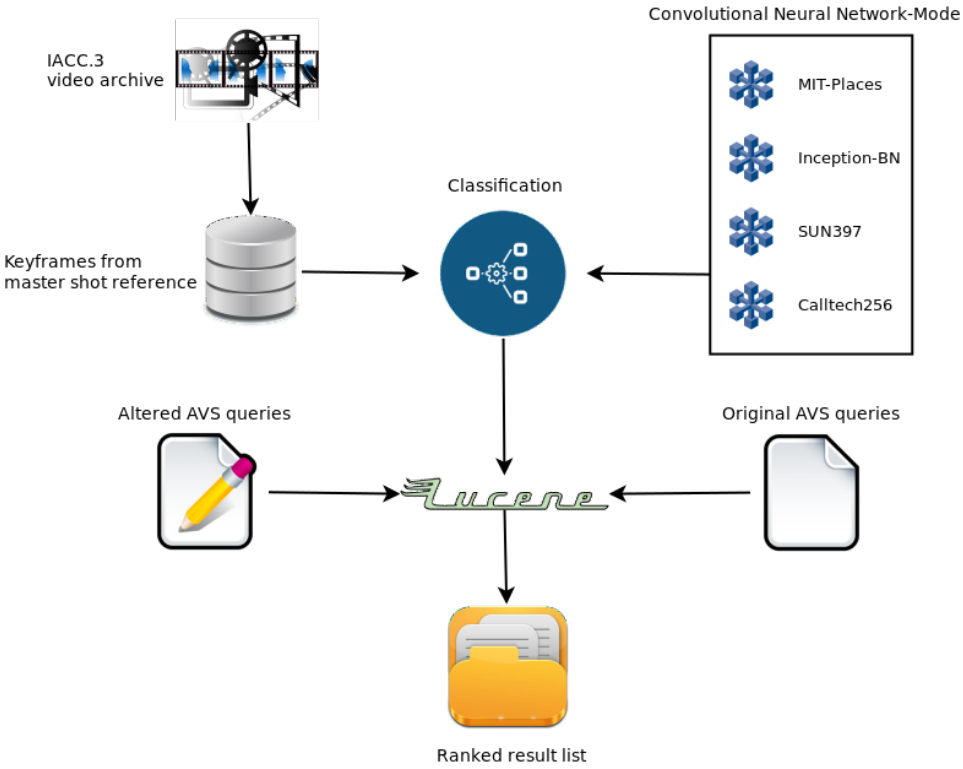

Figure 1: Architecture of our approach

The overall architecture of our approach is illustrated in Figure 1. From

each of the 335.944 shots of the IACC.3 dataset the middle frame is taken as representative keyframe. On the other hand we have chosen four different convolutional neural network (CNN) models that are used to analyze the keyframes' content. The result of the classification of the keyframes with the CNN-models is written in a Lucene index. The unaltered and the manually adapted queries are used to generate a list of 1,000 ranked results per query. The individual steps are described in detail below.

2.2 Convolutional Neural Network-Models

One main part of our framework covers several CNN-models that are used for the content-based analyzes of the keyframes. The caffe model zoo[1] provides a collection of different CNN-models for different purposes, e.g., the Imagenet Large Scale Visual Recognition Challenge (ILSCVRC), recognition of places, flowers, gender, and others. We use two of the provided CNN-models from the model zoo and we trained two further more.

MIT-Places Places-CNN from MIT has 205 classes and is trained on 2.5 million images [10]. Classes are, e.g., airport terminal, bakery shop, hospital room, promenade, underwater, and many more. In this framework we use the model that is trained on the GoogLeNet-architecture.

Inception-BN The Inception-BN is trained on 21.841 classes with 14,197,087 images [4]. The evaluation of the Inception-BN CNN-model results in a Top-1 error rate of about 25 %, which decreases to 7.8 % calculating the Top-5 error rate.

SUN397 The SUN397 dataset covers 397 classes of environmental scenes and places [9]. We train two models based on the AlexNet [6] and the GoogLeNet [8] architecture. The resulting SUN397 CNN-model based on the GoogLeNet-architecture performed slightly better than the AlexNet-based CNN-model. The training of the CNN models is performed using the CAFFE framework [5]. The images are re-sized that the smaller side has a size of 256 pixels. Afterwards, the images are cropped with respect to the center resulting in a 256×256 sized image. Then the images are fed into a Lightning Memory-Mapped Database (LMDB), which is used as input for the CNN. The solver for the GoogLeNet based model uses *Adam* as gradient-based optimization method provided by CAFFE. As base learning

[1] https://github.com/BVLC/caffe/wiki/Model-Zoo

rate we use 0.001; momentum 1 and momentum 2 are set to 0.9 and 0.999, respectively. The training batch size is set to 64 images. For the training based on the AlexNet architecture we use also *Adam* as stochastic gradient optimization algorithm. The other parameters are used as provided with the standard configuration.

The technical specification of the machine we use for our work is as follows: an Intel® Core™i7-6800K CPU with 3.40GHz, 64 GB of DDR4 RAM with 2,666 MHz, an ASUS GeForce GTX 1080 graphics card with 8 GB GDDR5X memory, and a Samsung SSD 850 pro.

Calltech256 The calltech 256 dataset is provided by Griffin et al. [3]. It is a collection of more than 30,000 images divided into 256 classes. The training is done with the AlexNet- and GoogLeNet-architecture and the same training parameters as above. Finally, we use the GoogLeNet based Calltech256-model for the classification of the keyframes.

2.3 Classification and Indexing

Each keyframe of a master shot is classified by all CNN-models (MIT-Places, Inception-BN, Sun397, Calltech256). Each of the models returns a list of concepts that are sorted by the corresponding confidence interval for every keyframe. The concepts are stored as inverted list with the Lucene-framework [2], which allows a fast and efficient retrieval of the required results.

Before an index can be used efficiently, it has to be decided which combination of the CNN-classifiers perform best and how many concepts should be stored per keyframe. Therefore, we generate for every single CNN-classifier multiple Lucene indexes that consist of the Top-N classifications. For the MIT-Places, the Sun397, and the Calltech 256 CNN-models we generated in each case indexes with the Top-1, Top-2, Top-3, and Top-5 classifications. For the Inception-BN indexes we used the Top-1, Top-2, Top-3, Top-4, Top-5, Top-10, Top-15, Top-20, Top-25, Top-30, Top-50, and Top-100 classifications. Additionally, we generated each of the indexes without and with the metadata provided with the video. In order to find the best performing index we define a test-dataset that has been used for the Video Browser Showdown 2017 [7, 2] and evaluate each of these Lucene-indexes based on this dataset. The best performing indexes are combined together and the top four indexes are used for the automatic and the manual assisted runs.

[2] http://lucene.apache.org/core/

3 Ad-hoc Video Search Results

We submitted four different result lists for the fully-automatic run and four different result lists for the manually-assisted run. The network-models and the configurations that are used for the runs are shown in Table 1.

Run	Network	M	Q/J	Performance
A1	Inception-BN (20), MIT-Places (3)	✓	✓	5.2%
A2	Inception-BN (20), MIT-Places (3)	✓		6.0%
A3	Inception-BN (20)	✓	✓	6.2%
A4	Inception-BN (20)	✓		6.9%
M1	Inception-BN (20), MIT-Places (3)	✓	✓	9.1%
M2	Inception-BN (20), MIT-Places (3)	✓		10.2%
M3	Inception-BN (20)	✓	✓	8.6%
M4	Inception-BN (20)	✓		9.5%

Table 1: Overview of submitted runs. The tick in column "M" means that the metadata of the videos is used. There is a tick in the column "Q/J"if query optimization for automatic runs is used or the junk filter for manual-assisted runs.

In our preliminary tests it turned out that the Inception-BN model performs best when it returns 20 classes with the highest confidence value. The MIT-Places network works best returning the three most likely classes. In the Video Browser Showdown Competition it has been shown that the metadata of the videos improves the video-retrieval results a lot. Therefore, we use the provided metadata in every run.

For the query optimization for the automatic run we use Lucenes PorterStemFilter and StopFilter. The performance values show that the use of query optimization as done with our approach impairs the result in comparison to the use of the CNN-networks without query optimization. The junk filter used for the manual-assisted runs is a list of concepts that returns weird content. These concepts are, e.g., chromatic, spectral, laser, supernova, etc. Interestingly, also this filter decreases the quality of the retrieved lists.

More details about the performance of the individual runs are illustrated in Figure 2. Configuration 4 obtained the by far best result of the automatic runs. As can be seen in the diagram, it especially performed particularly well for query 543 ("Find shots of a person communicating using sign language") and query 552 ("Find shots of a person wearing any kind of hat").

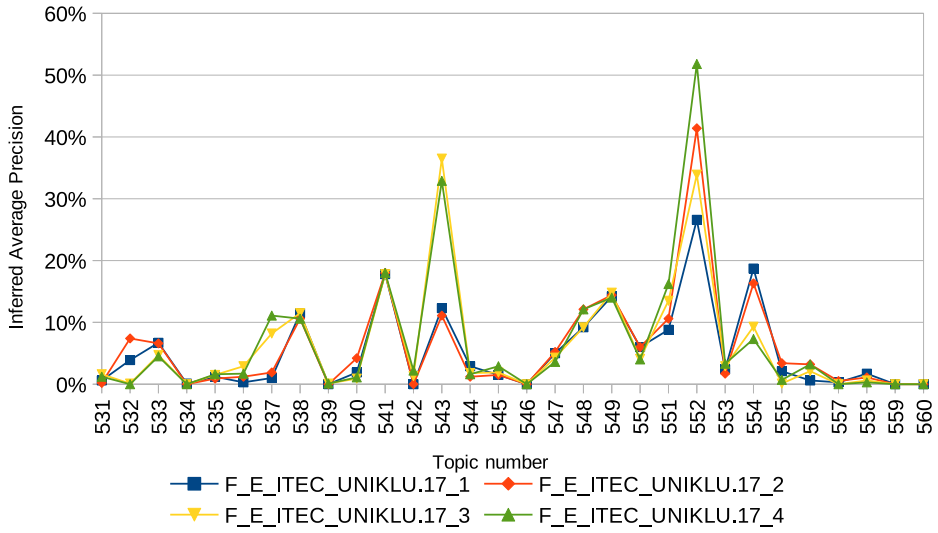

Figure 2: Results of the test runs

3. Configuration 1 obtained the by far best result. As can be seen in the diagram, it especially performed particularly well for query 528 ("Find shots of a person wearing a helmet"). In opposite to this good results there where several, where almost no match has been found. These where queries 534 ("Find shots of a person talking behind a podium wearing a suit outdoors during daytime"), 539 ("Find shots of an adult person running in a city street"), 542 ("Find shots of at least two planes both visible"), 546 ("Find shots of a male person falling down"), 557 ("Find shots of person holding, throwing or playing with a balloon "), 559 ("Find shots of a man and woman inside a car"), and 560 ("Find shots of a person holding, opening, closing or handing over a box"). These queries have in common that parts of the queries interact with an other part of the query or they are related somehow. That can not be depicted with our approach.

All manual-assisted runs show a very similar behaviour. The human in the loop can alter the query that it matches the available concepts better. They are also able to find synonyms that can not be found automatically. I.e, adding to query 559 ("Find shots of a man and woman inside a car") the search term *passenger*. But also the manual-assisted runs have queries with almost no match.

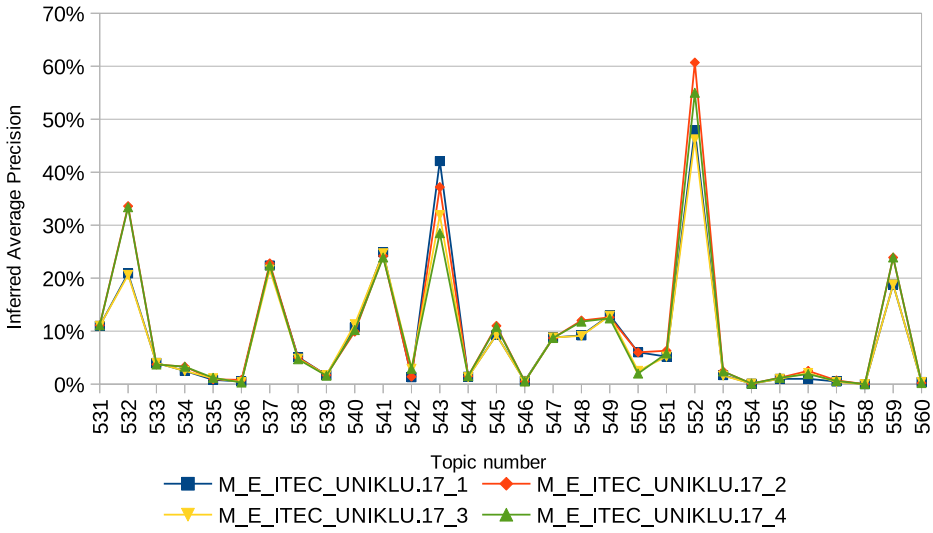

Figure 3: Results of the test runs

4 Conclusion

In this paper we describe our approach to the very challenging Ad-hoc video search task for TRECVID 2017. We experimented with different CNN-networks (MIT-Places, Inception-BN, Sun397, Calltech256) that were partly retrained from scratch. The results of the manual-assisted runs show significantly better results than the fully automatic runs. This shows that the user in the loop can improve well working automatic systems clearly.

References

[1] George Awad, Asad Butt, Jonathan Fiscus, David Joy, Andrew Delgado, Martial Michel, Alan F. Smeaton, Yvette Graham, Wessel Kraaij, Georges Quénot, Maria Eskevich, Roeland Ordelman, Gareth J. F. Jones, and Benoit Huet. Trecvid 2017: Evaluating ad-hoc and instance video search, events detection, video captioning and hyperlinking. In *Proceedings of TRECVID 2017*. NIST, USA, 2017.

[2] Claudiu Cobârzan, Klaus Schoeffmann, Werner Bailer, Wolfgang Hürst, Adam Blažek, Jakub Lokoč, Stefanos Vrochidis, Kai Uwe Barthel, and Luca Rossetto. Interactive video search tools: a detailed analysis of

the video browser showdown 2015. *Multimedia Tools and Applications*, 76(4):5539–5571, Feb 2017.

[3] G. Griffin, A. Holub, and P. Perona. Caltech-256 object category dataset. Technical Report 7694, California Institute of Technology, 2007.

[4] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International Conference on Machine Learning*, pages 448–456, 2015.

[5] Yangqing Jia, Evan Shelhamer, Jeff Donahue, Sergey Karayev, Jonathan Long, Ross Girshick, Sergio Guadarrama, and Trevor Darrell. Caffe: Convolutional architecture for fast feature embedding. In *Proceedings of the 22nd ACM international conference on Multimedia*, pages 675–678. ACM, 2014.

[6] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 25*, pages 1097–1105. Curran Associates, Inc., 2012.

[7] Klaus Schoeffmann, David Ahlström, Werner Bailer, Claudiu Cobârzan, Frank Hopfgartner, Kevin McGuinness, Cathal Gurrin, Christian Frisson, Duy-Dinh Le, Manfred Del Fabro, Hongliang Bai, and Wolfgang Weiss. The video browser showdown: a live evaluation of interactive video search tools. *International Journal of Multimedia Information Retrieval*, 3(2):113–127, 2014.

[8] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9, 2015.

[9] Jianxiong Xiao, James Hays, Krista A Ehinger, Aude Oliva, and Antonio Torralba. Sun database: Large-scale scene recognition from abbey to zoo. In *Computer vision and pattern recognition (CVPR), 2010 IEEE conference on*, pages 3485–3492. IEEE, 2010.

[10] Bolei Zhou, Agata Lapedriza, Jianxiong Xiao, Antonio Torralba, and Aude Oliva. Learning deep features for scene recognition using places database. In *Advances in neural information processing systems*, pages 487–495, 2014.