

**FACULTY  
OF MATHEMATICS  
AND PHYSICS**  
Charles University

**MASTER THESIS**

Bc. Tomáš Souček

**Deep Learning-Based Approaches for  
Shot Transition Detection and  
Known-Item Search in Video**

Department of Software Engineering

Supervisor of the master thesis: doc. RNDr. Jakub Lokoč, Ph.D.

Study programme: Computer Science

Study branch: Artificial Intelligence

Prague 2020

I declare that I carried out this master thesis independently, and only with the cited sources, literature and other professional sources. It has not been used to obtain another or the same degree.

I understand that my work relates to the rights and obligations under the Act No. 121/2000 Sb., the Copyright Act, as amended, in particular the fact that the Charles University has the right to conclude a license agreement on the use of this work as a school work pursuant to Section 60 subsection 1 of the Copyright Act.

In Prague, July 30, 2020

Hereby, I would like to thank my supervisor doc. RNDr. Jakub Lokoč, Ph.D. for his valuable advice, suggestions, and support he gave me. I am also thankful he enabled me to present the work at international conferences.

Furthermore, I thank the Department of Software Engineering for providing me almost limitless GPU resources for my research.

This research has been supported by Czech Science Foundation (GAČR) project 19-22071Y, GA UK project 1310920 and SVV project 260451.

Title: Deep Learning-Based Approaches for Shot Transition Detection and Known-Item Search in Video

Author: Bc. Tomáš Souček

Department: Department of Software Engineering

Supervisor: doc. RNDr. Jakub Lokoč, Ph.D., Department of Software Engineering

Abstract: Video retrieval represents a challenging problem with many caveats and sub-problems. This thesis focuses on two of these sub-problems, namely shot transition detection and text-based search. In the case of shot detection, many solutions have been proposed over the last decades. Recently, deep learning-based approaches improved the accuracy of shot transition detection using 3D convolutional architectures and artificially created training data, but one hundred percent accuracy is still an unreachable ideal. In this thesis we present a deep network for shot transition detection TransNet V2 that reaches state-of-the-art performance on respected benchmarks. In the second case of text-based search, deep learning models projecting textual query and video frames into a joint space proved to be effective for text-based video retrieval. We investigate these query representation learning models in a setting of known-item search and propose improvements for the text encoding part of the model.

Keywords: deep learning, shot boundary detection, known-item search, representation learning

# Contents

<b>Introduction</b>	<b>2</b>
Our Contribution . . . . .	4
Thesis Structure . . . . .	6
Authorship . . . . .	6
<b>1 Shot Boundary Detection</b>	<b>7</b>
1.1 Related work . . . . .	7
1.1.1 Deep Learning Methods . . . . .	9
1.1.2 Datasets . . . . .	10
1.2 TransNet . . . . .	12
1.2.1 Model Architecture . . . . .	12
1.2.2 Datasets and Evaluation Metric . . . . .	13
1.2.3 Training Details . . . . .	14
1.2.4 Prediction Details . . . . .	15
1.2.5 Results . . . . .	15
1.3 TransNet V2 . . . . .	18
1.3.1 Limitations of TransNet . . . . .	18
1.3.2 Datasets and Data Augmentation . . . . .	19
1.3.3 Architecture Improvements . . . . .	22
1.3.4 Other Changes . . . . .	25
1.4 Experiments . . . . .	28
1.4.1 Training Details . . . . .	28
1.4.2 Results . . . . .	28
1.4.3 Related Work Reevaluation Details . . . . .	30
1.4.4 Ablation Study . . . . .	31
<b>2 Text-Based Video Retrieval</b>	<b>37</b>
2.1 Related Work . . . . .	37
2.1.1 Weakly Supervised Approaches . . . . .	39
2.1.2 Winners of TRECVID Ad-hoc Video Search . . . . .	40
2.2 Our Method . . . . .	43
2.2.1 Problem Statement . . . . .	43
2.2.2 Video Representation . . . . .	43
2.2.3 Text Representation . . . . .	44
2.2.4 Loss Function . . . . .	45
2.3 Experiments . . . . .	46
2.3.1 Datasets and Evaluation Metrics . . . . .	46
2.3.2 Training Details . . . . .	47
2.3.3 Results . . . . .	47
2.3.4 Ablation Study . . . . .	49
<b>Conclusion</b>	<b>53</b>
<b>Bibliography</b>	<b>54</b>
<b>List of Figures and Tables</b>	<b>65</b>

# Introduction

For humans, vision plays the most important role in building representation of the surrounding environment. We rely on sight heavily – some even estimate that information from our eyes accounts for eighty percent of stimuli from the environment. It is, therefore, no wonder that, with the help of cheap recording devices that everybody carries in their pockets, we have become obsessed with capturing what we see.

Smartphones enabled us to record every moment of our lives, and personal archives of photos and videos started growing rapidly. With the rise of social networks and the internet in general, we began not only capturing photos and videos but also sharing them online. In 2013 it was reported that just only Facebook’s databases contained 250 billion photos with 350 million new photos added every day by its users<sup>1</sup> – a figure that has probably grown since. Video platform YouTube announced in 2019 that it had been receiving more than 500 hours of video every minute<sup>2</sup>. To put that in context, humans live shorter lives than the length of videos uploaded to the service every day.

With such amount of multimedia being recorded, new problems and challenges arise. Given the enormous sizes of collections, it is impossible to sequentially browse through the data. Efficient methods for work with multimedia collections need to be utilized. The use-cases for these methods vary from a plethora of methods for searching and transcribing the content to the summarization of the individual parts of the collections [1, 2, 3]. Multimedia in the collections are, however, usually not manually annotated, and contain only basic metadata such as date, time, and location, if any at all.

In recent years, thanks to deep learning, we have seen huge improvements in many areas, including automatic annotation of images, videos, and other types of multimedia. Yet video, one of the richest type of multimedia, still presents multiple challenges, such as its enormous size. Compared to other types of media like text, audio, or images, video sizes are whopping – a short clip can easily require a hundred times as much space as a single image. Comparison is even starker in the case of audio and text. As the deep learning-based approaches usually require large datasets, applying deep learning on video-related tasks depends upon the collection of harder to obtain and more time consuming to annotate datasets when compared to their image counterparts. Furthermore, training of the models requires more computational power and time due to the increased dimensionality of the problem.

A popular approach to circumvent the need for large training datasets or lack of computational power in end-to-end deep learning is to decompose a problem into multiple sub-problems, solving each independently. In the video domain, that means, for example, to extract image features from each frame and train a model utilizing only the extracted features instead of the high dimensional frames themselves [2, 4]. In the domain of self-driving cars, one can create a model predicting a depth map from multiple images [5] and then another model for

---

<sup>1</sup><https://www.theverge.com/2013/9/17/4741332>

<sup>2</sup><https://www.cnbc.com/2018/03/14/with-over-1-billion-users-heres-how-youtube-is-keeping-pace-with-change.html>

obstacle detection using the images enriched with their depth [6]. In text-based systems, it is not uncommon to build on top of Word2Vec-like [7] pre-trained embeddings instead of training embeddings from scratch [2, 4]. In general, the less training data there is, the more likely there will be benefits to utilizing the multi-step approach.

Many video-related methods take the decomposition approach to an extreme by discarding any temporal information from a video and working only with single frames or simply averaging multiple frames’ features [2, 4]. Surprisingly, until very recently [8], these methods dominated many video related benchmarks, probably due to lack of annotated video data. Strangely, even with the introduction of large annotated video datasets [9], we do not see such a sharp boost in performance, which can be seen in the image domain. Some theorize it is in part due to vagueness and ambiguity of actions observed in videos. Action ‘*playing tennis*’ can for different people mean vastly different clips. TV broadcasts from Wimbledon, table tennis tournament with friends in a basement, a kid hitting a ball in a backyard, or a computer game are all valid alternatives. Even though there is also ambiguity between objects, it is usually less pronounced.

To overcome the ambiguity and to correct errors of automatic methods, there has been research in human-assisted approaches [10]. They revolve around assisted browsing in the collections by utilizing novel user interfaces [11], hierarchical collection maps [12], or iterative query refinement by positive and negative examples [13, 14]. However, a comparison of such approaches is difficult because a person needs to be present in the evaluation loop. In recent years competitions such as Video Browser Showdown (VBS) [10] or Lifelog Search Challenge (LSC) [15] emerged to accelerate research in human-assisted approaches, in particular in a task of known-item search. The known-item search (KIS) task represents a situation where a user searches for a given item (usually an image or a segment from a video) in a large collection of data. With an increasing amount of multimedia content we generate, there is a wide range of image or video collections where know-item search scenario may play an important role – a personal photo or video archive, footage from CCTV cameras, databases of news clips or medical videos to name a few.

Over the years, Video Browser Showdown served as a stage for an evaluation of many known-item search approaches in large video collections. These are important concepts mentioned by winning teams:

**Powerful query initialization.** It is beneficial to limit search space by an initial query that filters out most of the unrelated items or, with enough luck, immediately discovers the searched video segment. For many years, color [16] or edge sketches were widely used; however, these are useful only if a user knows the exact visual representation of a searched scene. Further, with advances in deep learning, concept and text search replaced sketches as it is a more effective approach [17, 18]. Lastly, note that the initial query also plays an important role in many query refinement methods introduced in the next paragraph as they require negative but also positive samples, which are hard to gather without good initialization.

**Effective query refinement.** In large collections with a lot of similar content, it is unlikely that the searched scene will be found on the first try with

a query. Many KIS tools support either assisted text query reformulation based on presented results or encourage users to select positive and negative examples to further narrow and rearrange the result set [13, 14]. Also, ‘find similar’ function is widely used to retrieve similar content from the whole collection [19, 18] – nowadays usually implemented as nearest-neighbor search in high-dimensional representations of the content computed by deep neural networks.

**Fast assisted browsing.** Browsing is utilized if a scene is not found using only the approaches mentioned above. Good browsing approaches should exploit information from initial query results but also consider further exploration. Some methods of browsing involve computation of 2D image maps [20, 12], and some utilize the power of virtual reality [11]. Recently, Bayesian approach that samples images to display based on their probability proved to be successful [14].

**Intuitive user interface.** The tools are operated by not only the authors but also by novice users. A cluttered user interface or hard-to-understand retrieval models result in lower performance of a tool when operated by novices.

**Key frame selection.** In general, videos may be too long and may contain different unrelated scenes. Therefore, the search is often performed on shorter video segments. Usually, a segment of a video is represented by a single frame (keyframe). The selection of the segment and its keyframe can be performed by thresholding differences between (multiple) adjacent frames and their visual features [21]. However, setting threshold too low results in oversampling, which increases database size and clutters result lists. Too high threshold causes some unique video segments to be missed. Recently more accurate shot boundaries, detected by deep learning-based methods [22], have been used instead of rather vaguely defined segments.

With these concepts in mind, at VBS 2019 [10] on V3C1 1000 hour dataset [21], the best performing teams of experts were able to solve all ten tasks where the search clip was played to an audience, and six out of ten tasks where only a textual description was available. At VBS 2020, on the same dataset, the best team in expert session solved five out of six visual tasks and eight out of ten textual tasks. For the next years, much bigger datasets are planned; however, given the rapid pace of innovation in deep learning and other related areas, we expect to see even better results in the foreseeable future.

## Our Contribution

In this thesis, we propose, implement, and evaluate new methods and improvements in two key areas of video retrieval. Namely a shot boundary detection and text search in video or image collections. For the shot boundary detection – a task to detect continuous video sequence captured by a single camera – we present a state-of-the-art method based on deep learning that outperforms both standard thresholding methods as well as more recent learning-based methods on multiple public benchmarks. For text search in video collections, we improve



W2VV++ [4] – a model that computes the similarity between a text and video by projecting both modalities into joint vector space using a neural network. We enrich W2VV++ with a more powerful natural language model and discuss its greatly superior performance on some tasks while achieving a bit lower performance on others.

Our work is a culmination of many years of research primarily focused on known-item search in video collections. Some of the work presented in this thesis has been published at international conferences. Aside from the already published content, the thesis contains more detailed method descriptions as well as additional experiments and ablation studies, while other aspects of known-item search are mostly left out as the sole focus of the thesis is shot boundary detection and text search. The main publications regarding the content of the thesis are the following:

1. **A framework for effective known-item search in video** [22]  
Full paper describing effective approaches to known-item search. The paper also introduces TransNet shot boundary detection network. Published at ACM International Conference on Multimedia 2019 (CORE A\*).
2. **TransNet: A deep network for fast detection of common shot transitions** [23]  
Short paper slightly extending the version published at ACM Multimedia. Published on Arxiv.
3. **A W2VV++ Case Study with Automated and Interactive Text-to-Video Retrieval** [24]  
Full paper studying W2VV++ query representation learning model [4] in text-based video retrieval scenarios. The paper also introduces our BERT extension to the W2VV++ model. Accepted to ACM International Conference on Multimedia 2020 (CORE A\*).
4. **TransNet V2: An effective deep network architecture for fast shot transition detection** [25]  
Short paper describing our TransNet V2 model.

We also list some of the author’s publications in the field of known-item search:

5. **Interactive Video Retrieval in the Age of Deep Learning – Detailed Evaluation of VBS 2019** [10]  
Journal paper analyzing the results of VBS 2019. Published in IEEE Transactions on Multimedia (IF = 6.051).
6. **VIRET: A video retrieval tool for interactive known-item search** [26]  
Short paper presenting our VIRET tool and showing an analysis of interaction logs from VBS 2019. Published at ACM International Conference on Multimedia Retrieval 2019.
7. **VIRET at Video Browser Showdown 2020** [17]  
Demo paper describing latest version of our retrieval tool. Published at ACM International Conference on Multimedia Modeling 2020.

Other demo papers [19, 27] were published on the occasions of the VBS and LSC competitions at MMM and ACM ICMR respectively. Papers *Revisiting SIRET Video Retrieval Tool* [28] and *Using an Interactive Video Retrieval Tool for LifeLog Data* [29] were already presented in the author’s bachelor thesis.

We proudly report that we achieved first and second place at VBS 2018 and VBS 2019 respectively. At VBS 2020, two tools [17, 14] using our shot boundary detection method and a simplification of the W2VV++ model, as reported in [24], achieved first and second place. Furthermore, we achieved third and second place at LSC 2018 and LSC 2019 respectively.

## Thesis Structure

The thesis is divided into two chapters. The first chapter introduces methods for shot boundary detection and presents TransNet – a neural network for shot detection (Section 1.2). Further in the chapter, significant improvements to TransNet are made, and a new network TransNetV2 is introduced (Section 1.3). Finally, related works are reevaluated for a fair comparison with TransNetV2 in Section 1.4, and an ablation study is made. The second chapter introduces approaches towards text search in image and video collections, especially the W2VV++ model by Li et al. [4] (Section 2.1), and our extension W2VV++BERT is presented in Section 2.2. Both models are thoroughly evaluated together with an ablation study in Section 2.3.

Source code for TransNetV2, including a version with a trained model for easy integration and reevaluation, training scripts, and dataset manipulation scripts are provided as an attachment to the thesis as well as available online at <https://github.com/soCzech/TransNetV2>. Source code for the W2VV++BERT network, together with trained weights and details for feature extraction, are also attached to the thesis as well as available online at [https://github.com/soCzech/w2vvpp\\_bert](https://github.com/soCzech/w2vvpp_bert).

## Authorship

All experiments presented in this thesis have been conducted solely by the author of the thesis with the only exception of the original TransNet model, described in Section 1.2, which has been created by Mgr. Jaroslav Moravec. However, further TransNet evaluations were done by the author of this thesis. Also, the model’s description in Section 1.2 as well as the paper *TransNet: A deep network for fast detection of common shot transitions* [23] and corresponding sections of the paper *A framework for effective known-item search in video* [22] were written by the author of this thesis with the help of his supervisor. TransNetV2 presented in Section 1.3 has been solely the author’s work.

As the work presented in this thesis has been published at international conferences, some of the thesis’ content may correspond to the author’s publications listed above. All such possible correspondences were written by the author of this thesis with the help of his supervisor.

# 1. Shot Boundary Detection

Commonly, a video structure is as follows: The video is divided into scenes and each scene into one or more shots. A shot is a continuous frame sequence captured by a single camera action [30]. Some works introduce stories, that group semantically related scenes [31], or threads, that group similar shots, e.g. captured from the same camera point [32]. Shots are, however, the most studied and the most used since shot detection is considered a fundamental step in video analysis. Information about shots is being exploited in video summarization [33], video retrieval for advanced browsing and filtering [34], or even content-based copy detection [35]. However, information about the transitions is not available in the video format. Therefore, automated shot boundary detection methods need to be employed.

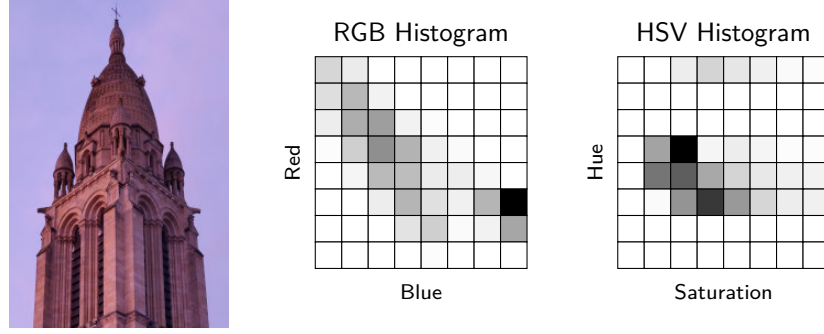
Any successful method must take into account that shot changes can be either immediate (hard cuts) or gradual. Common types of gradual transitions include dissolves (interleaving of two shots over a certain number of video frames), fade-ins and fade-outs (also considered as special types of dissolves where one shot is a blank image of a single color) and wipes (one shot slides from a side on top of the other shot). However, there are also many more exotic geometric transformations from one shot to another one. To make matters worse, shot boundary detectors must distinguish between shot transitions and sudden changes in a video caused by flashing light or partial occlusion of the scene by an object passing closer to the camera. Fast camera motion or motion of an object in the scene should also not be mistaken for a shot transition. This may indicate that some semantic representation of a scene is necessary to correctly segment a video.

Lastly, in some cases, a shot boundary is not a well-defined concept and question whether there is a transition may be subjective. Here we list some ambiguous cases and leave it for the reader to decide whether it should be a transition or not. Is a moment when captions are displayed at the end of a movie a transition? What if they raise from the bottom of the frame? A camera slowly enters a dark room – is it fade-out? Newscast with two reporters side-by-side – what if there is cut in a window of one reporter? Yet, with these examples, we only scratch the surface of the problem. Therefore, when used in the wild, shot boundary detection methods need to be tuned to eliminate these ambiguities based on a particular task at hand.

## 1.1 Related work

There has been a lot of research in shot boundary detection methods. The methods range from the most basic ones, that utilize only pixel-wise differences [36] effective for cut detection in stationary shots with a small number of moving objects, to more robust techniques that have been developed in the last years. Firstly, we list some of the ‘standard’ methods not based on neural networks, further in the section deep learning approaches are discussed. For a more complete overview of the ‘standard’ methods, we point the reader to some of many review studies available [31].

**Color histograms.** Widely used technique for shot boundary detection. It



**Figure 1.1:** Visualization of RGB and HSV histogram of a single image. The intensity of each bin indicates a number of image pixels with the given color. Third dimension depicting green or value respectively is not shown.

is based on computing histograms for each frame and thresholding distance between consecutive histogram representations. Instead of traditional RGB histograms, some works use HSV histograms to reduce disturbance in illumination [37] (comparison shown in Figure 1.1) or LAB histograms since they better approximate the way humans perceive color [35]. To improve the detection rate, frames can be divided into multiple patches with histograms computed for each patch [38]. Also, instead of a distance-based comparison,  $\chi^2$  comparison of color histograms is sometimes used.

**Feature based methods.** One of the earliest feature-based methods computes changes in edges of subsequent frames [39]. It is based on observation, that during transition, new edges appear far from the locations of old edges and vice versa. However, the work of Rainer Lienhart [40] shows that the method brings no significant improvements over color histograms. Nonetheless, edge-like features are utilized in shot boundary detector by Shao et al. [41] where histogram of gradients is used as a secondary method to HSV histogram. Apostolidis et al. [42] take advantage of scale and rotation invariant SURF descriptors [43] to measure differences between a pair of frames.

**Clustering.** Given a feature vector for each frame such as a color histogram or more recently a vector computed by a neural network, a clustering algorithm can be run to determine shot boundaries. Verma et al. [44] use a special form of hierarchical clustering to join consecutive frames into shots while Baraldi et al. [45] utilizes clustering to determine which shots belong to a particular scene.

**Support vector machines (SVMs).** Given a set of adjacent frame similarities, it may seem arbitrary to select a threshold value that decides whether there is a transition or not, especially for gradual transitions. In the work of Chasanis et al. [46], SVMs are trained on a sliding window of neighboring frame similarities to predict shot boundaries instead of using a simple threshold. Tsamoura et al. [47] increase the chance of detection by adding new similarity/distance metrics based, for example, on Color Coherence Vectors [48] to the SVM’s input feature vector.

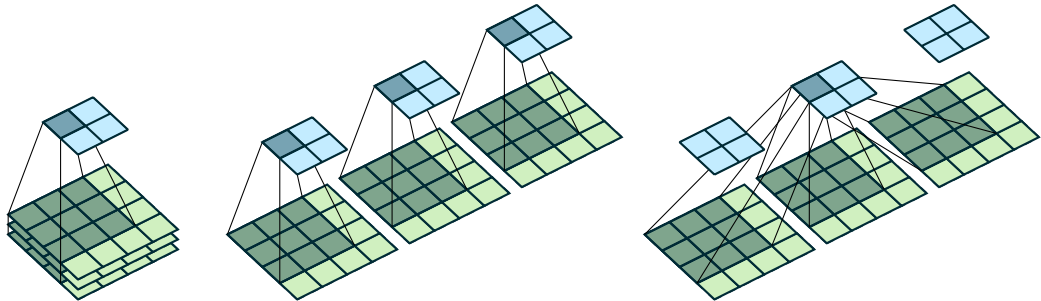
**Flash detection.** Commonly, some videos contain either photographic

flashes or overexposed frames due to change in illumination of a camera sensor, e.g. when a light is turned on. It is not uncommon to perform a post-processing step that compares frames or their features such as luminance values that are adjacent to potential shot boundary [49]. If there is no significant change observed in the adjacent frames, probably a flash occurred.

**Other false positive suppression methods.** Motion in a scene or motion of a camera can result in many false positives. Camera motion estimation [50] or optical flow [51] methods are used to reduce the number of false alarms, especially for gradual transitions. When not using SVMs, a threshold for transition detection has to be set. Work of Yeo et al. [52] sets the threshold adaptively since using the same threshold for different video genres can result in many false positives in one and false negatives in another.

Between years 2001 and 2007 there had been automatic shot boundary detection (SBD) challenge held annually at TRECVID (TREC Video Retrieval Evaluation) [53] with teams utilizing many of the described techniques; however, it was discontinued due to no observed improvements over the last years of the challenge. Significant improvements came with deep learning revolution when, for example, the work of Hassanien et al. [54] achieved on the RAI dataset F1 score of 0.94 improving previous state-of-the-art by 0.1 from 0.84 [55, 42]. Therefore, the next paragraphs introduce deep learning approaches towards the shot boundary detection.

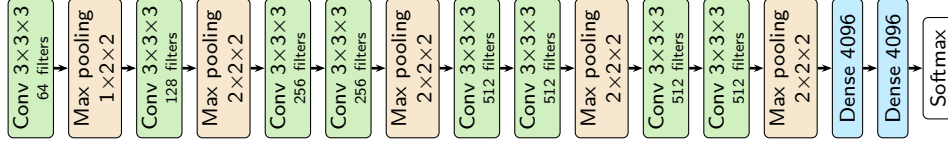
### 1.1.1 Deep Learning Methods



**Figure 1.2:** Comparison of early fusion (left), late fusion (middle) and 3D convolutions (right). In the case of late fusion, some aggregation over frames must be done to capture temporal information (not shown).

Exciting results of Krizhevsky et al. [56] sparked a great interest in image classification research using convolutional neural networks. These networks, trained in a fully supervised manner, learn a rich semantic representation that can be repurposed to novel generic tasks [57]. Therefore, one of the first deep learning SBD works [58] revolves around utilizing this readily available ‘deep’ representation. It uses FC-6 features from AlexNet neural network [56] and employs cosine similarity between frames’ features in the decision process, whether there is transition.

To utilize temporal information in a neural network directly, many approaches have been developed. The *late fusion* [59] approach extracts features from individual images. The features are then merged, for example, by averaging them



**Figure 1.3:** C3D architecture by Tran et al. [61]. C3D net has 8 convolution, 5 max-pooling, and 2 fully connected layers, followed by a softmax output layer.

over time or concatenating a fixed number of them. Fully connected layers are placed atop the aggregated representation. The *early fusion* approach stacks  $N$  (subsequent) frames in channel dimension therefore increasing number of channels in input from three (RGB) to  $3 \times N$ . *Two-stream* architecture [60] utilizes two networks – one for single frame and another one that processes optical flow information from multiple adjacent frames. Figure 1.2 shows a visual comparison between these approaches.

However, all of the above approaches utilize only 2D convolutions. First widely popular network utilizing 3D convolutions C3D (Figure 1.3) introduced in the work of Tran et al. [61] showed modest improvements over 2D approaches. Bigger I3D network [62], closely resembling 2D convolutional network InceptionV1 [63], brought further improvements and also showed the benefits of the Two-Stream approach still hold even for 3D convolutional networks. Other improvements were achieved by separation of 3D convolutions into spatial-only and temporal-only convolutions [64, 65].

Using 3D convolutions for shot boundary detection has been popularized by Gygli [66] and Hassanien et al. [54]. The later work introduces DeepSBD framework consisting of a CNN-based classification step, a merging step, and a post-processing step. The network, based on C3D architecture, takes 16 subsequent frames and predicts whether the segment contains sharp or gradual transition. The logits are, however, not used directly, but they are fed to an SVM classifier to give a labeling estimate. Further, consecutive segments with the same labeling are merged, and Bhattacharyya distance between color histograms of the first and the last frame of the proposed transition segment is computed. If the distance is small, the segment is considered as false positive and removed from a set of transitions.

The work of Gygli removes all post-processing steps by using only predictions from a 3D convolutional network. The network consists of 5 convolutional layers with a much smaller number of parameters than C3D. It takes 10 subsequent frames and predicts whether there is a transition between the middle frames. Because of the fully convolutional nature, the network can be stretched to take  $N$  frames and produce output for the middle  $N - 9$  frames eliminating the need for processing most of the frames multiple times. Further, this approach can localize the exact position of a transition, which is impossible in DeepSBD. However, reported performance is worse than the one reported by Hassanien et al.

### 1.1.2 Datasets

The above mentioned deep learning methods all rely on large annotated datasets. Since most of the available datasets are small and also used for evaluation, both Gygli [66] and Hassanien et al. [54] overcome the need for the large dataset by

generating synthetic training examples. Both works generate sharp transitions (hard cuts), dissolves, and simple horizontal wipes. The DeepSBD system further enriches a set of possible transitions by non-linearly interleaving dissolves and more complex wipes. Gygli adds artificial flashes to the non-transition sequences to make the network invariant to these kinds of changes.

For evaluation TRECVID SBD datasets are commonly used; however, they are old and publicly unavailable. A small, manually annotated dataset of broadcasting videos, mainly documentaries and talk shows from the archive of Italian TV station Rai Scuola [55], has been used by many works. Further, the same authors released manually annotated shot and scene boundaries for all 11 episodes of the BBC educational TV series Planet Earth [45]. The whole dataset contains around 4900 shots and 670 scenes. Recently, new database ClipShots [67] was released containing 4039 online videos with manually double annotated 128636 cut transitions and 38120 gradual transitions. The dataset contains videos from Youtube and Weibo covering more than 20 categories, including sports, TV shows, animals, etc. with hand-held camera vibrations, large object motions, and occlusion. Its test set consists of 500 videos with 5876 cut transitions and 2422 gradual transitions.

The authors of the ClipShots dataset also introduce their system based on a three-step pipeline. Firstly, SqueezeNet [68] features for each frame are used to compute similarities between frames to reduce the number of candidates for transitions. A cut detector is applied to the transition candidates, and, in the end, if no cut transition is detected, a gradual detector is applied. For the cut detector, either C3D network or 2D ResNet-50 with the input of 6 concatenated subsequent frames is used, with the latter achieving better results. For gradual transition detection, both the DeepSBD-like system and a 3D version of ResNet-18 [69] are tested. The version with ResNet performs per-frame classification as well as transition regression similar to region proposal in object detection. According to the authors, their ResNet based system outperforms DeepSBD<sup>1</sup>.

---

<sup>1</sup>However, the only code provided by the authors is the reimplementaion of DeepSBD, which contains an evaluation script that does not account for double detection of transitions and possibly other errors. Therefore, the reported results should be taken with a grain of salt.

## 1.2 TransNet

This section introduces TransNet, scalable architecture for shot boundary detection introduced in [23, 22]. The network features multiple dilated 3D convolutional operations per layer and achieves state-of-the-art results on the RAI dataset [55]. Firstly, we describe the model architecture, then we introduce performed experiments and report their results. Further, in the next section, improvements to the TransNet are presented. Some texts in this section overlap with our paper [23]. These texts were written by the author of this thesis.

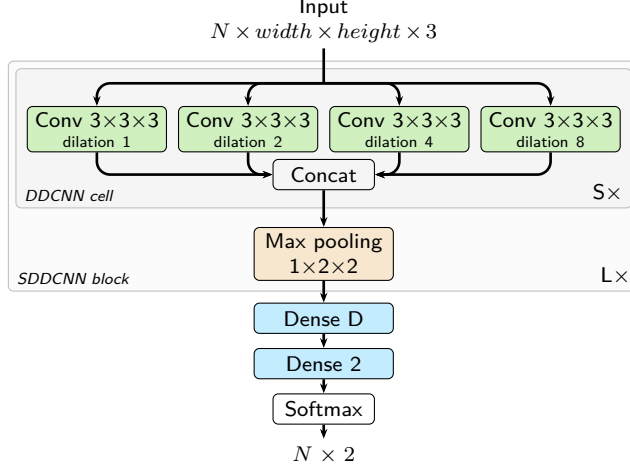
### 1.2.1 Model Architecture

The proposed TransNet architecture (Figure 1.4) is inspired by many very successful convolutional architectures for image classification [56] or action recognition [61]. Commonly these architectures feature a layer or a cell that consists of a single or multiple convolutional operations, each with different parameters. These cells are stacked to form the whole network. To reduce the spatial and temporal resolution of the network, reduction cells are included in between some of the standard cells. These consist of either pooling operations or convolutions with greater strides. TransNet is built upon these concepts with the only exception of temporal pooling, which is not applied to precisely localize shot boundaries on the level of individual frames. In general, the network takes a sequence of  $N$  consecutive video frames and applies series of 3D convolutions returning a prediction for every frame in the input. Each prediction expresses how likely a given frame is a shot boundary.

Convolutional neural networks for video-related tasks such as C3D [61] introduced for action recognition employ many 3D convolution layers. However, a big problem with 3D convolutions is that even minimal  $3 \times 3 \times 3$  convolutions can be prohibitively expensive. Yet it is not uncommon for a transition to span across dozens of frames; therefore, it is necessary to ensure a wide temporal field of view for the convolution operations, which is computationally even more expensive. Also, the larger the convolutional kernels are, the bigger the number of parameters is, which can result in over-fitting, especially since shot boundary datasets are rather small compared to, for example, large image classification datasets such as ImageNet.

TransNet solves this problem by utilizing dilated convolutions that have been successfully applied to many tasks ranging from image segmentation [70] to audio generation [71]. The main building layer of the model, dubbed Dilated Deep CNN (DDCNN) cell, is designed to have a large field of view with a minimal number of parameters while still maintaining the ability to capture a change in two consecutive frames. The cell consists of four 3D  $3 \times 3 \times 3$  convolutional operations each with  $N_{in} \times 3 \times 3 \times 3 \times N_{out}/4$  learnable parameters where  $N_{in}$  is number of filters from the previous layer and  $N_{out}$  is number of filters outputted by the cell. Each of the four convolutions employs different dilation rates for the temporal dimension. The rates are 1, 2, 4, and 8, i.e. the first convolution is standard  $3 \times 3 \times 3$  convolution that looks one frame to the left and one frame to the right, the last convolution looks at the eighth frame to the left and the eighth frame to the right. The four convolutional outputs are concatenated, creating a





**Figure 1.4:** TransNet shot boundary detection network architecture [23]. Note that  $N$  represents the length of a video sequence, not batch size.

representation with  $N_{out}$  filters. Compared to standard convolution with the same number of output filters and the same field of view, the DDCNN cell achieves more than a six-fold reduction in the number of learnable parameters.

Multiple DDCNN cells on top of each other, followed by spatial max pooling, form a Stacked DDCNN block. The TransNet consists of multiple SDDCNN blocks, every next block operating on smaller spatial resolution but with a larger number of filters, further increasing the expressive power and the receptive field of the network. Two fully connected layers refine the features extracted by the convolutional layers and predict a possible shot boundary for every frame representation independently (layer weights are shared). ReLU activation function is used in all layers, with the only exception of the last fully connected layer with softmax output. Stride 1 and the ‘same’ padding is employed in all convolutional layers.

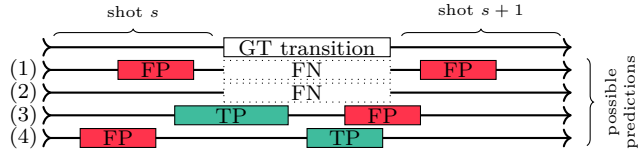
### 1.2.2 Datasets and Evaluation Metric

Following the works of Gygli [66] and Hassanien et al. [54], we generate the dataset synthetically. Unlike Hassanien et al. who generates the transitions prior to the training, we create transitions on the fly during training, i.e. each network is pre-trained with slightly different shots. This approach does not require to store pre-generated combinations of a first shot, a transition, and a second shot and allows for completely arbitrary shots joined with any transition. We take predefined temporal segments from the TRECVID IACC.3 dataset [72]. The dataset contains approximately 4600 Internet Archive videos with a mean duration of almost 7.8 minutes. During training, pairs of the predefined video segments are randomly selected from a pool of available ones. More specifically, we consider segments of 3000 IACC.3 randomly selected videos. Segments with less than 5 frames were excluded, and from the remaining set, only every other segment was picked, resulting in selected 54884 segments.

The validation dataset consists of additional 100 IACC.3 videos not present in the training set that were manually labeled by Moravec [23]. The dataset contains approximately 3800 shots. For testing, the RAI dataset [55] of ten

manually annotated videos is used. The videos are mainly short documentaries or talk shows from an archive of an Italian TV station.

Following the work of Baraldi et al. [55], we use the F1 score as the evaluation metric<sup>2</sup>. Baraldi et al. report the F1 score as an average of individual F1 scores for each video. We rather use the standard F1 score – a function of true/false positives and false negatives from all the videos but report both where appropriate. In Figure 1.5, we show some cases of detected transitions considered to be true positives, false positives, or false negatives. A true positive is detected only if the detected shot transition overlaps with the ground truth transition (3, 4 in green). A false positive is detected, if the predicted transition has no overlap with the ground truth (1, 4 in red) or the transition is detected for the second time (3 in red). A false negative is detected if there is no transition overlapping with the ground truth (1, 2 dotted) – the ground truth transition is missed.



**Figure 1.5:** Visualization of the evaluation approach. Predicted transitions shown with solid and missed with dotted rectangles. Figure taken from [23].

### 1.2.3 Training Details

The training samples are generated on demand by randomly sampling two videos, taking first not yet selected shot from both videos, and joining the shots by a random type of a transition. Only transitions considered for training are hard cuts and dissolves. The position of the transition is generated randomly. For dissolves, also its length is generated randomly from the interval  $[5, 30]$ . The length of each training sequence  $N$  is selected to be 100 frames. The size of the input frames is set to  $48 \times 27$  pixels.

For each frame, the network learns to predict whether there is a transition between the current frame and the next frame. Even in the case of dissolves, when the transition is over multiple frames, the network is trained to predict only the middle frame as a shot boundary. Negative training samples with no transition are not used since the network learns it from the no-transition segments of the input sequence.

The proposed architecture contains the following meta-parameters. We instigate the best meta-parameter setting by a grid search and report the results in Section 1.2.5.

1.  $S$  – the number of DDCNN cells in a SDDCNN block,
2.  $L$  – the number of SDDCNN blocks,
3.  $F$  – the number of filters in the first SDDCNN block (doubled in each following SDDCNN block),

<sup>2</sup>The original source code of the evaluation method from Baraldi et al. is available at <http://imagelab.ing.unimore.it/imagelab/researchActivity.asp?idActivity=19>

4.  $D$  – the number of neurons in the dense layer.

Prior training, weights are initialized by Glorot initializer [73], biases are initialized by zeros. A batch size of 20 was used for all investigated networks. To prevent over-fitting to synthetically generated transitions, the networks are trained only for 30 epochs, each with 300 batches resulting in 180,000 transitions in total. The best model is selected according to its performance on the validation set. We use Adam optimizer [74] with the default learning rate 0.001 and cross-entropy loss function. Depending on the architecture, the whole training takes approximately two to four hours to complete on a single Tesla V100 GPU.

### 1.2.4 Prediction Details

The network predicts the likelihood of a transition for all  $N = 100$  input frames. During validation and testing, only predictions for middle 50 frames are used due to incomplete temporal information for the first/last frames. Therefore, when processing a video, the input window is shifted by 50 frames between individual forward passes through the network. At the start and the end of a video, the first frame and the last frame respectively are duplicated 25 times to pad the video to ensure no unexpected transitions are generated at the video’s ends.

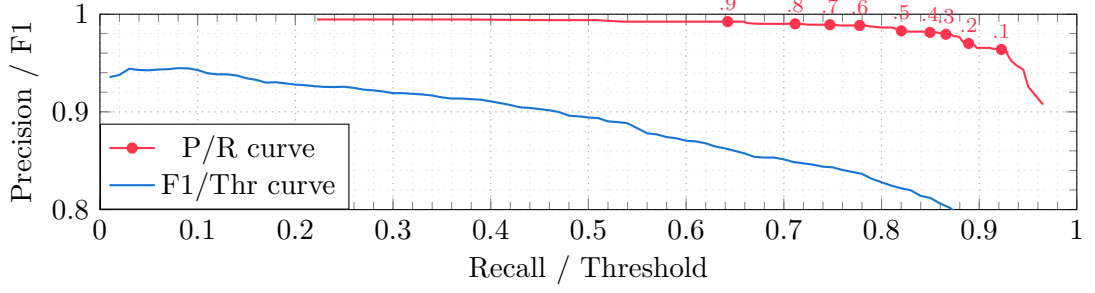
For a video, a list of shots is constructed in the following way: A shot starts at the first frame when the predicted likelihood of a transition drops below a threshold  $\theta$  and ends at the first frame when the predicted likelihood exceeds  $\theta$ . Since the network is trained to predict only one transition frame per any transition, even in case of long dissolves, we lower the acceptance threshold  $\theta$  to 0.1 instead of using the common 0.5 in all our experiments as it performed reasonably well for most of the models.

### 1.2.5 Results

As already mentioned in Section 1.2.3, the grid search is performed on four main meta-parameters of the architecture. In Table 1.1 F1 scores of investigated models are reported for validation (IACC.3) and test (RAI) datasets. Based on the evaluations, the best performing model is considered the one with 16 output filters in every convolution operation in the first SDDCNN block, two DDCNN

Dataset	F8 L2 S1 D128	F8 L2 S2 D128	F8 L2 S3 D128	F8 L3 S1 D128	F8 L3 S2 D128	F8 L3 S3 D128	F8 L4 S1 D128	F8 L4 S2 D128	F8 L4 S3 D128	F16 L2 S1 D256	F16 L2 S2 D256	F16 L3 S1 D256	F16 L3 S2 D256	F16 L4 S1 D256	F16 L4 S2 D256
IACC.3	71.0	70.9	72.0	72.0	71.7	71.6	70.4	71.4	69.5	<b>73.4</b>	71.6	<b>72.7</b>	<b>73.1</b>	71.6	69.9
RAI	92.9	<b>94.4</b>	93.6	93.4	93.1	93.8	93.4	<b>94.4</b>	91.9	92.2	93.6	91.4	<b>94.0</b>	91.8	92.9

**Table 1.1:** Meta-parameter grid search results on validation (IACC.3) and test (RAI) datasets. Reported values are the F1 score in percents with three top-performing models in bold. Data taken from [22].



**Figure 1.6:** Precision/Recall curve for the best performing model with corresponding thresholds  $\theta$  next to the points (in red) and F1 score dependency on the threshold (in blue). Measured on the RAI dataset. Figure taken from [23].

cells in every one of the three SDDCNN blocks, and with 256 neurons in the dense layer ( $F=16$ ,  $L=3$   $S=2$ ,  $D=256$ ).

Since the validation dataset contains various sequences of frames where even annotators are not sure whether there is a shot transition, the reported scores for the validation data are lower. Besides, even the top-performing TransNet models face problems with the detection of some transitions, for example, false positives in dynamic shots and false negatives in gradual transitions. On the validation dataset, the selected model detected 1058 false positives and 679 false negatives with respect to the annotation. This is in contrast to the RAI dataset results reported in Table 1.2, where the network achieves a lower number of false positives than false negatives. Based on manual inspection of the videos, we conclude that the RAI videos do not contain many highly dynamic shots (i.e. resulting in false positives) compared to the IACC.3 validation set while containing difficult dissolves spanning over dozens of frames (i.e. resulting in false negatives).

The performance comparison of related works is shown in Table 1.3. The average F1 score 94% of our top-performing model on the RAI dataset is on par with the score reported by Hassanien et al. [54]. The overall F1 score even slightly outperforms the work of Hassanien et al., even though they proposed a network with more than 40 times as many parameters trained for a larger set of transition types. Furthermore, our model has the advantage that no additional post-processing is needed.

Video	#T	TP	FP	FN	P	R	F1
V1	80	57	2	23	96.6	71.3	82.0
V2	146	132	5	14	96.4	90.4	93.3
V3	112	111	4	1	96.5	99.1	97.8
V4	60	59	5	1	92.2	98.3	95.2
V5	104	101	8	3	92.7	97.1	94.8
V6	54	53	3	1	94.6	98.1	96.4
V7	109	103	1	6	99.0	94.5	96.7
V8	196	181	4	15	97.8	92.3	95.0
V9	61	55	2	6	96.5	90.2	93.2
V10	63	57	0	6	100.0	90.5	95.0
Overall	985	909	34	76	96.4	92.3	94.3

**Table 1.2:** Per video results on the RAI dataset. For each video, the total number of transitions (#T), true positives (TP), false positives (FP), false negatives (FN), precision (P), recall (R) and F1 score (F1) are shown. Table taken from [23].

	Baraldi et al.	Gygli	Hassanien et al.	ours
average	84 [55]	88 [66]	<b>94</b> [54]	<b>94</b>
overall	-	-	93.4 [54]	<b>94.3</b>

**Table 1.3:** The Average and overall F1 scores for the RAI test dataset of the best architectures. The overall F1 scores are computed by calculating precision and recall over the whole dataset, not just a single video. Table taken from [23].

## 1.3 TransNet V2

The original TransNet network, as described in the previous section, has a set of limitations. In this section, we discuss them in detail and propose changes to mitigate them. In the next section, we thoroughly evaluate and discuss the proposed solutions. The contributions in this Section are presented in paper [25].

### 1.3.1 Limitations of TransNet

Shots for TransNet training are created artificially without taking into account their real distribution in the wild, aside from focusing on the most prevalent types of transitions. Even though it is convenient, automatically constructing training samples has multiple downsides. Commonly, in the real videos, subsequent shots share the same scene only captured from a different angle by another camera or at a different time. These shots can have very similar features such as color histograms, which makes them impossible to detect by such simple features. In TransNet training, as shots are concatenated randomly, the concatenated shots often do not share semantic meaning across the shots – the shots can be completely arbitrary – which does not force the network to learn more advanced features needed for difficult transitions in the real distribution. Another problem is the selection of the segments/shots. In the case of the IACC.3 dataset, they are detected automatically by a shot detection algorithm, which itself has false negatives and false positives. The false negatives do not present a challenge since they are scarce, and the probability of sampling undetected transitions is low as the actual shots are usually many seconds long. However, false positives in high dynamic scenes mean that such hard negatives are missing in the dataset. Since the dynamic scenes are probably the hardest for any detection algorithm, it may be needed to manually label at least some dynamic scenes and use them as hard negatives. This approach was taken, for example, by Hassanien et al. Finally, the last problem is that the artificially created dataset contains only a fixed set of selected transition types. However, not many types of transitions are commonly used, so this does not present a big problem.

The datasets used for TransNet validation and testing are very limited in size as well as very limited in transition types presented in the data. Also, the validation dataset is created by a single person without any peer review or independent verification, and the videos themselves contain mostly user-generated content of poor quality that no longer reflects the current state of user-generated videos. Nowadays, many cell phones contain high-resolution cameras with high dynamic range (HDR) support and image stabilization. HDR suppresses over-exposures and under-exposures that commonly resulted in false positives. Optical image stabilization and advanced digital stabilization [75] reduces handshake very prevalent in content from older devices. Further, professional video equipment that produces even less of such artifacts is becoming ubiquitous in amateur video production. Our validation and test sets should also reflect that.

With the automatically precisely generated transitions without any compression and resizing artifacts, we see rapid over-fitting even already after a few hundred batches, and the technique of early stopping needs to be employed. That means the model is not trained until convergence, but only until the performance

on the validation set stops improving. While training the model further improves loss and performance on the synthetically generated datasets, it harms performance on the real data. To mitigate the over-fitting, there have been developed many techniques such as L2 regularization or dropout [76]. These techniques impose restrictions preventing the model from being stuck in bad local minima and forcing the model to focus on all activation values instead of only a small discriminative set that is sensitive to noise.

Another approach to over-fitting is to vary input data to make models robust to such changes. In the image domain, there have been developed many techniques for data augmentation revolving around contrast, color or brightness changes, input masking [77], and others. Recently, even reinforcement algorithms were used to create the best augmentation methods for a certain task [78]. In our work, however, it is necessary to augment not only video frames but also the transition generation process itself. Such augmentation is unique to shot boundary detection task, and not much work has been done in this area since only a handful of works use automatically generated datasets. We discuss our solutions for the input data generation and augmentation in the next paragraphs.

### 1.3.2 Datasets and Data Augmentation

Unlike Hassanien et al. [54], we refrain from a fixed pre-generated dataset, which enables us to employ various types of augmentation. We also utilize recently published large manually annotated shot boundary dataset ClipShots. We describe the dataset and augmentation methods, including their technical details, in this Section.

#### Large Dataset

With the introduction of ClipShots [67], a dataset purposefully collected for shot boundary detection, we no longer have to rely on automatic transition generation since the dataset contains 166,756 manually annotated transitions. Hard cuts consist of 77 percent of the dataset, while the rest is gradual transitions, including dissolves and wipes. For training, we extract 160 frames long segments, each with a transition in the middle, then during training, a randomly cropped segment of length  $N = 100$  is used. This way we ensure each training segment contains a transition. We assume hard negatives are contained in these segments, and we do not explicitly train the network on sequences without any transition.

However, as reported in the results section, interestingly, training only on real data does not achieve the best performance. We therefore also utilize both IACC.3 and ClipShot datasets for automatic transition generation. Note the manual annotation of ClipShots does not have the problem of false positives in dynamic scenes, as discussed in Section 1.3.1; therefore, we could also benefit from that fact compared to training only on IACC.3. For the train set, we extract 300 frames long segments from each scene from the start, the middle, and the end of the scene while skipping some, if the scene is shorter. For scenes shorter than 300 frames, we store the whole scene. During training, we select two random segments and randomly crop them to the length of 100 frames and join them by a random transition at a random position. If a segment is shorter than 100

frames and the position of the transition means the final transition sample would be shorter than 100 frames, the sample is discarded and not used for training.

Aside from our original IACC.3 100 video validation set, we use 457 videos from the official ClipShots train set for validation. For testing the official ClipShots test set [67], BBC Planet Earth documentary series [45] and RAI [55] datasets are used. Again only predictions for 50 middle frames from the whole 100 frame input sequence are used to eliminate errors due to limited context.

## Shot Augmentation

We apply standard image augmentation to each shot with all images in the shot being augmented the same way in order not to create random color changes in a single shot. When generating transition artificially, we augment the shot prior to the shot joining. Firstly shot frames are flipped left to right with probability 0.5 and top to bottom with probability 0.1. Further, standard TensorFlow image operations adjusting saturation, contrast, brightness, and hue are utilized. Saturation and contrast of a shot are changed by a random factor from range  $[0.8, 1.2]$ . Brightness and hue are changed by random delta from range  $[-0.1, 0.1]$ . We also use *Equalize*, *Posterize* and *Color* operations from Python image library PIL<sup>3</sup>. Each operation is applied with probability 0.05, *Posterize* randomly keeps four to seven bits of the original color, *Color* is applied with random factor from range  $[0.5, 1.5]$ .

## Transition Types

Similarly to the original TransNet, we generate hard cuts and dissolves. We generate 50% of hard cuts and 50% of dissolves, the length of each dissolve is randomly uniformly selected from the set of even lengths  $\{2, 4, \dots, 28, 30\}$ . We generate only even lengths of dissolve transitions so that the ground truth position of the transition is exactly defined (for each frame we predict whether there is a transition from the current frame to the next frame, i.e. in case of odd lengths the transition can be either to the left or to the right of the middle frame).

<sup>3</sup><https://pillow.readthedocs.io>, re-implemented in TensorFlow at <https://github.com/tensorflow/tpu/blob/master/models/official/efficientnet/autoaugment.py>.



**Figure 1.7:** Examples of additional transition types. Standard wipe (left), flower scene sliding in while church scene sliding out (middle) and flower scene sliding in while church scene stationary (right).



As the ClipShots test set contains also wipes, we experiment with adding wipes into possible transition types. In 5% of dissolves, we generate wipe instead of the dissolve. We consider both horizontal and vertical wipes. We also consider sliding in the entering scene, sliding out the exiting scene, or both. See Figure 1.7 for illustrations of different types of wipes. However, we observe no improvement in performance with wipes in the train set; therefore, we refrain from generating them.

### Color Transfer

To force the network to learn more advanced local features instead of simple global features, we introduce shot color augmentation technique we call color transfer. Given two shots we transfer color from one shot  $s_1$  to the other shot  $s_2$  by first transforming both shots to CIE Lab color space, then we compute the new shot  $s'_2$  by the following equation:

$$s'_2 = \frac{\sigma_1}{\sigma_2}(s_2 - \hat{s}_2) + \hat{s}_1$$

where  $\hat{s}_i$  is mean and  $\sigma_i$  standard deviation of pixel values for respective shots. The equation is applied pixel-wise on each of the three Lab channels independently. Finally, we transform the new shot back to RGB color space. An example of the color transfer can be seen in Figure 1.8. During training, the color transfer is applied randomly to 10% of generated input sequences.

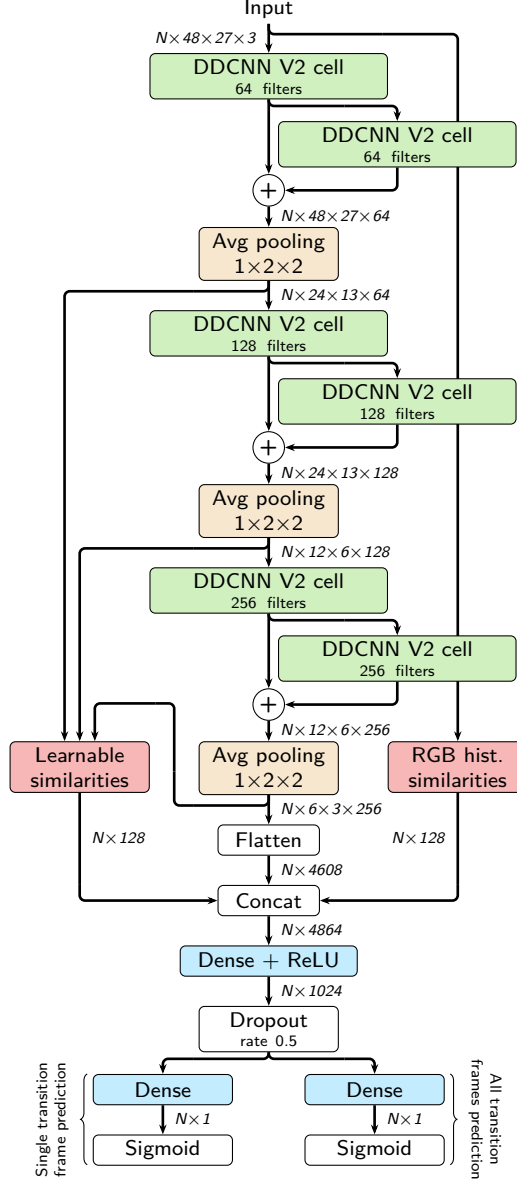


**Figure 1.8:** Example of color transfer augmentation technique between two shots.

### Suppressing False Positives

We consider adding two types of special augmentation to reduce false positives caused by handshake and rapid change of illumination e.g. by a passing object in front of a light source. Handshake is applied to five percent of train sequences by randomly removing the top (or bottom)  $k \in \{1, \dots, 5\}$  pixels from the first  $m$  frames and removing bottom (or top respectively)  $k$  pixels from the subsequent  $N - m$  frames. Finally, the frames are bilinearly resized to their original shape. Illumination change is applied to five percent of train sequences by performing the standard shot augmentation to only part of the sequence.

As the RAI dataset contains multiple sequences where color is changed between two subsequent frames, the illumination augmentation slightly improves the results. However, the opposite is seen on ClipShots and BBC datasets, where the

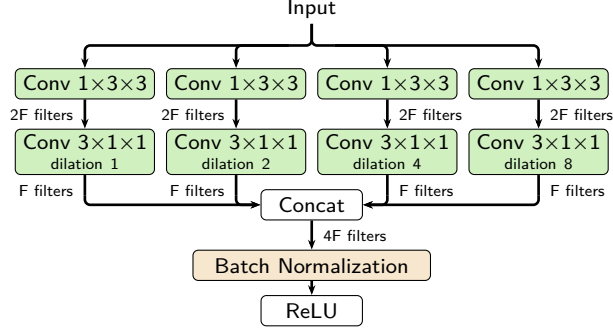


**Figure 1.9:** TransNet V2 shot boundary detection network architecture. Note that  $N$  represents the length of a video sequence, not batch size.

addition of this type of augmentation creates more false negatives than it creates true negatives since these phenomena are not prevalent in these test sets. Therefore in the final model training, we use neither artificial illumination changes nor handshake augmentation. Also, further manual inspection reveals the network can learn to suppress flashes purely from unaugmented data (Figure 1.15A).

### 1.3.3 Architecture Improvements

Our TransNet V2 is based on the original TransNet network with three SDDCNN blocks, each with two DDCNN cells. However, we make a wide range of changes that substantially improve the network’s performance. A schema of the TransNet V2 network is shown in Figure 1.9, and all the changes are described in detail in the following paragraphs.



**Figure 1.10:** DDCNN V2 cell with 4F filters.

### Convolution Kernel Factorization

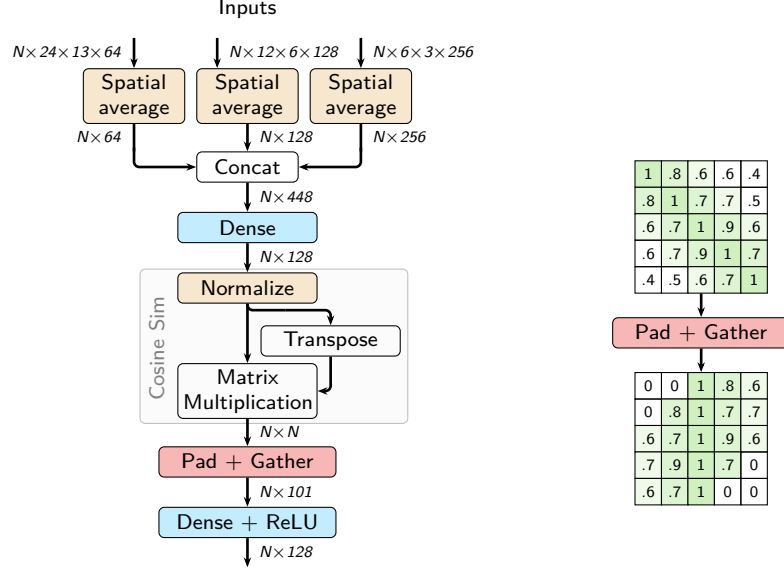
The TransNet benefits from using four decoupled convolutions instead of a single one since the decoupling reduces the number of parameters and the network is less prone to over-fitting, and also speeds up the computation. We further investigate how to factorize the convolution kernel to reduce over-fitting while preserving the benefits of a large field of view. In the image domain, depthwise separable convolutions have been introduced. The depthwise spatial convolution acts on each input channel separately and is followed by a pointwise (the standard  $1 \times 1$ ) convolution, which combines the resulting output channels. This way, the network is limited to learn only factorizable kernels; however, Chollet [79] shows it improves classification performance of InceptionV3 network [80] on ImageNet [81].

In video domain, Xie et al. [64] disentangles 3D  $k \times k \times k$  convolutions into 2D  $k \times k$  spatial convolution and 1D temporal convolution with kernel size  $k$  to improve I3D’s [62] performance on multiple datasets. Practically the separable 3D convolution can be implemented by two standard 3D convolutions with kernel shapes  $1 \times k \times k$  for the spatial and  $k \times 1 \times 1$  for the temporal convolution. This factorization of the convolutional kernel forces the network to learn to extract image features in the first step and to compare them temporarily in the second step. It also potentially reduces number of trainable parameters –  $N_{in} \times k^2 \times F$  for the spatial kernel and  $F \times k \times N_{out}$  for the temporal kernel compared to standard  $N_{in} \times k^3 \times N_{out}$ . If the number of input filters  $N_{in}$  is the same as output filters  $N_{out}$ , then if the number of filters  $F$  in between the spatial and temporal convolution is smaller than  $N_{out} \cdot k^3 / (k^2 + k)$ , the number of trainable parameters of the separable 3D convolution is lower than the number of parameters of the standard convolution. For kernel size  $k = 3$  we may select any  $F < 2.25N_{out}$  while still lowering the parameter count.

In our case, we observe that setting  $F = N_{out}$  is too extreme parameter reduction, hampering the performance of the model. However, setting  $F = 2N_{out}$  improves the performance substantially. Figure 1.10 shows the new version of the DDCNN block with factorized convolutions.

### Frame Similarities as Features

As already discussed in the related work section, many methods extract individual frame features and use them to compute similarity scores between consequent



**Figure 1.11:** Learnable frame similarities computation with visualization of Pad + Gather operation (right).

frames. The scores are then used as an input to a machine learning model such as SVM that predicts the likelihood of transition purely from the similarity scores. Similarly, we improve the performance of TransNet’s final fully-connected classifier by enriching the convolutional features by frame similarities. We consider frame similarities computed both from handcrafted and learnable features. A simple RGB color histogram with  $8^3 = 512$  bins is used for the handcrafted features. For the learnable features, we take outputs of all SDDCNN blocks. Those outputs of shape  $N \times H_l \times W_l \times C_l$  are spatially averaged into vectors of shape  $N \times C_l$  and vectors from different levels  $l$  are concatenated thus every one of the  $N$  frames is represented by vector  $v_i$  of length  $\sum_{l=0}^L C_l$ . A single linear layer without activation function is applied to reduce the dimension of the vector  $v_i$  to 128.

For each frame, we compute cosine similarities of its handcrafted or learned features with fifty frames preceding the frame and fifty frames following the frame. The similarities are transformed by a single fully connected layer with the ReLU activation function into 128-dimensional space and are concatenated with the convolutional features. If not all of the fifty preceding or following frames are available due to the limited length  $N$  of the input sequence zeros are used instead. Figure 1.11 depicts such computation for the learned features.

## Shortcuts

Some transitions are easy to detect by simple differencing in a minimal number of layers, others need the full depth of the network to be detected. We add a residual connection in each Stacked DDCNN block from the output of the first DDCNN cell in the block to the input of the spatial pooling operation.

## Batch Normalization

Neural network training is dependent on the initialization of the network’s parameters. If parameters are not initialized carefully, activations inside the network can rise to extreme values, which in turn can result in getting stuck in poor local minima. We employ batch normalization technique [82] that normalizes outputs of each layer by mean and variance computed from the input batch. It allows us to worry less about the learning rate and be less careful about parameter initialization. More importantly, it also acts as a regularizer, so the network is less prone to over-fitting.

## Multiple Classification Heads

The original TransNet predicts every transition as a single frame transition, no matter the length. This representation has its benefit that (almost) every frame belongs to a scene – no frames are ‘inside’ a transition. Further, if two transitions are very close to each other, there is no ambiguity if it is a single long transition or two separate transitions. However, this representation does not provide information on transition length. Therefore we add a second prediction head that predicts all in-transition frames instead of only the middle one. In some of our preliminary experiments adding the second head slightly improves performance. We hypothesize the network can learn more easily what constitutes as a transition and how long the transition is. Note that even though the information on the length of a transition is readily available, we do not use it in any way in the evaluation phase. However, we believe utilizing this information could, for example, eliminate many double detections when a single long transition is detected multiple times.

### 1.3.4 Other Changes

#### Optimizer

In recent years there have been countless efforts to improve the standard stochastic gradient descent optimization method. RMSProp [83] and Adam [74] have become some of the most widely adopted optimizers with both utilizing estimates of gradients’ second moments. However, many state-of-the-art works in the image domain [78, 84] refrain from using them and instead use rather primitive stochastic gradient descent with momentum while achieving better results. In our task, we observe that Adam optimizer is more effective in minimizing the loss than SGD with momentum; however, we see performance improvements on validation and test sets when using SGD with momentum compared to Adam.

#### Regularization

Artificially generated training data distribution is inherently different from the real data distribution. To reduce generalization error, we restrict the model’s parameter space by also minimizing the square of  $L_2$  norm of the parameters. Such action increases error on artificially generated validation data; however, it greatly reduces error on real data. Given the work of Loshchilov et al. [85], we also investigate whether decoupling the  $L_2$  regularization from optimizer step

computation – a technique known as weight decay – affects model’s performance. While  $L_2$  regularization and weight decay are (after hyper-parameter re-scaling) equivalent in the case of vanilla SGD, in more advanced optimizers, gradient moments estimations are affected by the square of  $L_2$  norm of parameters in the loss. Loshchilov et al. argue this phenomenon is responsible for the inferior performance of Adam. However, in our case, when using SGD with momentum, we do not see any significant difference between  $L_2$  regularization and weight decay.

Another form of regularization we employ is Dropout [76]. It addresses the over-fitting problem by randomly dropping units from the neural network during training. In TransNet V2, we drop activations of the first fully-connected layer with a probability 0.5.

### Class Imbalance

The main classification head of the network is trained with only one frame labeled as transition and the other frames as no transition. In the case of long dissolve transitions, missing the middle transition frame even by a single position results in a large penalty; therefore, the network is inclined not to predict these transitions with high confidence at all. The original TransNet solves the problem by adjusting the decision threshold to 0.1. We look at the problem more generally and weight the positive class by  $\gamma$ . Based on the exact task and dataset at hand, we may increase or decrease  $\gamma$  to increase recall or precision respectively. For example, in retrieval, 100% recall is crucial, whereas high precision may be preferred elsewhere. In our experiments,  $\gamma = 5$  is used.

### Loss Function









To train both classification heads we use the standard cross entropy loss (Equation 1.1) with sequence predictions  $\hat{y}$ , sequence ground truth  $y$  and possibly a weight  $w$  of the positive *is-a-transition* class. Since we predict likelihood of a transition for every frame in an input frame sequence  $\mathbf{x}$  of length  $N$ , the length of  $\hat{y}$  and  $y$  vectors is also  $N$ .

$$\mathcal{L}(\hat{y}, y, w) = - \sum_{i=1}^N \left[ w y_i \log \hat{y}_i + (1 - y_i) \log(1 - \hat{y}_i) \right] \quad (1.1)$$

The joint objective is shown in Equation 1.2. The first term is the loss of the classification head  $f^s$  predicting single transition frame for arbitrarily long transition, the second term is the loss of the classification head  $f^a$  predicting all transition frames and is weighted by  $\lambda_a = 0.1$ . These loss terms are averaged over the batch. Last term of the objective is  $L_2$  regularization of model’s parameters  $\theta$  weighted by  $\lambda_p = 0.0001$ .

$$\min \frac{1}{|\mathcal{B}|} \sum_{(\mathbf{x}, y^s, y^a) \in \mathcal{B}} \left[ \mathcal{L}(f^s(\mathbf{x}; \theta), y^s, \gamma) + \lambda_a \mathcal{L}(f^a(\mathbf{x}; \theta), y^a, 1) \right] + \frac{\lambda_p}{2} \sum_{\vartheta \in \theta} \|\vartheta\|_2^2 \quad (1.2)$$

In Figure 1.12 we show a sample frame sequence  $\mathbf{x}$  with both immediate and gradual transition and its two types of ground truth vectors  $y^s$  and  $y^a$  for the two classification heads. Note that artificially generated train frame sequences

$\mathbf{x}$	=								
$y^s$	=	0	0	1	0	0	0	1	0
$y^a$	=	0	1	1	1	1	0	1	0

**Figure 1.12:** Frame sequence with its two types of ground truth.

contain only a single transition per sequence; however, when training with real data, it is possible to have multiple transitions in a single frame sequence if the transitions are close to each other.

## 1.4 Experiments

In this section, we describe in detail the training process of the TransNet V2 model. Further, we compare our model to related works, also detailing our reevaluation process for it. Lastly, an ablation study is made to justify our decisions in making TransNet V2.

### 1.4.1 Training Details

We initialize weights of all convolution operations by He normal initializer [86]. Fully connected layers are initialized by TensorFlow’s default Glorot initializer [73]. Biases are initialized by zeros. We optimize the loss function (Equation 1.2) by stochastic gradient descent with momentum set to 0.9 and fixed learning rate 0.01.

The standard TransNet V2 model is trained only by artificially generated sequences from IACC.3 and ClipShots datasets as described in Section 1.3.2. Fifty percent of transitions we generate are hard cuts and fifty percent are dissolves of max length 30 frames. Besides, we consider training TransNet V2 by 15% of real transitions from the ClipShots dataset, 35% of automatically generated hard cuts, and 50% of automatically generated dissolves. Input to the network is frame sequences of length  $N = 100$  with each frame of size  $48 \times 27$ .

We train the network for 30 epochs, each with 750 batches of size 16. In case the network is trained also in part by real manually annotated transitions from the ClipShots dataset, we add 20 additional epochs, i.e. the network is trained in total by 600,000 transitions. Training by artificial data only longer than 30 epochs is unnecessary since the network then overfits. The best performing model on our ClipShots validation set is selected. Together with validation, the training takes approximately 17 hours on a single Tesla V100 16GB GPU. TensorFlow deep learning library has been used for all the experiments.

### 1.4.2 Results

We report results of TransNet V2 trained both with artificial data only and with 15% of real data. Additionally, we report the results of the original TransNet retrained by our data generation pipeline (Section 1.3.2). For each network we show mean F1 scores<sup>4</sup> and their standard deviations on the test sets in Table 1.4. The statistics are computed from three best epochs from each of three independent runs as measured on the ClipShots validation dataset. We see a clear dominance of TransNet V2, especially on the ClipShots dataset. Utilizing real transitions improves results on BBC Planet Earth dataset while it harms performance on ClipShots and especially on RAI. We further discuss this phenomenon of better results with artificially generated data compared to the real data in the next section.

We compare TransNet V2 to related work in Table 1.5. For TransNet models, we report the F1 score of the best model selected out of the nine instances based on

---

<sup>4</sup>The same evaluation metric as in Section 1.2.2 is used. However, due to minor errors in ground truth of some test sets, we also count correctly any detection that misses ground truth by at most two frames. With correct ground truth its effect compared to the original metric is minimal.



Model	ClipShots	BBC	RAI
TransNet	$71.9 \pm 3.5$	$94.0 \pm 0.6$	$93.1 \pm 0.6$
TransNet V2	<b><math>77.5 \pm 0.3</math></b>	$95.1 \pm 0.5$	<b><math>93.2 \pm 0.9</math></b>
TransNet V2 (15% real transitions)	$77.0 \pm 0.8$	<b><math>96.5 \pm 0.5</math></b>	$91.2 \pm 1.1$

**Table 1.4:** Comparison of the original TransNet and TransNet V2. Mean F1 scores and standard deviations in percents. Computed from 9 model instances (3 best epochs of 3 independent runs as measured on ClipShots validation set).

its performance on the ClipShot validation set. To best of our knowledge, the best shot boundary detectors available, as reported by their authors, are DeepSBD by Hassanien et al., DSM by Tang et al., and our original TransNet. On RAI dataset these methods report F1 score 93.4%, 93.5%, and 94.3% respectively. However, Hassanien et al. report results on neither ClipShots nor BBC dataset. Tang et al. report results on ClipShots but their validation method differs from our method<sup>4</sup> described in Section 1.2.2 and for example incorrectly counts double detection. Therefore we reevaluate both shot boundary detection models and report only these results. We discuss reevaluation details in Section 1.4.3.

Model	ClipShots	BBC	RAI
TransNet	74.8	94.6	93.4
TransNet V2	77.5	95.8	<b>94.4</b>
TransNet V2 (15% real transitions)	<b>77.9</b>	<b>96.2</b>	93.9
TransNet <sup>†</sup>	73.5	92.9	94.3
Hassanien et al. [54]	75.9*	92.6*	93.9*
Tang et al. [67], ResNet baseline	76.1*	89.3*	92.8*

<sup>†</sup> The original TransNet as reported in Chapter 1.2 and in [23]. Reevaluated.

\* Our reevaluation with the best threshold. See Section 1.4.3 for more details.

**Table 1.5:** Comparison of TransNet V2 with related work. F1 scores in percents. In the case of TransNet entries, the model with the best F1 score on the validation set is shown.

TransNet V2 clearly outperforms related work on both ClipShots and BBC Planet Earth datasets, on the latter almost halving the error achieved by the previous state-of-the-art. On the RAI dataset, all methods perform comparably. Relative low performance on ClipShots can be attributed to the fact that it contains multiple seemingly unannotated videos or video parts. Further, in many cases, a frame is annotated as a transition incorrectly. We show some interesting transitions with our predictions in Figure 1.15 to illustrate our model’s strengths and weaknesses, in Figure 1.17 raw predictions are shown on a long video sequence comparing the original TransNet and TransNet V2. Here we list some of the main takeaways:

- Many times the ground truth incorrectly labels flash as a transition; however, the model can correctly ignore it. Nonetheless, if there is an illumination change in multiple subsequent frames, the model struggles (Figure 1.15A).

- Long transitions with custom animations or fade-ins with uncommon colors are usually missed by the model (Figure 1.15B, only selected frames from the transition shown).
- It can happen that the model misses a transition that is easily distinguishable for humans. However, sometimes these transitions are missing in the ground truth data (Figure 1.15C).
- The model struggles with heavy motion blur (Figure 1.15D).
- Many times it is subjective whether there is a transition (Figure 1.15E).

All in all, on ClipShots dataset the best instance of TransNet V2 achieves 5.7k true positives, 1.7k false positives, and 1.5k false negatives. This is in stark contrast to BBC Planet Earth dataset, where the model achieves almost six times as many false negatives as false positives (exactly 4537 true positives, 55 false positives, and 307 false negatives). Such discrepancy can be attributed to already mentioned missing annotations and many long, difficult, and sometimes ambiguous transitions in ClipShots dataset.

Concerning the RAI dataset, all models perform comparably with a slight exception of Tang et al. However, we refrain from making any conclusions as the dataset contains mainly TV shows with visual effects that account for possibly many ambiguities in the ground truth. For example, the frame sequence in Figure 1.13 is contained with slight variations in the dataset more than 20 times, and classifying it as transitions can result in almost doubling the number of false positives achieved by our method.



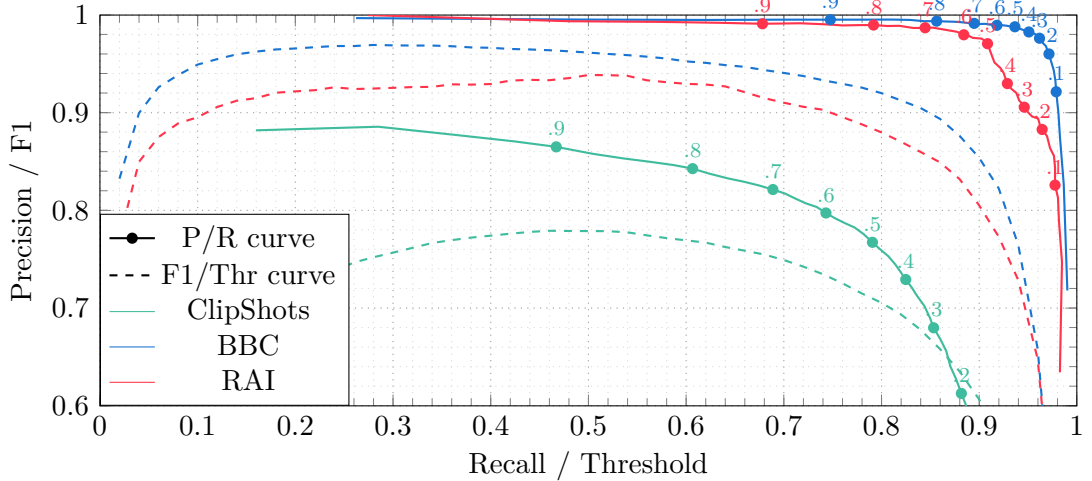
**Figure 1.13:** Ambiguous frame sequence from the RAI dataset labeled as *no-transition* in the ground truth data.

There are many more ambiguities similar to Figure 1.13; therefore we conclude it is necessary to adjust the model’s threshold or even training data to compensate for particularities of a task at hand in case the reader’s definition of a transition differs from our train and test data. For that reason, we show precision-recall curves and F1 scores as a function of the model’s threshold for our best model TransNet V2 trained with 15% real transitions on all test sets (Figure 1.14).

### 1.4.3 Related Work Reevaluation Details

Hassanien et al. train a neural network with an input of 16 subsequent frames to predict the likelihood of transition anywhere in between the input frames. The network is trained to distinguish between no transition, cut transition, and gradual transition. The network’s logit predictions are fed through SVM to predict transitions and color histograms are used to suppress false positives. However, the publicly available code<sup>5</sup> only contains the neural network model; therefore,

<sup>5</sup>Available at <https://github.com/melgharib/DSBD>.



**Figure 1.14:** Precision/Recall curve with corresponding thresholds next to the points (solid) and F1 score dependency on the threshold (dashed) for the best performing TransNet V2 model on all test sets.

we only use the model without any post-processing. As we do not distinguish between a cut and a gradual transition, we sum their probabilities into one class. To generate per-frame predictions, we assign the predicted transition probability to the middle 8 frames of the 16 frame input sequence and shift the input window every time by 8 frames. Note that our metric is independent of the length of predicted or ground truth transition as long as the prediction overlaps with ground truth at least at one frame. We compute the F1 score for thresholds  $\{0.1, 0.2, \dots, 0.9\}$  and use 0.9 as it produces on average the best F1 score on the test sets. Our method of generating transitions from the predictions achieves a similar result on the RAI dataset as the authors’ SVM and color histogram method (93.4% vs. 93.9%); therefore, we are confident that its results on ClipShots and BBC datasets objectively represent work of Hassanien et al. Note it is scientifically questionable to select the decision threshold directly on a test set as the achieved results can overestimate the actual model’s performance. However, we want to compare ourselves to the best model available; therefore, we accept possible bias in the results.

Tang et al. utilize a multi-step process for shot boundary detection. However, the only publicly available code<sup>6</sup> contains only their ResNet-18 baseline similar to the work of Hassanien et al. When confronting the authors about the code and questionable validation method, we were only pointed to the baseline code; therefore, we evaluate the ResNet-18 baseline and report its results. As the network’s input and output are the same as in the case of DeepSBD, the same evaluation method was used with the threshold of 0.8 achieving the best results.

#### 1.4.4 Ablation Study

We thoroughly investigate our individual design decisions in the following ablation study. Firstly we examine the effect of different types of training data. When conducting the first experiments with TransNet V2, we trained it purely

<sup>6</sup>Available at [https://github.com/Tangshitao/ClipShots\\_baseline](https://github.com/Tangshitao/ClipShots_baseline).



**Figure 1.15:** Example TransNet V2 predictions compared to ground truth from the ClipShots test set. The first line below a frame sequence indicates ground truth scenes. The second line indicates the model's prediction (transitions are represented by thick solid line segments, the dotted line means no transition). We indicate the correctness of the ground truth or the predictions by OK/NOK on the left. Some frames were dropped for clarity.

with ClipShots dataset’s manually annotated transitions. However, such an approach results in a huge performance drop on both ClipShots and RAI datasets compared to the artificially generated dataset with 50/50 split between hard cuts and dissolves. Therefore, we investigate which type of transition is the most responsible for the performance drop. We substitute half of the real transitions from the training set by hard cuts or dissolves. As can be seen in Table 1.6, training the model on 50% real data and 50% artificially generated hard cuts has almost no effect on the performance while training on real data and artificially generated dissolves bring the model’s performance close to the artificial-data-only model. Based on this finding, we conclude it is important for any future work to concentrate efforts on ensuring dissolves are appropriately represented in a train set. Such a conclusion is consistent with the findings of Tang et al., who deliberately extended the ClipShots dataset with additional gradual transitions after observing a weak performance of their model on these transitions.

Training data	ClipShots	BBC	RAI
100% real transitions	$66.4 \pm 1.3$	<b><math>96.3 \pm 0.5</math></b>	$86.4 \pm 1.2$
50% real, 50% cuts	$68.2 \pm 0.7$	<b><math>96.6 \pm 0.7</math></b>	$84.4 \pm 0.6$
50% real, 50% dissolves	$75.3 \pm 0.8$	<b><math>96.3 \pm 0.5</math></b>	$90.7 \pm 0.7$
15% real, 35% cuts, 50% dissolves	$77.0 \pm 0.8$	<b><math>96.5 \pm 0.5</math></b>	$91.2 \pm 1.1$
50% cuts, 50% dissolves	<b><math>77.5 \pm 0.3</math></b>	$95.1 \pm 0.5$	<b><math>93.2 \pm 0.9</math></b>

**Table 1.6:** Effects of real and artificially generated transitions on TransNet V2 performance. Mean F1 scores and standard deviations in percents.

Interestingly in Table 1.6 we also see that manually annotated transitions improve performance of the model on BBC Planet Earth – the dataset that contains high-quality content mostly with plain hard cuts without any peculiar gradual transitions, visual banners or animations commonly present in TV studio broadcast. This is in stark contrast to our initial assumption that real manually annotated transitions would be necessary to detect exotic types of animated transitions as in Figure 1.15B but less useful for hard cuts. However, we conclude these exotic transitions are quite uncommon even in the ClipShots dataset, and the added benefit of the real data revolves rather around improving the model’s performance on difficult hard cuts. Even though BBC Planet Earth dataset does not contain extremely dynamic shots, surprisingly, the dataset still includes many difficult hard cuts, as can be seen in Figure 1.16. Therefore we train TransNet V2 with 15% of real transitions, 35% of artificial hard cuts, and 50% of artificial dissolves, which still significantly improves model’s performance on BBC Planet Earth while achieving similar performance as the artificial-data-only model on the other test sets.

A natural question is whether our design changes to the original TransNet network are needed to achieve better performance. To answer this, we investigate multiple design decisions – namely the addition of frame similarity features to the final classifier, shortcuts that effectively halve the shortest path through the convolutional layers, and separable convolutions that factorize large  $3 \times 3 \times 3$  convolution kernels into spatial-only and temporal-only kernels. As seen in Table 1.7, all design decisions prove to be valuable, especially on more challenging





**Figure 1.16:** Difficult hard cuts from BBC Planet Earth dataset. Mind the resolution of the displayed frames is still more than 7 times larger than the network’s inputs size ( $48 \times 27$ ).

ClipShots dataset where the original TransNet struggles (Table 1.4).

Model	ClipShots	BBC	RAI
TransNet V2	<b>77.5</b> $\pm$ 0.3	<b>95.1</b> $\pm$ 0.5	93.2 $\pm$ 0.9
Without frame similarity	77.1 $\pm$ 0.5	94.8 $\pm$ 0.8	<b>93.5</b> $\pm$ 0.6
Without shortcuts	76.9 $\pm$ 0.7	94.8 $\pm$ 0.9	<b>93.4</b> $\pm$ 0.5
Without separable convolutions	76.9 $\pm$ 0.6	94.8 $\pm$ 1.4	93.0 $\pm$ 1.2

**Table 1.7:** Effect of individual components on the performance of TransNet V2. Mean F1 scores and standard deviations in percents computed from 9 model instances.

Similarly to the original TransNet, we investigate the effect of the network’s depth on performance (Table 1.8). TransNet V2 consists of three SDDCNN V2 blocks, each with two DDCNN V2 cells with skip connection and ended by spatial average pooling, i.e. in total  $3 \times 2$  DDCNN V2 cells. We increase the depth by adding one more SDDCNN V2 block with twice as many filters as the previous block (512 filters). We also test adding one DDCNN V2 cell in each of three SDDCNN V2 blocks while keeping the number of filters in all SDDCNN V2 blocks the same. Both variants do not perform as well as the  $3 \times 2$  variant. We hypothesize it is caused by easier over-fitting to the artificial train data – a phenomenon consistently seen in many of our tests when the number of parameters was increased.

Model	ClipShots	BBC	RAI
TransNet V2	<b>77.5</b> $\pm$ 0.3	<b>95.1</b> $\pm$ 0.5	<b>93.2</b> $\pm$ 0.9
$4 \times 2$ DDCNN V2 cells	77.2 $\pm$ 0.6	94.9 $\pm$ 1.1	91.5 $\pm$ 1.1
$3 \times 3$ DDCNN V2 cells	76.8 $\pm$ 0.8	94.2 $\pm$ 1.3	92.6 $\pm$ 0.9

**Table 1.8:** Effect of network’s depth on performance. Mean F1 scores and standard deviations in percents computed from 9 model instances.

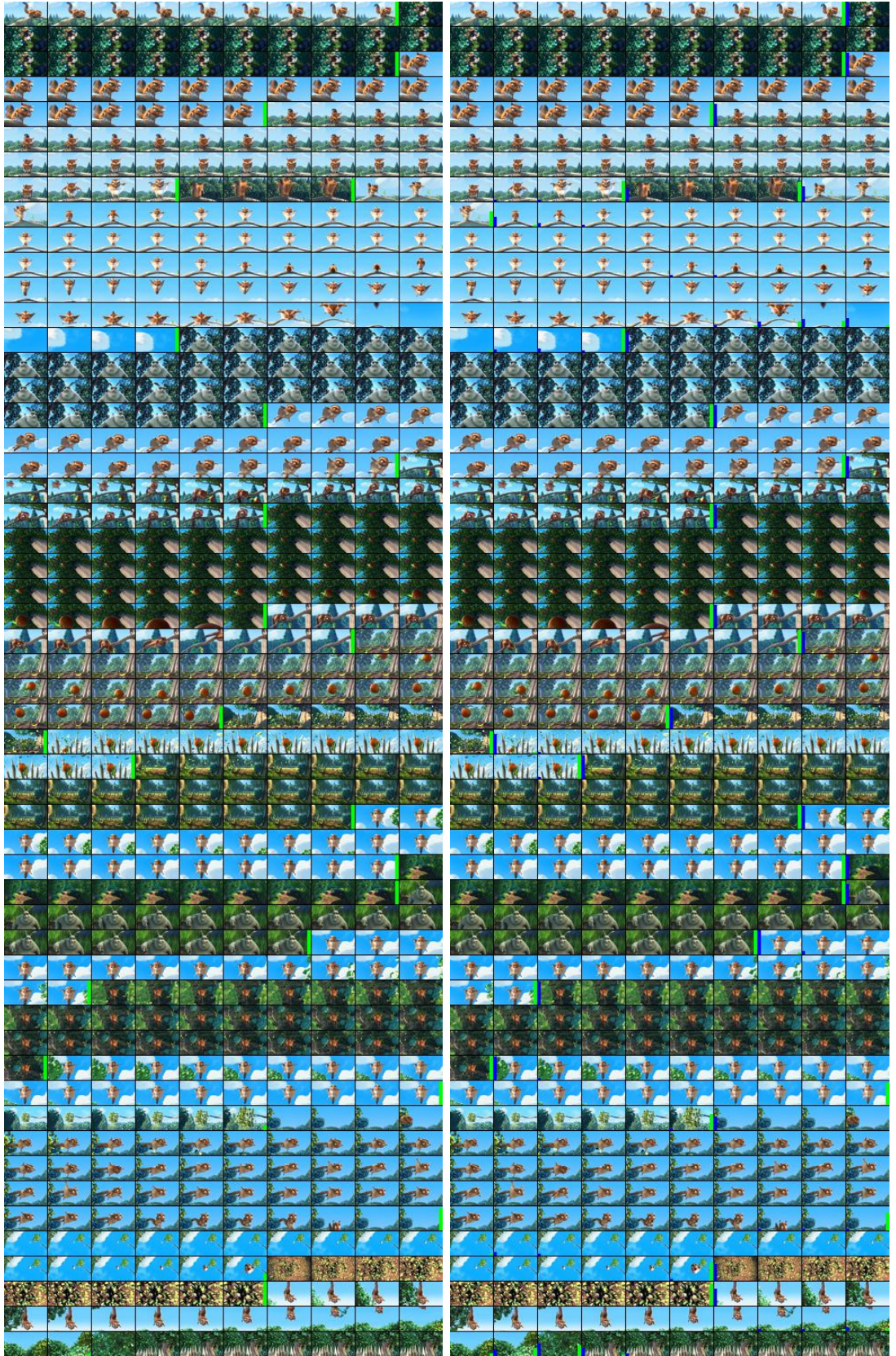
As already mentioned in Section 1.3.2, we train the model with additional augmentation techniques to see whether they are necessary to achieve good performance (Table 1.9). We observe that the addition of artificial camera shake

yields more false negatives than it reduces false positives. Similarly, we see that it is not necessary to generate more types of transitions than the standard hard cuts and dissolves to reach our state-of-the-art results. Finally, we test applying a random color transformation to part of the input frame sequence with a probability of five percent. We see slight improvement on the RAI dataset; however, there is no to a negative effect on ClipShots and BBC Planet Earth.

<b>Model</b>	ClipShots	BBC	RAI
TransNet V2	<b>77.5</b> $\pm$ 0.3	<b>95.1</b> $\pm$ 0.5	93.2 $\pm$ 0.9
Camera shake	76.9 $\pm$ 1.1	94.0 $\pm$ 0.8	93.2 $\pm$ 1.1
Wipe transitions	76.6 $\pm$ 0.6	94.5 $\pm$ 0.5	92.9 $\pm$ 1.0
Random color change	77.2 $\pm$ 1.0	94.5 $\pm$ 1.0	<b>93.8</b> $\pm$ 0.3

**Table 1.9:** Effect of augmentation methods on the performance. Mean F1 scores and standard deviations in percents computed from 9 model instances.





**Figure 1.17:** Visual comparison of TransNet (left) and TransNet V2 (right). Predictions are shown in green, in blue are shown predictions of the second head of TransNet V2. For example, a transition on the ninth line is detected by TransNet V2 but not by TransNet. The original video authored by Blender Foundation licensed under CC-BY. Some frames skipped due to limited space.



## 2. Text-Based Video Retrieval

With the inception and steady rise of large video collections containing a diverse range of video topics comes the need for fast efficient retrieval. However, even small to medium-size collections are impossible for humans to be browsed manually, especially in a timely manner. Over the years, many sketch-based methods, operating with various visual features, were developed to ease retrieval in these collections [12, 16, 20, 19]. Nonetheless, it is inherently difficult, especially for a non-skilled end user, to correctly reproduce a searched object by color, edge, or another sketch. Therefore, recently with advances in deep learning, an option to describe the searched object in natural language gained popularity [14, 17].

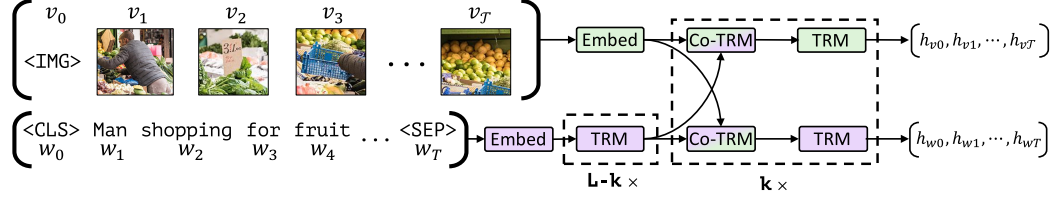
Retrieval in video collections by textual queries, also called ad-hoc video search (AVS), represents a great challenge as it is necessary to semantically understand the content of queries and videos. For example, a query *‘a person talking behind a podium wearing a suit outdoors after sunset’* requires the system to understand that person is not on the podium and *after sunset* means at night not while the sun is setting. Further, the same information needs to be extracted from the video to retrieve the scene. This example illustrates how AVS differs from classical concept-based retrieval – rather, it is a combination of concept and action detection, relation modeling, and text understanding.

Since 2016 the ad-hoc video search task is annually held at TRECVID with a goal to model the end-user search use-case. Each year’s task consists of 30 one-sentence queries for segments of video containing persons, objects, activities, locations, etc. and combinations of the former. Until 2018 IACC.3 dataset [72] of approximately 4600 Internet Archive videos with 600 hours of content was used. In 2019 a larger high-resolution V3C1 dataset [21] of 7475 Vimeo videos with 1000 hours of content was introduced and used onward.

In this chapter, the winning solutions of TRECVID Ad-hoc Video Search 2018 and 2019 are discussed, and a new system built upon work of the 2018 winner Li et al. [4] is proposed achieving results on par with the state-of-the-art on TRECVID AVS competition data from years 2016 and 2018. On TRECVID AVS 2017 data, the proposed system improves state-of-art by ten percent showing promising future research direction. Further, the proposed system also outperforms Li et al. on our in-house dataset with over two hundred challenging queries.

### 2.1 Related Work

With the introduction and broad availability of convolutional neural networks for image classification [56, 87, 80], many works repurposed those networks for concept extraction. For example, team NII-HITACHI-UIT [88], the winner of TRECVID AVS 2016, utilizes multiple readily available networks for concept detection, scene description as well as dense captioning; therefore, the AVS task is reduced into text-based retrieval task in which similarity scores between query text and video semantic features are computed using inverted index structure and TF-IDF weighting. These so-called concept-based retrieval systems are, however, limited to only fixed sets of concepts and cannot capture relations between the concepts as the only information available is *present* or *not present* with some



**Figure 2.1:** ViLBERT model [95] operates over image regions and text segments. Each stream is a series of transformer and co-attentional transformer layers enabling information exchange between modalities.

confidence measure.

One of the early works that tries to learn directly the matching between images and text is by Wang et al. [89]. It encodes images into 4096-dimensional vectors using VGG network [87]. Sentences are encoded using Word2Vec [7] vectors transformed into 18000-dimensional space using Fisher Vectors. A set of weights is then trained to minimize the distance between these two representations. Other work by Dong et al. [90] encodes sentences by concatenating their Bag-of-Words representations, Word2Vec vectors, and RNN outputs. These vectors are then passed through a multi-layer perceptron that is trained together with the RNN to project them into visual feature space created by e.g. ResNet-152. Aside from the text-to-image-space model trained and tested using Flickr30K dataset [91] (containing over 30 thousand images collected from the Flickr website with five captions each), the authors try to learn a text-to-video-space model utilizing MSVD dataset. The video features are extracted using the C3D network and averaged over time; however, better results are achieved by image-based ResNet-152 network instead of the 3D video network.

The above-mentioned approaches represent an image by a single vector that is considered as an aggregation of the important image regions. However, in the process of aggregation, some important information about individual regions can be lost. Lee et al. [92] introduce Stacked Cross Attention (SCAN), which performs text to image matching by combining similarities between individual image regions and a weighted sum of individual word vectors. Image region vectors are extracted using Faster R-CNN [93] pre-trained on Visual Genomes dataset [94] and one linear trainable layer is added to project the fixed region representations. Word vectors are produced by a trainable bi-directional GRU layer.

ViLBERT [95] network by Lu et al. (shown in Figure 2.1) takes attention-based approach a step further by repurposing BERT attention-based bidirectional language model [96]. The original BERT processes each word (token) of an input sentence independently – the only context is obtained through dozens of attention layers. With pretraining on a large language corpus, BERT and its variants achieved state-of-the-art results in many natural language processing tasks via transfer learning. ViLBERT’s input consists of image description and set of image region features that are extracted the same way as in the work Lee et al. The network is pre-trained on large Conceptual Captions [97] dataset to predict alignment score i.e. whether an image description matches a set of image regions’ features. In the end, the network is finetuned on a manually annotated high-quality Flickr30K dataset. The finetuning is performed by computing the

$$\min \sum_{i=1}^{|\mathcal{B}|} \left( \sum_{j \in \mathcal{N}(i)} [\delta + S(V_i, C_j) - S(V_i, C_i)]_+ + \sum_{j \in \mathcal{N}(i)} [\delta + S(V_j, C_i) - S(V_i, C_i)]_+ \right) \quad (2.1)$$

$$\max \sum_{i=1}^{|\mathcal{B}|} \log \left( \frac{e^{S(V_i, C_i)}}{e^{S(V_i, C_i)} + \sum_{j \in \mathcal{N}(i)} [e^{S(V_i, C_j)} + e^{S(V_j, C_i)}]} \right) \quad (2.2)$$

**Figure 2.2:** Max-margin ranking loss and noise-contrastive estimation loss.  $S(V_i, C_j)$  is a similarity score between video clip  $V_i$  and caption  $C_j$ ,  $\mathcal{N}(i)$  is a set of negatives for video/caption  $i$ ,  $\delta$  is the margin and  $[\cdot]_+$  is  $\max(0, \cdot)$ . In [8] authors allow for multiple positive pairs  $\mathcal{P}(i)$  due to uncertainty in video-caption alignment by substituting  $e^{S(V_i, C_i)}$  for  $\sum_{(x,y) \in \mathcal{P}(i)} e^{S(V_x, C_y)}$ .

alignment score for the correct image-text pair and other three sampled incorrect pairs. The scores are passed through softmax, and cross-entropy loss is computed. The twelve attention layers in ViLBERT outperform SCAN with only one attention in image retrieval on the Flickr30K dataset by almost 20% in terms of Recall@1 (recall at the first position) and by over 7% in Recall@10.

As mentioned by Dong et al., simply exchanging 2D features from networks such as ResNet for 3D features extracted by e.g. C3D is not sufficient to perform video retrieval. Moreover, they imply that video retrieval can be performed with better results by extracting only 2D features. However, it is unfortunate that, by operating only on single frames, some information about actions, etc. is inevitably lost. Work of Mithun et al. [98] argues that 3D video features such as I3D and 2D image features produced by e.g. ResNet focus on different characteristics and are complementary in some sense. The first focuses on action identification and the later on identifying objects in the frames. Therefore the authors introduce two spaces: activity-text space and object-text space. A text query is then evaluated in both spaces independently, and the fusion of those two rankings is performed. Nonetheless, work of Yu et al. [99] utilizing again only image-based ResNet-152 for feature extraction outperforms Mithun et al. by 45% for both Recall@1 and Recall@10 in video retrieval task on MSR-VTT [100] dataset (containing 10 thousand web video videos with 200 thousand short clips each paired with text descriptions).

### 2.1.1 Weakly Supervised Approaches

Learning mappings of text and videos into joint embedding space often requires large manually annotated datasets. Even though there are datasets such as MSR-VTT or Tumblr GIF (TGIF) [101], each with more than 100 thousand short clips (or gifs in the latter case) with their natural language descriptions, the work of Miech et al. [2] shows that even much larger datasets are greatly beneficial to video retrieval. The authors introduce HowTo100M dataset containing 136 mil-

lion video clips from 1.22 million narrated instructional web videos, each accompanied by automatically or human-generated audio transcriptions. Although the transcriptions need not to align with the visual content or they may be unrelated completely, a simple shallow network trained max-margin ranking loss (Equation 2.1) achieves state-of-the-art results on multiple benchmarks. For video, the network utilizes 2D and 3D features extracted by Resnet-152 pre-trained on ImageNet and ResNeXt-101 pre-trained on Kinetics respectively. The features are then aggregated by temporal max-pooling and concatenated. Transcriptions are embedded using Word2Vec and a shallow 1D convolutional network is used to extract fixed-sized representation. Further, a single linear fully-connected layer with a two-layer context gating function is used for both the visual and textual features. The authors stress that, during training, intra-video negative sampling strategy is critical for good performance – half of the negative pairs in the loss belong to the same original video to ensure that the learned embedding focuses on relevant aspects of the video clip.

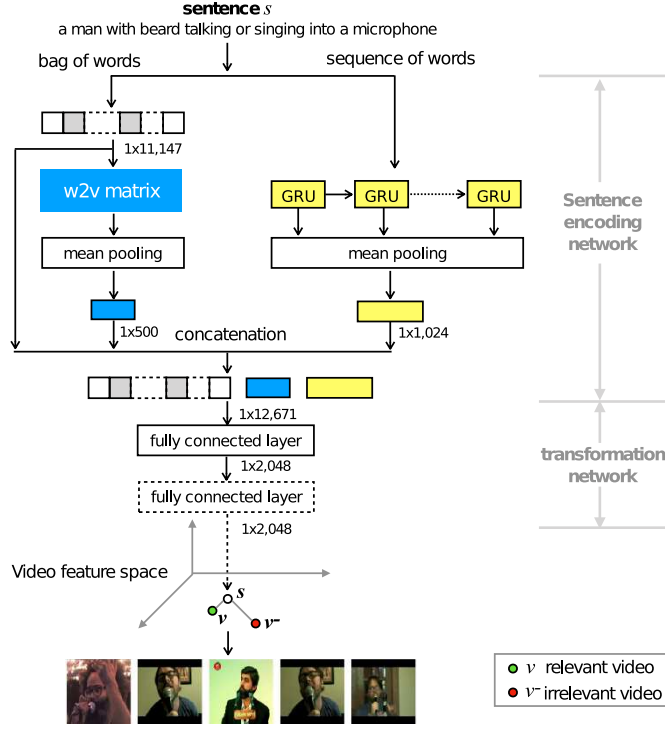
In video retrieval task on the MSR-VTT dataset, this network achieves very similar results as Mithun et al.; however, the network is only trained on HowTo-100M. Finetuning on MSR-VTT improves the results of Mithun et al. by more than 110% in Recall@1 and 77% in Recall@10. Additional work by the authors [8] shows the benefits of training video feature extractors (I3D or S3D) from scratch. Further it is suggested that noise-contrastive estimation loss ([102], Equation 2.2) outperforms standard max-margin ranking loss.

### 2.1.2 Winners of TRECVID Ad-hoc Video Search

This section summarizes approaches used by winning teams of TRECVID AVS challenge. Namely, it focuses on Dual Encoding by Dong et al. [103] and Word-To-Visual-Vector (W2VV++) by Li et al. [4] (shown in Figure 2.3) since these have been used in the winning submissions of 2018 and 2019 [104, 105, 106].

**Frame features.** Image-based 2D CNNs are used to extract features for individual frames. In particular Li et al. utilizes two CNNs: ResNeXt-101 from [107] and ResNet-152 from MXNet library [108] both trained on the ImageNet dataset with over 10 million images and over 10 thousand classes. Frames are resized to  $256 \times 256$  and CNN features (the output of the penultimate layer) are extracted from its 10 sub-images (i.e. five  $224 \times 224$  center and corner patches, all of them also horizontally flipped). The final frame features are the result of averaging those ten sub-image features and concatenation of ResNeXt-101 and ResNet-152 features.

**Video features.** For a given video clip frames are sampled every 0.5 seconds. Features for each frame are extracted as described above. Then the features are aggregated by one or more of the following approaches: (1) averaging features over time. (2) using a bidirectional recurrent neural network with e.g. GRU units with outputs of both directions concatenated and averaged over time. (3) using 1D convolution with outputs averaged over time. (4) computing differentiable version of VLAD (Vector of Locally Aggregated Descriptors) – NetVLAD [109]. (5) using graph network [110] to learn a fixed-sized hierarchical representation of the video among frames.



**Figure 2.3:** A diagram of the original version of W2VV++ model, taken from [4].

Wu et al. [104] utilize averaging over time (1). Further GRU (2), NetVLAD (4), graph network (5) are run in parallel on top of frame features. Video representation of Dong et al. [103] concatenates output of the approach (1), GRU (2) and 1D CNN (3) run on top of the GRU features. Li et al. [4] use only the feature averaging over time.

**Word features.** Over the years multiple approaches to word embeddings have been developed. All three mentioned works adopt a pre-trained 500-dimensional Word2Vec model trained on English tags associated with 30 million Flickr images supporting over 1.7 million words [90]. The 2019 version [106] of W2VV++ system by Li et al. also takes advantage of pre-trained BERT [96] sub-word embeddings.

**Sentence features.** One of the simplest representation is bag-of-words (BoW). The work of Li et al. constructs BoW by taking all words from a training set, excluding those that occur less than five times and those contained in the NLTK stopword list. Aside from the BoW features, Li et al. utilize Word2Vec pre-trained embeddings and single layer 1024-dimensional GRU network. The outputs of both methods are averaged over all words and concatenated together with BoW into  $(500 + 1024 + 11147)$ -dimensional vector. Works of Wu et al. and Dong et al. embed words using the Word2Vec pre-trained embeddings and use the same architecture as they use for video feature extraction. Further, they also allow to fine-tune the embeddings.

**Loss function.** Li et al. and Dong et al. project sentence features and video features into the common vector spaces by a single fully-connected layer followed by hyperbolic tangent or batch normalization respectively. Then all the approaches adopt max-margin ranking loss (Equation 2.1) with hard

negatives [111], i.e.  $\mathcal{N}(i) = \operatorname{argmax}_{j \neq i} S(V_i, C_j)$  or  $\operatorname{argmax}_{j \neq i} S(V_j, C_i)$  for the first and the second term respectively. The idea behind the hardest negative can be intuitively explained by the fact that the hardest negative determines success or failure in the Recall@1 metric. For practical reasons, the hardest negative is selected only from mini-batch instead of the whole dataset. The work of Li et al. further only uses the second term in the max-margin ranking loss with negatives for videos but not for sentences.

## 2.2 Our Method

As already mentioned, text-to-visual matching systems usually employ pre-trained visual and text feature extractors and learn only the mapping between these features. While some work focuses on training video feature extractors [8], to the best of our knowledge, no work trains advanced text feature extractors. The authors of [8] prove that training the whole video extractor benefits from the performance if large, even only automatically annotated, datasets are available. In this work, we experimentally prove the same for the text extractor. For that we adopt Transformer-based [112] state-of-the-art model RoBERTa [113], pre-trained on large corpora of English texts from the internet such as Wikipedia and news articles. Even though Transformers have already been used in the text-to-visual systems, e.g. Renmin University at TRECVID 2019 [106] and Contrastive Bidirectional Transformer [114] both utilized pre-trained BERT or MIECH et al. [8] adopted one Transformer’s attention layer into their language part of the system; we show that training the whole Transformer network for this task is beneficial.

### 2.2.1 Problem Statement

Given a video  $V = \{v_1, \dots, v_n\}$  with frames  $v_i$  and a text caption  $C = \{c_1, \dots, c_m\}$  with words  $c_i$ , we aim to learn a mapping of the video  $f(V; \theta_f)$  into  $\mathbb{R}^d$  and a mapping of the caption  $g(C; \theta_g)$  to the same space  $\mathbb{R}^d$  such that similarity  $S(V, C)$  is maximized for a video-caption pair, if the caption describes correctly the video, and minimized for a pair, if the caption does not describe the video. Note, we will omit the parameters  $\theta_f$  or  $\theta_g$  and refer to both the functions and their parameters by only  $f$  or  $g$  for simplicity. We define the similarity as cosine of  $f(V)$  and  $g(C)$  as

$$S(V, C) = \frac{f(V)^\top g(C)}{\|f(V)\|_2 \|g(C)\|_2}. \quad (2.3)$$

In this framework, text retrieval in a video database  $\mathcal{DB}$  is solved by computing  $f(V_i)$  for all videos  $V_i$ . Given a query  $Q$ ,  $g(Q)$  is computed and for each video similarity scores  $S(V_i, Q)$  are computed and the videos with the highest scores are returned. Since  $f(V_i)$  can be precomputed and  $g(Q)$  can be considered as a constant for a query of limited length, the time complexity is determined by the size of the database and the dimension  $d$  of the joint space  $\mathbb{R}^d$ , i.e.  $O(|\mathcal{DB}|d)$ . Note, in practice, a video can contain many unrelated segments while the query usually describes only one segment; therefore, videos are split into individual shots, i.e. in the context of this thesis  $V_i$  is not the whole video but only a single shot.

### 2.2.2 Video Representation

For video representation we use the W2VV++ (Word-To-Visual-Vector) system [4] developed by Li et al. As described in Section 2.1.2 a frame is extracted every 0.5 seconds, it is resized into  $256 \times 256$  and features are extracted from its 10 sub-images. The features are obtained by ResNext-101 [107] and Resnet-152 [108] both trained on ImageNet. The features are averaged across the image patches and all the video frames resulting in a fixed 4096-dimensional vector.

The only trainable part of the video mapping function  $f$  is a single fully-connected layer with hyperbolic tangent, i.e. the whole video mapping function is as follows:

$$f(V) = \tanh(W_f \mathcal{N}(V) + b_f) \quad (2.4)$$

where  $\mathcal{N}(\cdot)$  is the non-trainable mapping to 4096-dimensional space by CNNs averaged over frames described in the previous paragraph,  $W_f \in \mathbb{R}^{d \times 4096}$  and  $b_f \in \mathbb{R}^d$  are the trainable weights and biases.

### 2.2.3 Text Representation

Similarly to Li et al. three components are utilized – Bag of Words, Word2Vec, and newly BERT instead of RNN. Specifically, the Bag of Words component is defined as

$$BoW(C) = (\#(w_1, C), \dots, \#(w_{|\mathcal{V}|}, C)) \quad (2.5)$$

where  $w_i \in \mathcal{V}$  are words from a vocabulary and  $\#(w, C)$  is number of occurrences of a word  $w$  in the text  $C$ . We use the same vocabulary as Li et al. with  $|\mathcal{V}| = 11147$  words.

Word2Vec component utilizes a pre-trained 500-dimensional model trained on English tags associated with 30 million Flickr images supporting over 1.7 million words [90]. A text caption  $C$  is represented by the average of the individual word embeddings  $e(c_i)$ . The embedding is not fine-tuned and stays fixed throughout the whole training.

$$W2V(C) = \frac{1}{|C|} \sum_{i=1}^{|C|} e(c_i) \quad (2.6)$$

The last component of the text architecture is BERT neural network [96]. BERT’s architecture is a multi-layer bidirectional Transformer encoder based on the work by Vaswani et al. [112] (Figure 2.4). It uses only self-attention and point-wise, fully connected layers stacked into transformer blocks with information passing between words (sub-words) only via the self-attention. In our work, we use the 110 million parameter variant of BERT called BERT<sub>BASE</sub> with 12 Transformer blocks, size of the hidden layers 768, and 12 self-attention heads. For detailed architecture description, we point readers to the original work of Vaswani et al. [112] and the BERT paper [96].

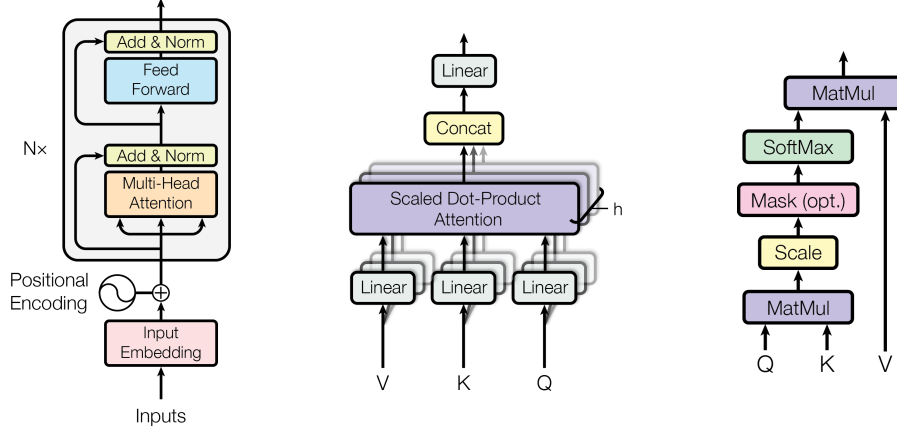
BERT outputs a 768-dimensional vector for each input sub-word. We average all sub-word vectors to obtain representation  $BERT(C)$  of the original text caption  $C$ . We initialize BERT’s weights by the ones provided by the authors of *RoBERTa: A Robustly Optimized BERT Pretraining Approach* [113] and further train them on our dataset.

Similarly to the video network, the text network  $g$  ends by concatenating the text representations and applying a fully-connected layer with hyperbolic tangent and projects a text caption into the joint text-video space:

$$g(C) = \tanh(W_g [BoW(C); W2V(C); BERT(C)] + b_g). \quad (2.7)$$

In the equation,  $[\dots]$  is vector concatenation,  $W_g \in \mathbb{R}^{d \times (11147+500+768)}$  and  $b_g \in \mathbb{R}^d$  together with all the weights of BERT are the trainable parameters of  $g$ .





**Figure 2.4:** Schema of Transformer-based encoder such as BERT (left), multi-head attention (middle), and single scaled dot-product attention (right). Images from [112].

### 2.2.4 Loss Function

Our work employs the same variant of max-margin ranking loss (Equation 2.1) as Li et al. Specifically, the term with caption negatives is ignored and the set of video negatives contains only the hardest negative from a batch  $\mathcal{B}$ :

$$\min \sum_{i=1}^{|\mathcal{B}|} \left[ \delta + \operatorname{argmax}_{j \neq i} S(V_j, C_i) - S(V_i, C_i) \right]_+. \quad (2.8)$$

We discuss the decision and compare other losses in the next section.

## 2.3 Experiments

We first describe datasets used in our experiments for training, validation, and testing together with metrics used for evaluation. Then in Section 2.3.2 training details are provided. Section 2.3.3 presents the results of our model and compares them to the related work, mainly W2VV++ [4]. Finally, an ablation study in Section 2.3.4 justifies key design choices such as the loss function.

### 2.3.1 Datasets and Evaluation Metrics

We use the same train, validation and test sets as in the work of Li et al. with the features provided by the authors<sup>1</sup>. For train set MSR-VTT [100] and Tumblr GIF (TGIF) [101] is used. TGIF dataset contains over 100 thousand animated GIFs with their natural language captions. MSR-VTT dataset contains 200 thousand short clips each paired with text descriptions in 10 thousand videos. For validation is used TRECVID 2016 Video-to-Text task dataset [115] with two thousand Vine videos each described by a sentence by two annotators. To stay consistent with Li et al. we utilize only 200 videos provided by the authors<sup>1</sup>. Testing is performed on the IACC.3 dataset [72] used for TRECVID AVS task from 2016 to 2018 containing approximately 4600 Internet Archive videos with 600 hours of content. The videos are split into 335,944 short clips by automatically detected shot boundaries provided by the authors of the dataset. Further for testing we use over two hundred sentence-based queries manually collected for 20 thousand frames uniformly sampled from the V3C1 dataset (dubbed 20k-V3C1).

Suppose we have a set  $\mathcal{Q} = \{(C_i, \mathcal{T}_i)\}_i$  of caption-video pairs with the possibility that a caption  $C_i$  can correspond to many target videos  $V \in \mathcal{T}_i$ . Given a query (text caption)  $C$  a model assigns each video  $V$  a position (rank) in the result list  $r(C, V)$ . Good model will assign low  $r(\cdot, \cdot) \rightarrow 1$  for relevant videos and large  $r(\cdot, \cdot) \rightarrow |\mathcal{DB}|$  for irrelevant ones. We use the following metrics to measure model performance. Recall@k (R@k) is defined as:

$$\text{Recall@k} = \frac{1}{|\mathcal{Q}|} \sum_{(C_i, \mathcal{T}_i) \in \mathcal{Q}} \left( \frac{1}{|\mathcal{T}_i|} \sum_{V \in \mathcal{T}_i} \mathbb{I}[r(C_i, V) \leq k] \right) \quad (2.9)$$

where  $\mathbb{I}[\cdot]$  is indicator function which is one if the argument is true and zero otherwise. As we use Recall@k in scenario where there is always only a single target video ( $\mathcal{T}_i = \{V_i\}$ ) the expression in the bracket can be simplified into  $\mathbb{I}[r(C_i, V_i) \leq k]$ . Further, the official TRECVID AVS metric is based on mean average precision (mAP). The standard mAP is defined as:

$$\text{mAP} = \frac{1}{|\mathcal{Q}|} \sum_{(C_i, \mathcal{T}_i) \in \mathcal{Q}} \left( \frac{1}{|\mathcal{T}_i|} \sum_{V \in \mathcal{T}_i} \frac{|\{\nu \in \mathcal{T}_i \mid r(C_i, \nu) \leq r(C_i, V)\}|}{r(C_i, V)} \right) \quad (2.10)$$

where the innermost fraction is Precision@k with  $k = r(C_i, V)$ . If the set of target videos  $\mathcal{T}_i$  contains only one video  $V_i$  the metric is also called mean reciprocal rank (MRR)  $1/|\mathcal{Q}| \cdot \sum_{(C_i, \{V_i\}) \in \mathcal{Q}} 1/r(C_i, V_i)$ .

Since it is difficult to evaluate all 335,944 shots from IACC.3 manually, performance at TRECVID AVS task is measured by inferred average precision (infAP)

<sup>1</sup>Publicly available at <https://github.com/li-xirong/avs>.

[116]. Compared to standard (mean) average precision it eliminates bias created by only manually labeling the retrieved shots while unretrieved shots are not labeled.

### 2.3.2 Training Details

In all our experiments, unless otherwise stated, the joint space dimension  $d$  is set to 2048. Prior training, weights  $W_f$  and  $W_g$  are initialized by Glorot initializer [73], biases  $b_f$  and  $b_g$  are initialized by zeros and BERT network by pre-trained weights from [113]. Batch size of 96 clip-text pairs is selected to fit into 16GB of VRAM of a Tesla V100 GPU. We use the Adam optimizer [74] with a learning rate of  $10^{-5}$  and linear warm-up for the half of the first epoch (approx. 1700 steps). The learning rate is linearly decayed to zero in a span of 60 epochs; however, the best performing model on the validation set is selected which is usually achieved after approximately ten epochs. With the video features extracted beforehand, the training itself takes a few hours on a single Tesla V100 GPU. Pytorch and its fairseq<sup>2</sup> library with the implementation of BERT has been used for all the experiments.

### 2.3.3 Results

We report the results of three variants: A) the whole model as described in Section 2.2 is used. B) the Word2Vec portion of the text encoding network is removed. C) both Word2Vec and Bag-of-Words parts of the text network are removed, i.e. only BERT is used for text projection into the joint space. Similarly, we report results of the original W2VV++ [4] as well as W2VV++ with only Bag-of-Words in the text network, i.e. without Word2Vec and RNN.

Table 2.1 shows the performance of various models on TRECVID AVS tasks for years 2016 through 2018. We see slightly lower performance of our BERT based systems on 2016 and 2018 tasks while we see significantly better results for 2017 tasks. A single model combining BERT, Word2Vec, and Bag-of-Words even outperforms the second-best performer of TRECVID AVS 2019 challenge both in single model setting (infAP 22.8) and also in an ensemble setting where multiple Dual Encoding models were combined to produce the result (infAP 23.9 [106]). Surprisingly the original W2VV++ system is outperformed by its BoW-only variant in all three years. We hypothesize it is due to the fact that (inferred) mean average precision heavily favors results with the target item in the first position. For example, a model returning a target item always in the top 10 results can achieve worse mAP than a model returning a target item a few times at the first position but other times not returning it at all.

The described phenomenon can be seen in results on the 20k-V3C1 dataset show in Table 2.2. BoW-only variant of W2VV++ retrieves the searched frame at the first position in almost twelve percent of 202 sentence based queries. However, when we look at the percentage of queries such that the target frame is retrieved in the top ten positions, BoW-only W2VV++ already performs the worst of all the models. Further, the difference can be seen for the top 100 positions where our model outperforms BoW-only W2VV++ by almost 11 percentage points.

---

<sup>2</sup>Available at <https://github.com/pytorch/fairseq>

Model	TV16	TV17	TV18	Average
W2VV++ [4]	15.5	21.5	10.7	15.9
W2VV++ (BoW only)*	<b>15.6</b>	21.8	<b>11.0</b>	16.1
<b>Ours</b> (BERT)	13.7	22.5	9.8	15.3
<b>Ours</b> (BERT + BoW)	15.0	23.4	10.3	16.2
<b>Ours</b> (BERT + BoW + Word2Vec)	14.6	<b>24.1</b>	10.2	<b>16.3</b>
Dual Encoding [103] <sup>†</sup>	<b>16.5</b>	22.8	<b>11.7</b>	<b>17.0</b>

\* W2VV++ utilizes BoW, Word2Vec and RNN, this version uses only BoW.

<sup>†</sup> Results taken from [106].

**Table 2.1:** Model comparison on TRECVID AVS tasks for years 2016 to 2018. Values are in percents of inferred mean average precision averaged over three runs. Results for W2VV++ are courtesy of the authors.

Model	R@1	R@10	R@100	MRR
W2VV++ [4]	7.9	28.7	60.4	14.6
W2VV++ (BoW only)	<b>11.9</b>	27.2	57.4	<b>16.5</b>
<b>Ours</b> (BERT)	8.9	28.2	65.3	14.8
<b>Ours</b> (BERT + BoW)	9.9	27.2	65.8	16.1
<b>Ours</b> (BERT + BoW + Word2Vec)	9.4	<b>30.2</b>	<b>68.3</b>	16.4

**Table 2.2:** Model comparison on 20k-V3C1 dataset (20k frames, 202 frame-caption pairs). Recall@k and mean reciprocal rank shown in percents. Note R@100 represents recall given 0.5 percent of the original dataset.

Also visual comparison of the full W2VV++ model and our BERT extension can be seen in Figures 2.5, 2.6 and 2.7.

In Table 2.3 results on TRECVID 2016 Video-to-text dataset are shown. Note the dataset was used for validation in our experiments as well as experiments of Li et al. Every video in the dataset is captioned by two annotators – hence the set A and set B. For validation purposes only the set A was used. The results show that all our models outperform both W2VV++ variants in all metrics. Surprisingly the BERT-only model outperforms the other combinations.

Model	Set A			Set B		
	R@1	R@10	MRR	R@1	R@10	MRR
W2VV++ [4]	42.5	79.0	55.7	43.0	82.5	56.9
W2VV++ (BoW only)	39.5	79.0	52.3	41.0	82.0	54.3
<b>Ours</b> (BERT)	<b>45.0</b>	82.0	<b>58.4</b>	<b>47.5</b>	<b>86.5</b>	<b>59.0</b>
<b>Ours</b> (BERT + BoW)	43.0	<b>82.5</b>	56.3	45.5	85.0	57.8
<b>Ours</b> (BERT + BoW + W2V)	43.5	80.5	57.1	45.0	82.5	57.0

**Table 2.3:** Model comparison on TRECVID 2016 Video-to-text dataset.

Loss	TV16	TV17	TV18	Average
Max-margin, text-to-visual (Eq. 2.8)	14.6	<b>24.1</b>	<b>10.2</b>	<b>16.3</b>
Max-margin, bidirectional	<b>14.7</b>	23.0	10.0	15.9

**Table 2.4:** Variants of max-margin loss and their performance on TRECVID AVS tasks. Results are in percents of inferred mean average precision averaged over three runs. Our full model (BERT + BoW + W2V) was used.

Loss	Batch size	TV16	TV17	TV18
NCE	128	12.1	15.4	9.0
	512	13.0	19.6	8.7
	2048	14.0	<b>20.3</b>	10.1
Max-margin	128	<b>14.8</b>	20.0	<b>10.4</b>
	512	12.9	16.9	8.5
	2048	10.6	12.1	7.0

**Table 2.5:** Comparison of NCE loss and max-margin loss on TRECVID AVS tasks. Our BERT + BoW model without BERT fine-tuning was used.

### 2.3.4 Ablation Study

We perform multiple ablation studies. In Table 2.4 we compare our text-to-visual max-margin ranking loss (Equation 2.8) to bidirectional (text-to-visual + visual-to-text) variant that minimizes not only similarity of hardest negative video to a caption but also similarity of hardest negative caption to a video (adaptation of Equation 2.1). We see slightly better results for the text-to-visual version of the loss on TRECVID AVS tasks with our full model (BERT + BoW + W2V).

Further we compare max-margin ranging loss to noise-contrastive estimation (NCE) loss used by Miech et al. [8] (Table 2.5). For comparison, we use our BERT + BoW model without BERT fine-tuning due to large batch sizes that do not fit into VRAM of a single GPU. Therefore the results are worse than those reported in Table 2.1. For noise-contrastive estimation loss, we see a poor performance with small batch size while the performance approaches max-margin loss for large batch sizes which is consistent with observations of Miech et al. [8]. On the contrary, increasing the batch size for max-margin loss harms the performance – we argue it is due to the fact that as batch size increases the more likely the batch contains similar scenes and therefore the hardest negative is sometimes actually a viable positive sample.

Retrieval using our joint space model is highly demanding for computational resources due to the fact that it requires the computation of cosine similarity for every clip in a database. Reducing the dimension  $d$  of the joint space from 2048 to 128 reduces computation time sixteen-fold. However, as shown in Table 2.6, it severely hampers the performance – it seems the network does not have enough freedom in the parameter space to learn good mapping. Nonetheless, we were able to reduce the dimension to 128 without any loss in performance by principal component analysis (PCA). The projection matrix is computed using only the video database feature vectors which makes the process suitable for any video

Joint Space Dimension	R@1	R@10	R@100	MRR
128	3.5	23.3	55.9	9.8
512	8.4	26.2	64.4	14.0
2048 (standard)	9.4	<b>30.2</b>	<b>68.3</b>	16.4
2048 + PCA into 128 dimensions	<b>9.9</b>	28.7	67.8	<b>17.4</b>

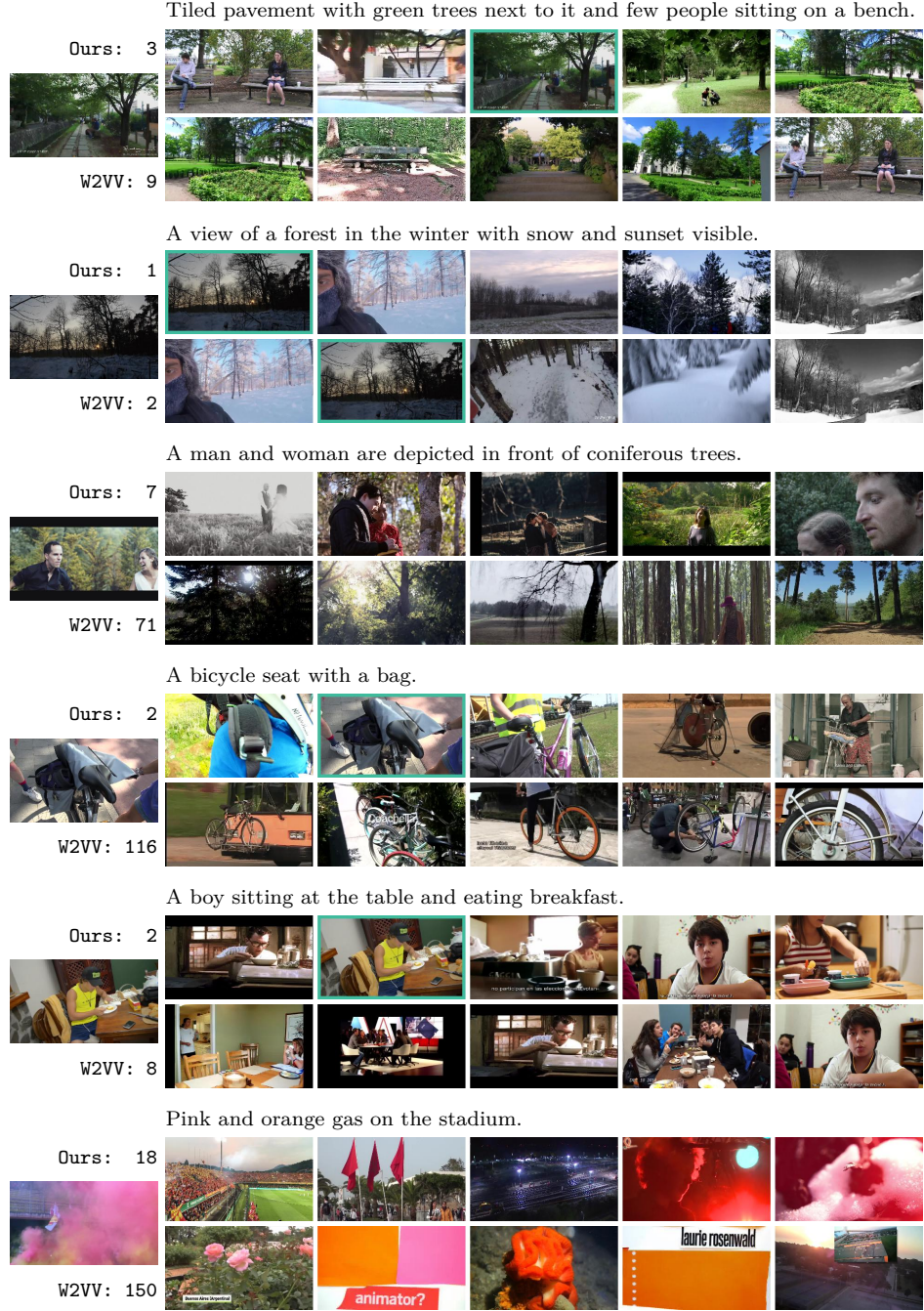
**Table 2.6:** Performance dependency on the joint space dimension on our in-house 20k-V3C1 dataset. Our full model (BERT + BoW + W2V) was used.

Model	# patches	R@1	R@10	R@100
W2VV++ (BoW only)	1	7.9	26.2	52.0
	10	<b>11.9</b>	<b>27.2</b>	<b>57.4</b>
<b>Ours</b> (BERT + BoW + W2V)	1	<b>9.9</b>	28.2	67.3
	10	9.4	<b>30.2</b>	<b>68.3</b>

**Table 2.7:** Performance dependency on number of patches per frame. Measured on 20k-V3C1 dataset.

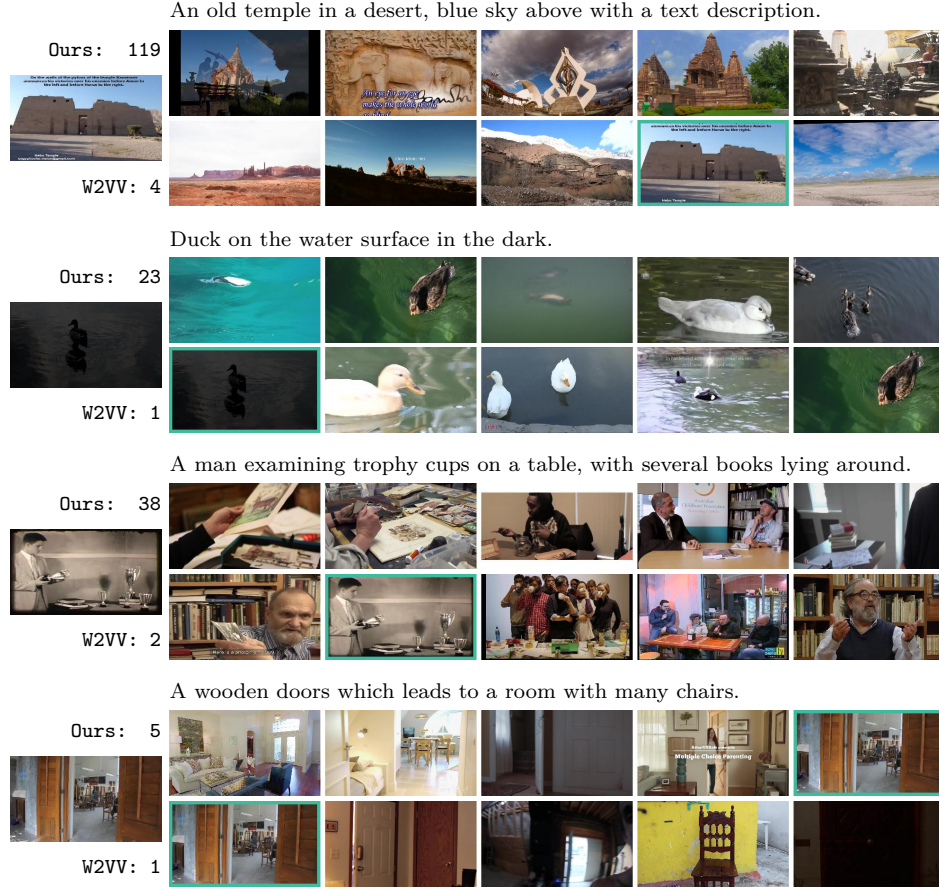
database since the text queries are not needed for the matrix computation. However, unlike the standard PCA, the feature vectors  $f(V)$  are not centered prior to the projection matrix computation since the original joint space is optimized for cosine similarity, not euclidean distance, and the centering does not preserve angles. With the centering, the results are comparable to directly training the network with the joint space of dimension 128.

The original W2VV++ system computes visual features for each frame by averaging features of its 10 patches – one center patch and four corner patches, all of them also horizontally flipped. This approach however increases extraction time ten-fold. In Table 2.7 we compare it with an approach that computes the features only once for the whole frame. We see a significant drop in performance for the BoW-only variant of W2VV++; however, our BERT + BoW + Word2Vec model’s performance does not change much.

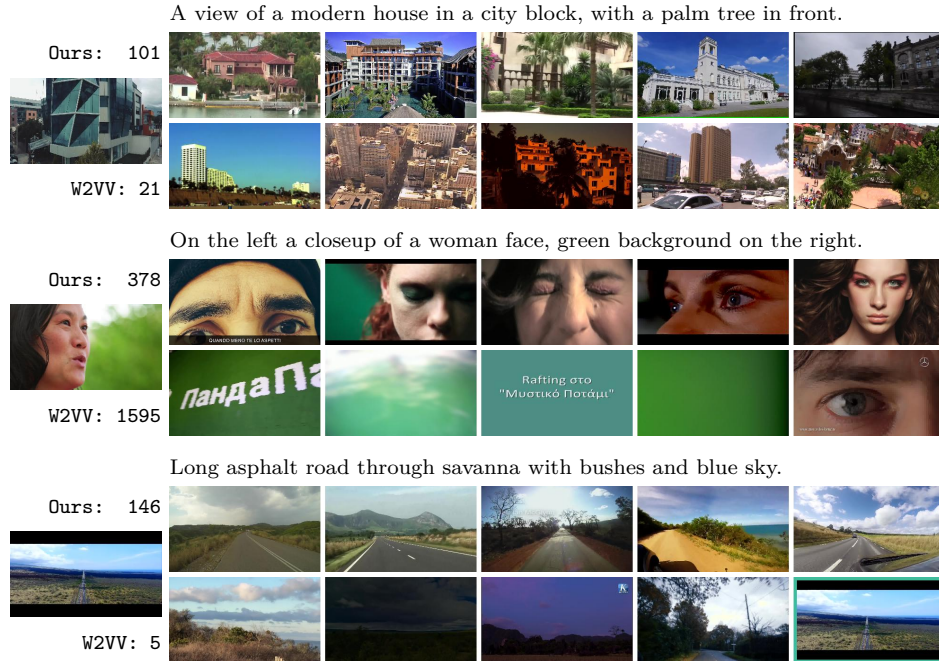


**Figure 2.5:** Comparison between W2VV++ and our BERT extension on 20k-V3C1 dataset, queries in favor of BERT. Show top 5 results of our model (top) and W2VV++ (bottom) for a given query. The target image is shown on the left with its position according the two models. See Figures 2.6 and 2.7 for more examples.





**Figure 2.6:** Continuation of Figure 2.5, queries in favor of W2VV++.



**Figure 2.7:** Continuation of Figure 2.5, unclear and wrong queries. Even though W2VV++ outperforms our model on the first and third query, its top results do not match the query well.



# Conclusion

This thesis approaches two key problems of video retrieval – shot boundary detection and text-based search. We present TransNet V2, a network for the detection of common shot transitions, which tackles an important initial step of video analysis processes. We address multiple issues of previous shot detectors and discuss in detail the training process as well as training data acquisition. We also reevaluate and compare ourselves to other recent approaches to shot boundary detection. The results show our method can outperform related works on multiple public benchmarks, and we believe it can be of great help in many video pre-processing pipelines of various multimedia search/analytics frameworks that require information about shots.

Further, we investigate text-to-visual matching systems utilized for text-based search in video collections. We experimentally prove that adding large Transformer-based text encoders improves the performance of such systems on some tasks if the whole encoder is trained. All in all, we believe that both the shot boundary detector as well as our extension to the text-based search system show promising future research directions in both areas.

# Bibliography

- [1] Jingqing Zhang, Yao Zhao, Mohammad Saleh, and Peter J Liu. Pegasus: Pre-training with extracted gap-sentences for abstractive summarization. *arXiv preprint arXiv:1912.08777*, 2019.
- [2] Antoine Miech, Dimitri Zhukov, Jean-Baptiste Alayrac, Makarand Tapaswi, Ivan Laptev, and Josef Sivic. Howto100m: Learning a text-video embedding by watching hundred million narrated video clips. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2019.
- [3] Kaiyang Zhou, Yu Qiao, and Tao Xiang. Deep reinforcement learning for unsupervised video summarization with diversity-representativeness reward. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.
- [4] Xirong Li, Chaoxi Xu, Gang Yang, Zhineng Chen, and Jianfeng Dong. W2vv++: Fully deep learning for ad-hoc video search. In *Proceedings of the 27th ACM International Conference on Multimedia*, MM ’19, page 1786–1794, New York, NY, USA, 2019. Association for Computing Machinery.
- [5] Zhichao Yin, Trevor Darrell, and Fisher Yu. Hierarchical discrete distribution decomposition for match density estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.
- [6] Charles R. Qi, Wei Liu, Chenxia Wu, Hao Su, and Leonidas J. Guibas. Frustum pointnets for 3d object detection from rgb-d data. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.
- [7] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 26*, pages 3111–3119. Curran Associates, Inc., 2013.
- [8] Antoine Miech, Jean-Baptiste Alayrac, Lucas Smaira, Ivan Laptev, Josef Sivic, and Andrew Zisserman. End-to-end learning of visual representations from uncurated instructional videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.
- [9] Jamie Ray, Heng Wang, Du Tran, Yufei Wang, Matt Feiszli, Lorenzo Torresani, and Manohar Paluri. Scenes-objects-actions: A multi-task, multi-label video dataset. In *Proceedings of the European Conference on Computer Vision (ECCV)*, September 2018.
- [10] L. Rossetto, R. Gasser, J. Lokoc, W. Bailer, K. Schoeffmann, B. Muenzer, T. Soucek, P. A. Nguyen, P. Bolettieri, A. Leibetseder, and S. Vrochidis.

Interactive video retrieval in the age of deep learning - detailed evaluation of vbs 2019. *IEEE Transactions on Multimedia*, pages 1–1, 2020.

- [11] Aaron Duane, Cathal Gurrin, and Wolfgang Huerst. Virtual reality lifelog explorer: Lifelog search challenge at acm icmr 2018. In *Proceedings of the 2018 ACM Workshop on The Lifelog Search Challenge*, LSC '18, page 20–23, New York, NY, USA, 2018. Association for Computing Machinery.
- [12] Kai Uwe Barthel, Nico Hezel, Konstantin Schall, and Klaus Jung. Real-time visual navigation in huge image sets using similarity graphs. In *Proceedings of the 27th ACM International Conference on Multimedia*, MM '19, page 2202–2204, New York, NY, USA, 2019. Association for Computing Machinery.
- [13] Björn Þór Jónsson, Omar Shahbaz Khan, Dennis C. Koelma, Stevan Rudinac, Marcel Worring, and Jan Zahálka. Exquisitor at the video browser showdown 2020. In Yong Man Ro, Wen-Huang Cheng, Junmo Kim, Wei-Ta Chu, Peng Cui, Jung-Woo Choi, Min-Chun Hu, and Wesley De Neve, editors, *MultiMedia Modeling*, pages 796–802, Cham, 2020. Springer International Publishing.
- [14] Miroslav Kratochvíl, Patrik Veselý, František Mejzlík, and Jakub Lokoč. Som-hunter: Video browsing with relevance-to-som feedback loop. In Yong Man Ro, Wen-Huang Cheng, Junmo Kim, Wei-Ta Chu, Peng Cui, Jung-Woo Choi, Min-Chun Hu, and Wesley De Neve, editors, *MultiMedia Modeling*, pages 790–795, Cham, 2020. Springer International Publishing.
- [15] Cathal Gurrin, Tu-Khiem Le, Van-Tu Ninh, Duc-Tien Dang-Nguyen, Björn Þór Jónsson, Jakub Lokoč, Wolfgang Hürst, Minh-Triet Tran, and Klaus Schöffmann. Introduction to the third annual lifelog search challenge (lsc'20). In *Proceedings of the 2020 International Conference on Multimedia Retrieval*, ICMR '20, page 584–585, New York, NY, USA, 2020. Association for Computing Machinery.
- [16] Adam Blažek, Jakub Lokoč, Filip Matzner, and Tomáš Skopal. Enhanced signature-based video browser. In Xiangjian He, Suhuai Luo, Dacheng Tao, Changsheng Xu, Jie Yang, and Muhammad Abul Hasan, editors, *MultiMedia Modeling*, pages 243–248, Cham, 2015. Springer International Publishing.
- [17] Jakub Lokoč, Gregor Kovalčík, and Tomáš Souček. Viret at video browser showdown 2020. In Yong Man Ro, Wen-Huang Cheng, Junmo Kim, Wei-Ta Chu, Peng Cui, Jung-Woo Choi, Min-Chun Hu, and Wesley De Neve, editors, *MultiMedia Modeling*, pages 784–789, Cham, 2020. Springer International Publishing.
- [18] Loris Sauter, Mahnaz Amiri Parian, Ralph Gasser, Silvan Heller, Luca Rossetto, and Heiko Schuldt. Combining boolean and multimedia retrieval in vitivr for large-scale video search. In Yong Man Ro, Wen-Huang Cheng, Junmo Kim, Wei-Ta Chu, Peng Cui, Jung-Woo Choi, Min-Chun Hu, and Wesley De Neve, editors, *MultiMedia Modeling*, pages 760–765, Cham, 2020. Springer International Publishing.

- [19] Jakub Lokoč, Gregor Kovalčík, Tomáš Souček, Jaroslav Moravec, Jan Bodnár, and Přemysl Čech. Viret tool meets nasnet. In Ioannis Kompatsiaris, Benoit Huet, Vasileios Mezaris, Cathal Gurrin, Wen-Huang Cheng, and Stefanos Vrochidis, editors, *MultiMedia Modeling*, pages 597–601, Cham, 2019. Springer International Publishing.
- [20] Klaus Schoeffmann, Bernd Münzer, Andreas Leibetseder, Jürgen Primus, and Sabrina Kletz. Autopiloting feature maps: The deep interactive video exploration (divexplore) system at vbs2019. In Ioannis Kompatsiaris, Benoit Huet, Vasileios Mezaris, Cathal Gurrin, Wen-Huang Cheng, and Stefanos Vrochidis, editors, *MultiMedia Modeling*, pages 585–590, Cham, 2019. Springer International Publishing.
- [21] Fabian Berns, Luca Rossetto, Klaus Schoeffmann, Christian Beecks, and George Awad. V3c1 dataset: An evaluation of content characteristics. In *Proceedings of the 2019 on International Conference on Multimedia Retrieval*, ICMR ’19, page 334–338, New York, NY, USA, 2019. Association for Computing Machinery.
- [22] Jakub Lokoč, Gregor Kovalčík, Tomáš Souček, Jaroslav Moravec, and Přemysl Čech. A framework for effective known-item search in video. In *Proceedings of the 27th ACM International Conference on Multimedia*, MM ’19, page 1777–1785, New York, NY, USA, 2019. Association for Computing Machinery.
- [23] Tomáš Souček, Jaroslav Moravec, and Jakub Lokoč. Transnet: A deep network for fast detection of common shot transitions. *CoRR*, abs/1906.03363, 2019.
- [24] Jakub Lokoč, Tomáš Souček, Patrik Veselý, František Mejzlík, Jiaqi Ji, Chaoxi Xu, and Xirong Li. A W2VV++ case study with automated and interactive text-to-video. Accepted to the 28th ACM International Conference on Multimedia, 2020.
- [25] Tomáš Souček and Jakub Lokoč. TransNet V2: An effective deep network architecture for fast shot transition detection. 2020.
- [26] Jakub Lokoč, Gregor Kovalčík, Tomáš Souček, Jaroslav Moravec, and Přemysl Čech. Viret: A video retrieval tool for interactive known-item search. In *Proceedings of the 2019 on International Conference on Multimedia Retrieval*, ICMR ’19, page 177–181, New York, NY, USA, 2019. Association for Computing Machinery.
- [27] Jakub Lokoč, Tomáš Souček, Přemysl Čech, and Gregor Kovalčík. Enhanced viret tool for lifelog data. In *Proceedings of the ACM Workshop on Lifelog Search Challenge*, LSC ’19, page 25–26, New York, NY, USA, 2019. Association for Computing Machinery.
- [28] Jakub Lokoč, Gregor Kovalčík, and Tomáš Souček. Revisiting siret video retrieval tool. In Klaus Schoeffmann, Thanarat H. Chalidabhongse, Chong Wah Ngo, Supavadee Aramvith, Noel E. O’Connor, Yo-Sung Ho,

- Moncef Gabbouj, and Ahmed Elgammal, editors, *MultiMedia Modeling*, pages 419–424, Cham, 2018. Springer International Publishing.
- [29] Jakub Lokoč, Tomáš Souček, and Gregor Kovalčík. Using an interactive video retrieval tool for lifelog data. In *Proceedings of the 2018 ACM Workshop on The Lifelog Search Challenge*, LSC '18, page 15–19, New York, NY, USA, 2018. Association for Computing Machinery.
  - [30] J. Yuan, H. Wang, L. Xiao, W. Zheng, J. Li, F. Lin, and B. Zhang. A formal study of shot boundary detection. *IEEE Transactions on Circuits and Systems for Video Technology*, 17(2):168–186, Feb 2007.
  - [31] Sadiq Abdulhussain, Abd Ramli, M. Saripan, Basheera Mahmmud, Syed Al-Haddad, and Wissam Jassim. Methods and challenges in shot boundary detection: A review. *Entropy*, 20(4):214, Mar 2018.
  - [32] C. Liu, D. Wang, J. Zhu, and B. Zhang. Learning a contextual multi-thread model for movie/tv scene segmentation. *IEEE Transactions on Multimedia*, 15(4):884–897, June 2013.
  - [33] Bin Zhao, Xuelong Li, and Xiaoqiang Lu. Hsa-rnn: Hierarchical structure-adaptive rnn for video summarization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.
  - [34] Jakub Lokoč, Gregor Kovalčík, Bernd Münzer, Klaus Schöffmann, Werner Bailer, Ralph Gasser, Stefanos Vrochidis, Phuong Anh Nguyen, Sitapa Rujikietgumjorn, and Kai Uwe Barthel. Interactive search or sequential browsing? a detailed analysis of the video browser showdown 2018. *ACM Trans. Multimedia Comput. Commun. Appl.*, 15(1):29:1–29:18, February 2019.
  - [35] Onur Küçüktunç, Uğur Güdükbay, and Özgür Ulusoy. Fuzzy color histogram-based video segmentation. *Computer Vision and Image Understanding*, 114(1):125 – 134, 2010.
  - [36] HongJiang Zhang, Atreyi Kankanhalli, and Stephen W. Smoliar. Automatic partitioning of full-motion video. *Multimedia Systems*, 1(1):10–28, Jan 1993.
  - [37] X. Ling, L. Chao, L. Huan, and X. Zhang. A general method for shot boundary detection. In *2008 International Conference on Multimedia and Ubiquitous Engineering (mue 2008)*, pages 394–397, April 2008.
  - [38] L. Rossetto, I. Giangreco, and H. Schuldt. Cineast: A multi-feature sketch-based video retrieval engine. In *2014 IEEE International Symposium on Multimedia*, pages 18–23, Dec 2014.
  - [39] Ramin Zabih, Justin Miller, and Kevin Mai. A feature-based algorithm for detecting and classifying scene breaks. In *Proceedings of the third ACM International Conference on Multimedia*, pages 189–200, 1995.
  - [40] Rainer W. Lienhart. Comparison of automatic shot boundary detection algorithms. In Minerva M. Yeung, Boon-Lock Yeo, and Charles A. Bouman,

- editors, *Storage and Retrieval for Image and Video Databases VII*, volume 3656, pages 290 – 301. International Society for Optics and Photonics, SPIE, 1998.
- [41] Hong Shao, Yang Qu, and Wencheng Cui. Shot boundary detection algorithm based on hsv histogram and hog feature. *5th International Conference on Advanced Engineering Materials and Technology*, 2015.
  - [42] Evlampios E. Apostolidis and Vasileios Mezaris. Fast shot segmentation combining global and local visual descriptors. *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6583–6587, 2014.
  - [43] Herbert Bay, Tinne Tuytelaars, and Luc Van Gool. Surf: Speeded up robust features. In Aleš Leonardis, Horst Bischof, and Axel Pinz, editors, *Computer Vision – ECCV 2006*, pages 404–417, Berlin, Heidelberg, 2006. Springer Berlin Heidelberg.
  - [44] Manisha Verma and Balasubramanian Raman. A hierarchical shot boundary detection algorithm using global and local features. In Balasubramanian Raman, Sanjeev Kumar, Partha Pratim Roy, and Debashis Sen, editors, *Proceedings of International Conference on Computer Vision and Image Processing*, pages 389–397, Singapore, 2017. Springer Singapore.
  - [45] Lorenzo Baraldi, Costantino Grana, and Rita Cucchiara. A deep siamese network for scene detection in broadcast videos. In *Proceedings of the 23rd ACM International Conference on Multimedia*, MM ’15, page 1199–1202, New York, NY, USA, 2015. Association for Computing Machinery.
  - [46] Vasileios Chasanis, Aristidis Likas, and Nikolaos Galatsanos. Simultaneous detection of abrupt cuts and dissolves in videos using support vector machines. *Pattern Recognition Letters*, 30(1):55 – 65, 2009.
  - [47] E. Tsamoura, V. Mezaris, and I. Kompatsiaris. Gradual transition detection using color coherence and other criteria in a video shot meta-segmentation framework. In *2008 15th IEEE International Conference on Image Processing*, pages 45–48, Oct 2008.
  - [48] Greg Pass, Ramin Zabih, and Justin Miller. Comparing images using color coherence vectors. In *Proceedings of the fourth ACM international conference on Multimedia*, pages 65–73, 1997.
  - [49] Yoshihiko Kawai, Hideki Sumiyoshi, and Nobuyuki Yagi. Shot boundary detection at trecvid 2007. In *TRECVID*, 2007.
  - [50] Markus Mühling, Ralph Ewerth, Thilo Stadelmann, Christian Zöfel, Bing Shi, and Bernd Freisleben. University of marburg at trecvid 2007: Shot boundary detection and high level feature extraction. In *TRECVID*, 2007.
  - [51] Stéphane Ayache, Jérôme Gensel, Georges Quénot, and Stéphane Ayache. Clips-lsr experiments at trecvid 2006. In *TRECVID*, 2006.

- [52] Boon-Lock Yeo and Bede Liu. Rapid scene analysis on compressed video. *IEEE Transactions on Circuits and Systems for Video Technology*, 5(6):533–544, Dec 1995.
- [53] Alan F. Smeaton, Paul Over, and Aiden R. Doherty. Video shot boundary detection: Seven years of trecvid activity. *Computer Vision and Image Understanding*, 114(4):411 – 418, 2010. Special issue on Image and Video Retrieval Evaluation.
- [54] Ahmed Hassanien, Mohamed A. Elgharib, Ahmed Selim, Mohamed Hefeeda, and Wojciech Matusik. Large-scale, fast and accurate shot boundary detection through spatio-temporal convolutional neural networks. *CoRR*, abs/1705.03281, 2017.
- [55] Lorenzo Baraldi, Costantino Grana, and Rita Cucchiara. Shot and scene detection via hierarchical clustering for re-using broadcast video. In George Azzopardi and Nicolai Petkov, editors, *Computer Analysis of Images and Patterns*, pages 801–811, Cham, 2015. Springer International Publishing.
- [56] 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.
- [57] Jeff Donahue, Yangqing Jia, Oriol Vinyals, Judy Hoffman, Ning Zhang, Eric Tzeng, and Trevor Darrell. Decaf: A deep convolutional activation feature for generic visual recognition. In *International conference on machine learning*, pages 647–655, 2014.
- [58] J. Xu, L. Song, and R. Xie. Shot boundary detection using convolutional neural networks. In *2016 Visual Communications and Image Processing (VCIP)*, pages 1–4, Nov 2016.
- [59] Andrej Karpathy, George Toderici, Sanketh Shetty, Thomas Leung, Rahul Sukthankar, and Li Fei-Fei. Large-scale video classification with convolutional neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2014.
- [60] Karen Simonyan and Andrew Zisserman. Two-stream convolutional networks for action recognition in videos. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 568–576. Curran Associates, Inc., 2014.
- [61] Du Tran, Lubomir Bourdev, Rob Fergus, Lorenzo Torresani, and Manohar Paluri. Learning spatiotemporal features with 3d convolutional networks. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, December 2015.

- [62] Joao Carreira and Andrew Zisserman. Quo vadis, action recognition? a new model and the kinetics dataset. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.
- [63] 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 (CVPR)*, June 2015.
- [64] Saining Xie, Chen Sun, Jonathan Huang, Zhuowen Tu, and Kevin Murphy. Rethinking spatiotemporal feature learning: Speed-accuracy trade-offs in video classification. In *Proceedings of the European Conference on Computer Vision (ECCV)*, September 2018.
- [65] Du Tran, Heng Wang, Lorenzo Torresani, Jamie Ray, Yann LeCun, and Manohar Paluri. A closer look at spatiotemporal convolutions for action recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.
- [66] Michael Gygli. Ridiculously fast shot boundary detection with fully convolutional neural networks. In *2018 International Conference on Content-Based Multimedia Indexing, CBMI 2018, La Rochelle, France, September 4-6, 2018*, pages 1–4, 2018.
- [67] Shitao Tang, Litong Feng, Zhanghui Kuang, Yimin Chen, and Wei Zhang. Fast video shot transition localization with deep structured models. *CoRR*, abs/1808.04234, 2018.
- [68] Forrest N. Iandola, Matthew W. Moskewicz, Khalid Ashraf, Song Han, William J. Dally, and Kurt Keutzer. Squeezenet: Alexnet-level accuracy with 50x fewer parameters and <1mb model size. *CoRR*, abs/1602.07360, 2016.
- [69] Kensho Hara, Hirokatsu Kataoka, and Yutaka Satoh. Can spatiotemporal 3d cnns retrace the history of 2d cnns and imagenet? In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.
- [70] Fisher Yu and Vladlen Koltun. Multi-scale context aggregation by dilated convolutions. *arXiv preprint arXiv:1511.07122*, 2015.
- [71] Aäron van den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew W. Senior, and Koray Kavukcuoglu. Wavenet: A generative model for raw audio. *CoRR*, abs/1609.03499, 2016.
- [72] George Awad, Asad Butt, Jonathan Fiscus, Martial Michel, David Joy, Wessel Kraaij, Alan F. Smeaton, Georges Quénot, Maria Eskevich, Roeland Ordeman, 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.



- [73] Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In Yee Whye Teh and Mike Titterton, editors, *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, volume 9 of *Proceedings of Machine Learning Research*, pages 249–256, Chia Laguna Resort, Sardinia, Italy, 13–15 May 2010. PMLR.
- [74] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980, 2014.
- [75] Chia-Kai Liang and Fuhao Shi. Fused video stabilization on the pixel 2 and pixel 2 xl. <https://ai.googleblog.com/2017/11/fused-video-stabilization-on-pixel-2.html>, 2017. [Online; accessed 19-May-2020].
- [76] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(56):1929–1958, 2014.
- [77] Terrance Devries and Graham W. Taylor. Improved regularization of convolutional neural networks with cutout. *CoRR*, abs/1708.04552, 2017.
- [78] Ekin Dogus Cubuk, Barret Zoph, Dandelion Mané, Vijay Vasudevan, and Quoc V. Le. Autoaugment: Learning augmentation policies from data. *CoRR*, abs/1805.09501, 2018.
- [79] Francois Chollet. Xception: Deep learning with depthwise separable convolutions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.
- [80] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.
- [81] J. Deng, W. Dong, R. Socher, L. Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, June 2009.
- [82] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *CoRR*, abs/1502.03167, 2015.
- [83] Tijmen Tieleman and Geoffrey Hinton. Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude. *COURSERA: Neural networks for machine learning*, 4(2):26–31, 2012.
- [84] Mingxing Tan and Quoc V. Le. Efficientnet: Rethinking model scaling for convolutional neural networks. *CoRR*, abs/1905.11946, 2019.
- [85] Ilya Loshchilov and Frank Hutter. Fixing weight decay regularization in adam. *CoRR*, abs/1711.05101, 2017.

- [86] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, December 2015.
- [87] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.
- [88] Duy-Dinh Le, Sang Phan, Vinh-Tiep Nguyen, Benjamin Renoust, Tuan A Nguyen, Van-Nam Hoang, Thanh Duc Ngo, Minh-Triet Tran, Yuki Watanabe, Martin Klinkigt, et al. Nii-hitachi-uit at trecvid 2016. In *TRECVID*, 2016.
- [89] L. Wang, Y. Li, J. Huang, and S. Lazebnik. Learning two-branch neural networks for image-text matching tasks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 41(2):394–407, Feb 2019.
- [90] J. Dong, X. Li, and C. G. M. Snoek. Predicting visual features from text for image and video caption retrieval. *IEEE Transactions on Multimedia*, 20(12):3377–3388, Dec 2018.
- [91] Peter Young, Alice Lai, Micah Hodosh, and Julia Hockenmaier. From image descriptions to visual denotations: New similarity metrics for semantic inference over event descriptions. *Transactions of the Association for Computational Linguistics*, 2:67–78, 2014.
- [92] Kuang-Huei Lee, Xi Chen, Gang Hua, Houdong Hu, and Xiaodong He. Stacked cross attention for image-text matching. In *Proceedings of the European Conference on Computer Vision (ECCV)*, September 2018.
- [93] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 91–99. Curran Associates, Inc., 2015.
- [94] Ranjay Krishna, Yuke Zhu, Oliver Groth, Justin Johnson, Kenji Hata, Joshua Kravitz, Stephanie Chen, Yannis Kalantidis, Li-Jia Li, David A Shamma, et al. Visual genome: Connecting language and vision using crowdsourced dense image annotations. *International journal of computer vision*, 123(1):32–73, 2017.
- [95] Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. In *Advances in Neural Information Processing Systems 32*, pages 13–23. Curran Associates, Inc., 2019.
- [96] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805, 2018.

- [97] Piyush Sharma, Nan Ding, Sebastian Goodman, and Radu Soricut. Conceptual captions: A cleaned, hypernymed, image alt-text dataset for automatic image captioning. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2556–2565, 2018.
- [98] Niluthpol Chowdhury Mithun, Juncheng Li, Florian Metze, and Amit K. Roy-Chowdhury. Learning joint embedding with multimodal cues for cross-modal video-text retrieval. In *Proceedings of the 2018 ACM on International Conference on Multimedia Retrieval, ICMR '18*, page 19–27, New York, NY, USA, 2018. Association for Computing Machinery.
- [99] Youngjae Yu, Jongseok Kim, and Gunhee Kim. A joint sequence fusion model for video question answering and retrieval. In *Proceedings of the European Conference on Computer Vision (ECCV)*, September 2018.
- [100] Jun Xu, Tao Mei, Ting Yao, and Yong Rui. Msr-vtt: A large video description dataset for bridging video and language. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.
- [101] Yuncheng Li, Yale Song, Liangliang Cao, Joel Tetreault, Larry Goldberg, Alejandro Jaimes, and Jiebo Luo. Tgif: A new dataset and benchmark on animated gif description. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.
- [102] Rafal Józefowicz, Oriol Vinyals, Mike Schuster, Noam Shazeer, and Yonghui Wu. Exploring the limits of language modeling. *CoRR*, abs/1602.02410, 2016.
- [103] Jianfeng Dong, Xirong Li, Chaoxi Xu, Shouling Ji, Yuan He, Gang Yang, and Xun Wang. Dual encoding for zero-example video retrieval. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.
- [104] Xiang Wu, Da Chen, Yuan He, Hui Xue, Mingli Song, and Feng Mao. Hybrid sequence encoder for text based video retrieval. In *TRECVID*, 2019.
- [105] Xirong Li, Jianfeng Dong, Chaoxi Xu, Jing Cao, Xun Wang, and Gang Yang. Renmin university of china and zhejiang gongshang university at trecvid 2018: Deep cross-modal embeddings for video-text retrieval. In *TRECVID*, 2018.
- [106] Xirong Li, Jinde Ye, Chaoxi Xu, Shanjinwen Yun, Leimin Zhang, Xun Wang, Rui Qian, and Jianfeng Dong. Renmin university of china and zhejiang gongshang university at trecvid 2019: Learn to search and describe videos. In *TRECVID*, 2019.
- [107] Pascal Mettes, Dennis C. Koelma, and Cees G.M. Snoek. The imagenet shuffle: Reorganized pre-training for video event detection. In *Proceedings of the 2016 ACM on International Conference on Multimedia Retrieval*,

- ICMR '16, page 175–182, New York, NY, USA, 2016. Association for Computing Machinery.
- [108] Tianqi Chen, Mu Li, Yutian Li, Min Lin, Naiyan Wang, Minjie Wang, Tianjun Xiao, Bing Xu, Chiyuan Zhang, and Zheng Zhang. Mxnet: A flexible and efficient machine learning library for heterogeneous distributed systems. *CoRR*, abs/1512.01274, 2015.
  - [109] Relja Arandjelovic, Petr Gronat, Akihiko Torii, Tomas Pajdla, and Josef Sivic. Netvlad: Cnn architecture for weakly supervised place recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.
  - [110] Feng Mao, Xiang Wu, Hui Xue, and Rong Zhang. Hierarchical video frame sequence representation with deep convolutional graph network. In *Proceedings of the European Conference on Computer Vision (ECCV) Workshops*, September 2018.
  - [111] Fartash Faghri, David J. Fleet, Jamie Ryan Kiros, and Sanja Fidler. VSE++: improved visual-semantic embeddings. *CoRR*, abs/1707.05612, 2017.
  - [112] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 5998–6008. Curran Associates, Inc., 2017.
  - [113] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692, 2019.
  - [114] Chen Sun, Fabien Baradel, Kevin Murphy, and Cordelia Schmid. Contrastive bidirectional transformer for temporal representation learning. *CoRR*, abs/1906.05743, 2019.
  - [115] George Awad, Jonathan Fiscus, David Joy, Martial Michel, Alan Smeaton, Wessel Kraaij, Maria Eskevich, Robin Aly, Roeland Ordelman, Marc Ritter, et al. Trecvid 2016: Evaluating video search, video event detection, localization, and hyperlinking. 2016.
  - [116] Emine Yilmaz and Javed A. Aslam. Estimating average precision with incomplete and imperfect judgments. In *Proceedings of the 15th ACM International Conference on Information and Knowledge Management, CIKM '06*, page 102–111, New York, NY, USA, 2006. Association for Computing Machinery.

# List of Figures

1.1	Visualization of RGB and HSV histogram of a single image . . . .	8
1.2	Comparison of early fusion, late fusion and 3D convolutions . . . .	9
1.3	C3D architecture . . . . .	10
1.4	TransNet shot boundary detection network architecture . . . . .	13
1.5	Visualization of the evaluation approach . . . . .	14
1.6	Performance of the best TransNet model . . . . .	16
1.7	Examples of additional transition types . . . . .	20
1.8	Color transfer augmentation technique between two shots . . . . .	21
1.9	TransNet V2 shot boundary detection network architecture . . . .	22
1.10	DDCNN V2 cell with 4F filters . . . . .	23
1.11	Learnable frame similarities computation with visualization of Pad + Gather operation . . . . .	24
1.12	Frame sequence with its ground truth . . . . .	27
1.13	Ambiguous frame sequence from the RAI dataset . . . . .	30
1.14	Performance of the best TransNet V2 model . . . . .	31
1.15	Example TransNet V2 predictions compared to ground truth . . .	32
1.16	Difficult hard cuts from BBC Planet Earth dataset . . . . .	34
1.17	Visual comparison of TransNet and TransNet V2 . . . . .	36
2.1	ViLBERT model for image/description matching . . . . .	38
2.2	Max-margin ranking loss and noise-contrastive estimation loss. . .	39
2.3	A diagram of W2VV++ model . . . . .	41
2.4	Schema of Transformer-based encoder such as BERT . . . . .	45
2.5	Comparison between W2VV++ and our BERT extension, queries in favor of BERT . . . . .	51
2.6	Comparison between W2VV++ and our BERT extension, queries in favor of W2VV++ . . . . .	52
2.7	Comparison between W2VV++ and our BERT extension, unclear and wrong queries . . . . .	52

# List of Tables

1.1	Meta-parameter grid search results . . . . .	15
1.2	Per video results on the RAI dataset . . . . .	17
1.3	Performance comparison of related works on the RAI dataset . . .	17
1.4	Comparison of the original TransNet and TransNet V2 . . . . .	29
1.5	Comparison of TransNet V2 with related work . . . . .	29
1.6	Effects of real and artificially generated transitions on TransNet V2 performance . . . . .	33
1.7	Effect of individual components on the performance of TransNet V2	34
1.8	Effect of network's depth on performance . . . . .	34
1.9	Effect of augmentation methods on the performance . . . . .	35
2.1	Model comparison on TRECVID AVS tasks . . . . .	48
2.2	Model comparison on 20k-V3C1 dataset . . . . .	48
2.3	Model comparison on TRECVID 2016 Video-to-text dataset . . .	48
2.4	Comparison of max-margin loss variants . . . . .	49
2.5	Comparison of NCE loss and max-margin loss . . . . .	49
2.6	Performance dependency on the joint space dimension . . . . .	50
2.7	Performance dependency on number of patches per frame . . . . .	50