2019 International Conference on Document Analysis and Recognition (ICDAR)

# Content Extraction from Lecture Video via Speaker Action Classification based on Pose Information

Fei Xu, Kenny Davila, Srirangaraj Setlur, Venu Govindaraju
*Computer Science and Engineering*
*University at Buffalo, SUNY*
*Buffalo, NY, US*
{*fxu3, kennydav, setlur, govind*}*@buffalo.edu*

***Abstract*—Online lecture videos are increasingly important e-learning materials for students. Automated content extraction from lecture videos facilitates information retrieval applications that improve access to the lecture material. A significant number of lecture videos include the speaker in the image. Speakers perform various semantically meaningful actions during the process of teaching. Among all the movements of the speaker, key actions such as writing or erasing potentially indicate important features directly related to the lecture content. In this paper, we present a methodology for lecture video content extraction using the speaker actions. Each lecture video is divided into small temporal units called action segments. Using a pose estimator, body and hands skeleton data are extracted and used to compute motion-based features describing each action segment. Then, the dominant speaker action of each of these segments is classified using Random forests and the motion-based features. With the temporal and spatial range of these actions, we implement an alternative way to draw key-frames of handwritten content from the video. In addition, for our fixed camera videos, we also use the skeleton data to compute a mask of the speaker writing locations for the subtraction of the background noise from the binarized key-frames. Our method has been tested on a publicly available lecture video dataset, and it shows reasonable recall and precision results, with a very good compression ratio which is better than previous methods based on content analysis.**

***Keywords*-Lecture Video Summarization; Action Classification; Temporal Segmentation; Key-frame Extraction**

## I. Introduction

Massive open online courses (MOOCs) and other learning portals provide a variety of online educational resources to large audiences. Among all these resources, lecture videos have become a very important medium due to their potential to impart knowledge while improving student engagement [1]. Given the popularity and increasing numbers of available lecture videos, automatic summarization of these videos could help the audience to skim their content without the need to go through the entire video or seek to specific content within the video.

In many existing lecture recordings, the speaker uses whiteboards or blackboards to provide handwritten explanations. In order to create systems for navigation and search of such lecture videos, it is important to consider the lecture content rather than relying on general video meta-data. Note that audio transcripts might be insufficient for graphical content (e.g. mathematical expressions) that might be described only on the handwritten whiteboard/blackboard content. Despite the availability of hardware capable of capturing this handwritten content directly, not that many lecture video recordings are being produced with such hardware. It would be very time-consuming to manually summarize many of these hour-long videos. As a result, we need automatic methods to extract meaningful key-frames to summarize whiteboard content from lecture videos.

During the course of a lecture, the speaker will perform various actions that have very distinctive features. Among these, writing and erasing are key actions used to generate or change the handwritten lecture content seen in these videos. We hypothesize that a complete system for lecture video summarization should consider the speaker actions explicitly. However, most of the existing methods for this purpose usually neglect the additional information provided by these speaker actions.

Based on our hypothesis, this paper presents a novel methodology for the extraction and summarization of the handwritten lecture video content without really looking at the content itself. Instead, we use a pose estimator [2], which provides body and hands skeleton data, to extract motion-based features for speaker action classification based on small temporal units called action segments. Then, by considering the spatio-temporal range of key actions, we extract the handwritten content by drawing video key-frames, as opposed to traditional methods which are purely based on handwritten content analysis. Then, we apply an existing model for whiteboard image binarization [3] on the extracted lecture video key-frames. Under the assumption that these lectures were recorded using a fixed camera, we further remove the background noise using an estimation of the handwritten content region (a foreground mask) based on the skeleton data of the hand of the speaker.

We tested our model using the AccessMath lecture video dataset [4]. Compared to recent works based on handwritten content analysis [3, 5], our proposed method obtains similar handwritten content recall and precision levels on the same

2379-2140/19/$31.00 ©2019 IEEE
DOI 10.1109/ICDAR.2019.00171



Authorized licensed use limited to: University at Buffalo Libraries. Downloaded on August 31,2020 at 14:02:45 UTC from IEEE Xplore. Restrictions apply.

dataset. At the same time, our summaries use 25% fewer key-frames than these methods (better compression ratio). However, our model is still complementary to content-based models, and combined versions using both content and speaker actions could produce better results. Our code and data annotations have been made publicly available at: https://github.com/adaniefei/AccessMath_Pose

## II. Background

Our work is related to general video summarization and key-frame extraction, especially for lecture videos. In addition, we consider methods for lecture speaker action analysis, and we briefly cover these topics in this section.

**Video Summarization.** This process generates meaningful short video representations. Video summaries are valuable to users and can be used for applications like indexing, detection, and retrieval of video content. In their survey, Truong and Venkatesh [6] define two types of video summarizations: key-frames and video skims. Key-frames are images that preserve the most significant video content. Video skims are short segments from the original video which might enhance the video summary with audio and motion features. Alternative, the usage of key-objects has been proposed for video summarization [7]. While the audio in lecture videos contains relevant information, we focus on the handwritten content by using key-frames.

**Key-frame Extraction.** Many of these methods are based on shot boundary detection. A shot is a sequence of frames captured by one camera without interruptions. These frames have redundancies and strong content-based correlation [8], and the goal is to select key-frames containing the most salient objects from the shot. Typically, frame-level visual features such as color histogram, optical flow, motion intensity, etc. are used to extract the best frames as key-frames. General key-frame extraction models can be adapted to lecture videos based on slides, where one might attempt to identify the slide transitions as shot boundaries and aim to get one key-frame per slide [9, 10]. However, lecture videos using whiteboards (or handwritten content in general), might be recorded in one or several shots, with switching between whiteboard, speaker and other objects. Simply extracting key-frames from these lecture videos via shot boundaries might produce either too many redundant key-frames or too few for single-short videos.

**Lecture Video Summarization.** In this section, we focus on whiteboard lecture video summaries based on key-frames. The process is generally divided into two main steps: content extraction and content summarization. The content extraction step aims to identify and extract all relevant content from the whiteboard image. Many methods use binarization for this purpose. For images with very uniform illumination, methods based on global thresholds like Otsu [11] might work well. Noisier illumination settings require localized thresholds such as the Sauvola method [12]. Davila and Zanibbi [3] estimate and subtract the background from each image and then use hysteresis between Otsu's binarizer and their own Random Forest Binarizer. In our work, we use this method to binarize the summary key-frames. Other recent approaches have used Deep Learning for the detection [5, 13] and binarization [13] of the handwritten whiteboard content.

Handwritten content summarization in the form of key-frames has been carried out using different strategies. Estimations of writing and erasing actions based on the analysis of handwritten content peaks have been used to select key-frames [14]. Recently, conflict minimization has been used to select key-frames, where content regions are said to be in conflict if they occupy the same region of the whiteboard at different times [3, 5, 13]. A method for navigation and search of lecture videos based on summaries has been proposed by Davila and Zanibbi [15].

**Lecture Speaker Action Analysis.** Multiple works have studied the poses and gestures used by the speaker during lecturing. Some of these gestures are used to indicate the importance of specific lecture content [16, 17], and these indications can be used for effective lecture video summarization [17]. Pose descriptors have been used to identify such relevant gestures on lecture videos [16]. Non-verbal language, including speaker gestures, has been used to predict the rating that a lecture video might receive [18]. The usage of multiple modalities, including pose analysis through OpenPose [2], have been theoretically proposed by Ude et al. [19] as means for lecture video summarization. We also use pose information from OpenPose [2] as the input for our speaker action classification model. More general models for action classification from imagery have been covered in the survey by Herath et al. [20].

## III. Methodology

Our method extracts key-frames from lecture videos representing the handwritten whiteboard content without explicitly looking at the written content. The full architecture of the proposed model is shown in Figure 1. The input is a lecture video and the output is a set of binary key-frames. The speaker pose is analyzed and speaker motion features are extracted for later speaker action classification. The speaker action sequence is analyzed to create temporal lecture segments, which usually represent semantic sub-sections of the lecture. The bounding box of the speaker skeleton key-points is used to estimate a mask of the regions representing written content and erasures. Then, we produce a small set of key-frames (one per lecture video segment), which summarize the whole lecture content. After binarization of these key-frames, the background elements are removed using the content region mask. Each of these steps is described in the following sub-sections.

**Assumptions.** Our method can be applied to lecture videos recorded with a stationary camera focused on the



Authorized licensed use limited to: University at Buffalo Libraries. Downloaded on August 31,2020 at 14:02:45 UTC from IEEE Xplore. Restrictions apply.

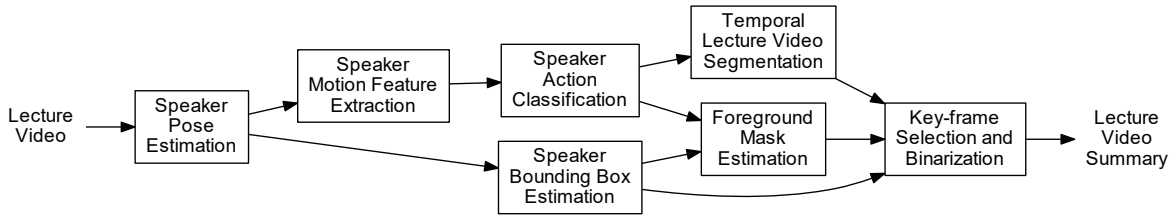



Figure 1. System Architecture. These are the major steps of our proposed methodology for lecture video summarization based on speaker actions.

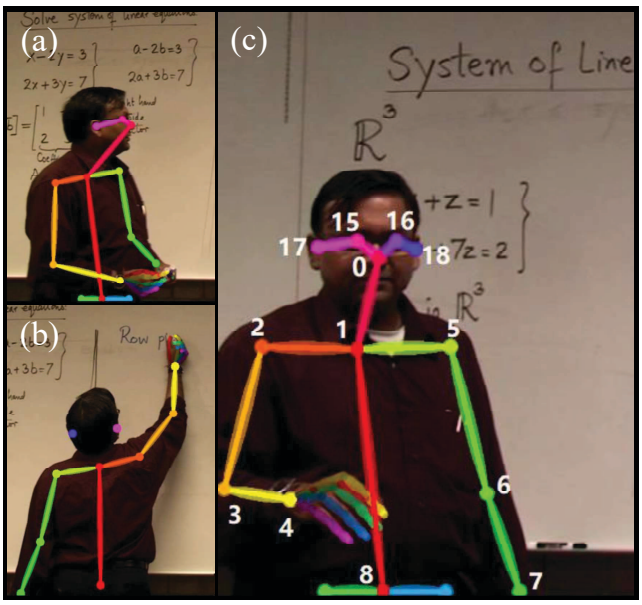



Figure 2. Examples of skeleton data produced by the OpenPose system [2] on lecture videos. We illustrate different poses such as the (a) side, (b) back and (c) the front. In (c), we have also included the original joint numbers as provided by OpenPose.

whiteboard without focus length variations (no zooming). The speaker should be in the camera scene most of the time, with the audience not being visible in the video. These assumptions are reasonable based on the characteristics of many available lecture videos on the internet.

### A. Speaker Pose Estimator

To identify speaker actions, our method classifies the speaker motions using the skeleton data of the speaker. We use OpenPose [2] to produce the skeleton including all visible joint positions (or key points) of both speaker body and hands for every frame. Three examples of skeleton data from OpenPose are illustrated in Figure 2. In Figure 2.c, joints are labelled following the order from the original OpenPose data. For lecture videos, we anticipate that the skeleton of the speaker upper body and writing hand is enough to classify actions. If the speaker is out of the image, no skeleton data is produced and we assign all joints a special value. Note that OpenPose does not consider temporal information at all and predicted locations might have jittering as well as missing data for frames where specific joints are occluded.

### B. Action Segments

The proposed speaker action classification system is like predicting the speaker status since the speaker is always performing one action at any given time. For example, the speaker can write, explain and erase without any temporal gaps (see Figure 3). Training videos are annotated with the speaker action using frame intervals. Then, we use fixed-length segments, which we call *action segments*, to represent and classify speaker actions. The action segments become the unit of analysis for later processes such as *temporal lecture video segmentation*, and *key-frame selection*.

**Action Labels.** We consider 8 actions: write, pick eraser, erase, drop eraser, out, out writing, out erasing, and explain. *Write* and *erase* are taken as the key-actions since they are directly related to the whiteboard content that we want to extract. *Pick/drop eraser* are linked to the *erase* action, but they are harder to detect due to their shortness and under-representation in the videos.

The *out* action captures when the speaker is out of the image and helps to extract key-frames without occluded content. The *out writing/erasing* are the situations when the speaker is mostly out of the image but possibly changes the whiteboard content around the image boundary. Humans can identify these actions from the temporal context. However, it is much harder for the algorithm since the hand might not be in the image. Finally, during the *explain* action, the speaker mostly moves around and gestures on handwritten content without changing it. Any actions other than these eight will also be labeled as *explain* as long as they do not affect the whiteboard content.

**Action Segment Labeling and Sampling.** Any given action segment will be labeled based on the majority class of the represented frames (see Figure 3). For a given video, action segments are sequentially sampled from the video timeline. To get more robust statistics for each action, especially around their boundaries, we densely sample the training videos by using overlapping tracks with different starting offsets (see Figure 3).



Authorized licensed use limited to: University at Buffalo Libraries. Downloaded on August 31,2020 at 14:02:45 UTC from IEEE Xplore. Restrictions apply.

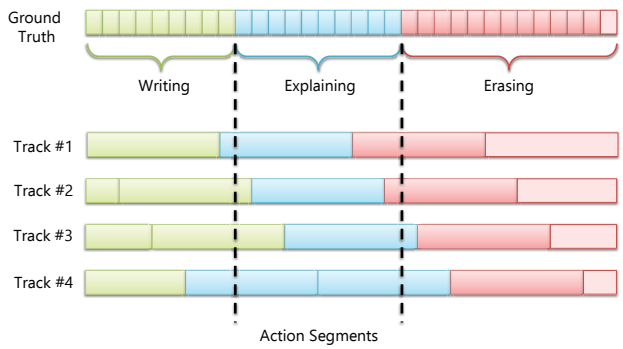
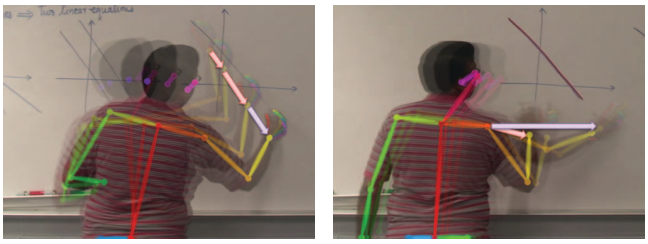



Figure 3. Action segments used for speaker action classification. At the top, the original frames are labeled by ranges where the speaker performs particular actions. Then, we show the multiple overlapping tracks that we use to sample action segments of a fixed length. Each action segment is assigned the majority class of the frames it covers.



(a) Joint Displacements (b) Pair-wise Joint Distances

Figure 4. Speaker motion analysis for action classification. We consider statistics from both the (a) joint displacements and (b) pair-wise joint distances. We illustrate this idea by using writing in (a) and erasing in (b).

### C. Speaker Action Classification

Different actions typically show different motion patterns. For each action segment, we extract speaker motion statistics to classify the speaker action. As shown in Figure 4, we use the frame-wise joint locations provided by OpenPose to extract two types of features: joint displacement features, and pair-wise joint distance features.

**Joint Displacement Feature.** For a given action segment, we compute temporal statistical features from raw and absolute, horizontal and vertical displacements of one selected joint for every pair of consecutive frames (see Figure 4.a). From the raw horizontal and vertical displacements, we compute the means, medians and covariance matrix (7 values). From the absolute differences, we compute the means only (2 values). We remove displacements between frame pairs where either of them has invalid skeleton data, obtaining between 0 and action segment length minus 1 displacements. When no valid displacements can be collected from the action segment, the feature values are set to 0. We add a confidence value to represent the percentage of valid displacements. A total of 10 values is used for each joint. Figure 4.a shows the joint displacements for joint 2.

**Pair-Wise Joint Distance Feature.** For each frame in one action segment, the pair-wise horizontal and vertical distance between joints is computed (see Figure 4.b). We compute the means and variances of both horizontal and vertical components (4 values). As with the previous feature type, we omit values for frames where either of the joints data is not available and also adds a confidence value representing the percentage of valid pair-wise distances for a total of 5 values per joint pair. Figure 4.b illustrates the pair-wise joint distance for joints $(2, 4)$.

**Skeleton Confidence Feature.** This value represents the percentage of frames having any valid skeleton data from the pose estimator during the action segment.

**Feature Normalization Factor.** Despite having multiple lecture videos of the same instructor, the relative height of the speaker in the image will vary from lecture to lecture due to changing environmental conditions and inconsistent camera shooting distance, and we need a method to reduce this variation. We use a particular pair of joints from the speaker skeleton to normalize any distance-based feature values. Since the upper body of the speaker is usually visible in the videos from the AccessMath dataset [4], we use the average of absolute distance between the joint pair $(1, 8)$ as our normalization factor (see Figure 2). This is done based on the knowledge that the speaker mostly shows his upper body within the scope of the frame in this dataset. In this sense, we use the global average of absolute distance between the joint pair $(1, 8)$ (see Figure 2) as our normalization factor since we observed this joint pair to have the most consistent size through each training video.

**Selected Joints and Features.** Since we take *write* and *erase* as key-actions, we choose joints around the speaker's head, shoulder and arms to get distinct motion-based features. Knowing the speaker handedness, we can select either joints $2, 3$ and $4$ (right-handed) or joints $5, 6$ and $7$ (left-handed), see Figure 2.c. Occluded joints also provide information when the speaker is facing the whiteboard, and we considered joints $0, 5$ and $6$ for a right-handed speaker. Similarly, the joint pair $(2, 5)$ provides information when the speaker turns around. For the AccessMath dataset, we selected the joints $2, 3, 4, 0, 5, 6$ to extract joint displacement features (60 values), and joint pairs $(2, 5), (2, 4)$ for extraction of Pair-wise Joint Distance features (10 values), which along with the skeleton confidence feature make a total of 71 features for speaker action classification. Other lecture video collections might require a different selection of joints based on the speaker handedness.

**Action Classifier.** For a given video, we create a sequence of action segments as described earlier, and we generate the 71 motion features for each action segment. We use Random Forests [21] to classify the speaker actions on each segment. The result is a sequence of speaker actions used later for temporal lecture video segmentation



Authorized licensed use limited to: University at Buffalo Libraries. Downloaded on August 31,2020 at 14:02:45 UTC from IEEE Xplore. Restrictions apply.

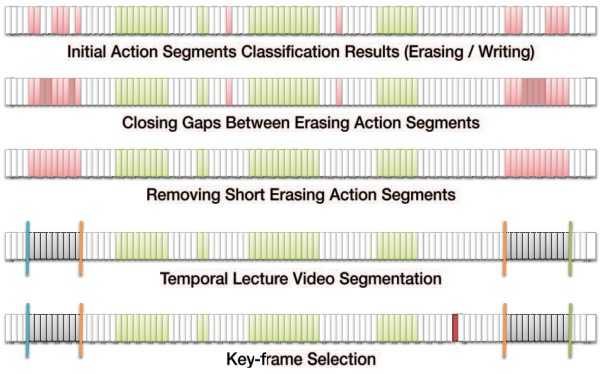



Figure 5. Temporal lecture video segmentation. The initial action segment classification results are refined by filling small gaps between non-contiguous erasing segments first, then short erasing segments are removed and the remaining ones are used to recursively split the video. From each resulting interval, at least one frame is extracted to generate the lecture video summary.

and foreground mask estimation.

### D. Temporal Lecture Video Segmentation

The input to this process is a sequence of speaker actions and the output is a set of disjoint lecture video segments. The goal is to identify the semantic temporal segments of the original lecture, where the speaker writes a considerable amount of whiteboard content and finalizes when the content is deleted from the whiteboard. In this sense, we can segment the video by detecting major erasing events in the lecture video timeline. Due to classification errors or short pauses taken by the speaker when erasing content, major erasing events might be split into multiple erasing intervals (see Figure 5). The temporal segmentation process starts by identifying intervals of consecutive action segments classified as erasing. Then, we identify major erasing events by merging any pair of erasing intervals having a small gap ($< 4$ seconds) between them as shown in Figure 5. Then, to avoid over-segmentation due to false positives of the erasing class, we remove short erasing intervals ($< 3$ seconds).

We run a greedy recursive segmentation approach that uses the resulting erasing intervals to produce a set of lecture video segments. Starting from a single video segment for the whole lecture, the greedy procedure ranks the erasing intervals within the segment by decreasing length and uses them to find a valid split interval where the lengths of the resulting segments would be above a minimum size ($\leq 50$ seconds). If a valid split interval is located, then the current video segment is split into two segments: before and after the erasing interval. Then the segmentation function is applied recursively on each segment. The recursion stops when the current video segment does not include any valid split interval. We used a grid search over training videos to identify good values for the thresholds used by this procedure.

### E. Foreground Mask Estimation

Under the assumption that the camera is always fixed, we use the speaker skeleton data to estimate the regions of the image where all writing and erasing actions take place. The output is a binary mask of the foreground regions where we anticipate that the handwritten content might be located. We start by computing frame-wise bounding boxes for the speaker writing hand using the skeleton data. We expand these bounding boxes by a factor of 25% because the skeleton data is based on joints. Then, based on action segment classification results, all bounding boxes associated with erasing and writing frames are combined to create the binary foreground masks (see Figure 6.c and 6.g).

### F. Key Frame Selection and Binarization

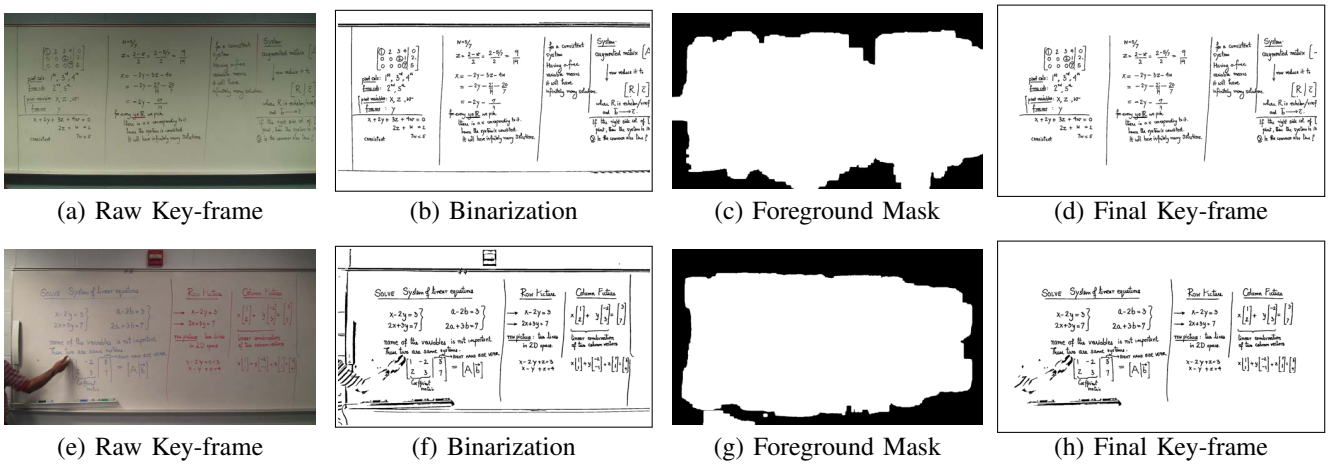
Based on the results from all the previous steps, a small set of frames representing the whole lecture video content is finally chosen. This requires selecting one or more key-frames per the semantic segment of the lecture video. In our model, we start with simple key-frame selection procedure which can handle easy key-frame extraction cases, and we fall back to a greedy algorithm to handle the hard cases.

The simpler key-frame selection algorithm locates intervals of *writing* and *out* actions first, then it checks if there is any *out* action segment after the last writing interval. This is multiple frames where the speaker is deemed to be out of the image after the last handwriting and before anything gets erased. If found, this represents an ideal case where all the handwritten content in the lecture video segment can be extracted cleanly using one frame (see Figure 6.a-d).

When the simpler algorithm fails, we use a greedy algorithm which tries to select a small set of key-frames covering all the handwritten content on the current lecture video segment. We generate a temporally aware mask of potential handwritten content, where every pixel records the index of the last frame when it might have been modified by the speaker during the current video segment. This is done based on sequential analysis of the writing events and their corresponding speaker hand bounding boxes. For a given frame, the mask can be used to estimate which pixels were already modified for the last time within the lecture segment and which ones could have been modified later. Then, we use the mask to select the frame which contains the largest number of pixels that were modified for the last time before the current frame time which are not occluded by the speaker bounding box (visible modified pixels). This frame is added to a temporary list, and the visible modified pixels covered by the frame are removed from the mask. The algorithm is repeated until the mask is empty or it cannot find a frame which covers at least 25% of pixels from the initial mask.



Authorized licensed use limited to: University at Buffalo Libraries. Downloaded on August 31,2020 at 14:02:45 UTC from IEEE Xplore. Restrictions apply.



(a) Raw Key-frame (b) Binarization (c) Foreground Mask (d) Final Key-frame

(e) Raw Key-frame (f) Binarization (g) Foreground Mask (h) Final Key-frame

Figure 6. Key-frame binarization and background removal. Each frame selected for the representation of a temporal lecture video segment is binarized, and the foreground mask estimation is used to remove background elements. In this example, the process is shown for both a correctly selected key-frame (a)-(d), and a key-frame which got erroneously selected due to noise in the pose estimation (e)-(h).

After binarization, all frames in the temporary list will be combined into a single key-frame using the OR operator.

To produce the lecture video summary, we binarize the selected key-frames using the same approach previously proposed by Davila and Zanibbi [3] for the original benchmark on the AccessMath Dataset (see Figure 6.b and 6.f). While this method is slow and potentially less effective than state-of-the-art binarization methods based on deep learning, it allows us to compare our model against theirs while avoiding another potential source of a performance difference.

To improve the precision of the final binary key-frames, we use the speaker skeleton bounding box to create a binary mask, and we only keep the connected components (CCs) that are mostly outside of the mask (75% of the CC pixels). Similarly, we also attempt to remove background elements by keeping only the CCs which are mostly contained within the foreground mask (75% of the CC pixels) computed in the previous step. The final result is binary images mostly containing handwritten whiteboard content with very few additional elements (see Figure 6).

## IV. Experiments

We test our approach using the AccessMath [4] dataset, which has 12 linear algebra lecture videos from a single speaker. These videos are 1080p at 30 FPS and have an average length of 49 minutes. The 12 videos are fully annotated at the binary level, with 5 videos reserved for training and 7 for testing [3]. We further annotated the speaker actions on the 5 training videos in order to train our action classifier model. These new annotations have been made publicly available.

### A. Action Segment Classification

In our first experiment, we try to identify an ideal length for action segments by considering lengths between 5 to 45 frames in increments of 5. We also test different numbers of overlapping tracks for action segment sampling: 1, 2, and 4 tracks. We use the training set to run a 5-fold cross-validation strategy, where one full video is reserved for validation every time. We use the scikit-learn [22] implementation of Random Forest classifier, with 64 trees, a maximum depth of 16, and the Gini criterion to chose each split. Different sampling strategies and action segment lengths result in a different number of samples for each training video. For example, for action segment length of 15, we obtained $24,417$, $48,830$ and $97,654$ training action segments for $1$, $2$ and $4$ tracks respectively. To get a comparable output for all conditions, we use a single track of action segments for each validation video and compute the frame-level accuracy by expanding the predicted action segment labels into individual frames. Our results are summarized in Figure 7.

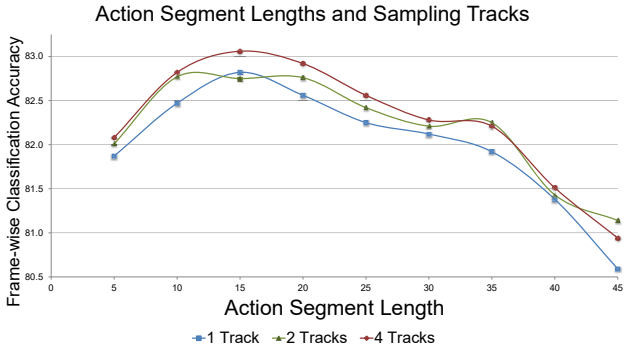



Figure 7. Frame-level speaker action classification accuracy. A comparison is made between the different action segment lengths (number of frames) versus different number of sampling tracks. Using additional tracks generally helps. Shorter segments generally achieve better frame-wise classification accuracy, but very short segments lose the ability to capture enough information about the speaker motion.



Authorized licensed use limited to: University at Buffalo Libraries. Downloaded on August 31,2020 at 14:02:45 UTC from IEEE Xplore. Restrictions apply.

As expected, the longer action segments lead to a decay in the frame-wise action classification accuracy since these impose a single label over a larger number of frames. However, shorter segments produce less stable statistics and more errors. An ideal segment length produces stable enough statistics with good accuracy both at segment and frame levels, and from Figure 7 we can observe that this is roughly achieved at a length of 15 frames. Furthermore, adding the parallel action segment tracks for the sampling of training segments generally help, and the best frame-level cross-validation accuracy is achieved using 4 tracks. These two parameters are fixed and a new action segment classifier is trained on the entire training set for the lecture video summarization experiments. Note that our feature vectors have a fixed length regardless of the action segment length. Other classification schemes can be explored where multiple segment lengths are combined in a pyramidal fashion.

### *B. Lecture Video Summarization*

We use the testing set from the AccessMath dataset to evaluate the effectiveness of our lecture video summarization model. We produce four versions of our summaries: unrefined binary, binary with speaker bounding box removal, binary with background removal, binary with both speaker and background removal. We compare against other models proposed for the AccessMath dataset which includes Conflict Minimization [3], Selection of Key-frames based on Maximums of the Sums of Content [3], and Handwritten Text Detection using Text Boxes [5] with and without temporal refinements. We followed the original evaluation protocol using the evaluation tools released by the authors of [3], where CCs from ground truth binary key-frames are matched against a set of summary key-frames and their recall, precision and f-score are computed both at global and per-frame levels. The system allows many-to-many matches between ground truth CCs and summary CCs in order to deal with split or merged summary CCs. Then, matching elements will be considered valid only if the pixel-wise recall and precision values are over 50%. Apart from content matching, the evaluation also considers video compression through the average number of summary frames produced by each method. Our final results are shown in Table I.

On average, our approach produces 12.29 frames per video, which is smaller than any of the baselines (see Table I). We found that the differences between the average number of frames produced by our method and each of the baselines were statistically significant (p-value $< 0.05$). In the ground truth, the ideal average is 11.14 frames per testing video, which means we are close to the ideal segmentation, but there is still room for improvement.

Our recall of CCs is also very high, better than the model based on TextBoxes [5], but slightly worse than the original system by Davila and Zanibbi [3]. As expected, our precision values are lower than both models because our unrefined binary key-frames include additional elements since we do not detect handwritten content explicitly. Pose estimation errors quickly propagate through our summarization pipeline. When the speaker body is mostly out of the image, OpenPose might fail to detect the partial skeleton. Then, our model incorrectly assumes that the speaker is out of a frame, and it might badly select it as a key-frame. In this scenario, the speaker mask will be empty and the speaker will not be removed in the refined key-frame causing a loss in precision, and potentially a loss in recall as well if the speaker is occluding handwritten content. Figure 6.e-h illustrates one of such failure cases.

Using both the foreground and speaker masks helps effectively remove many irrelevant elements thus increasing our precision. In particular, the foreground mask helps remove a large number of background elements (see Table I). We anticipate that the foreground mask would fail in cases where handwritten content has been placed on the whiteboard before the recording starts, and if the speaker never erases it on the video, then it might not be extracted at all. However, we never observed this happening over relevant handwriting in the AccessMath Dataset.

Finally, some under-represented actions such as *out writing* and *out erasing* also provide relevant information which could help our approach. However, we decided to omit them from the temporal lecture video segmentation pipeline, since they could not be detected reliably.

## V. Conclusion

In this paper, we have proposed a lecture video summarization model which is able to produce a small set of binary key-frames containing most of the handwritten whiteboard content from each lecture. On average, our model was able to achieve a better compression ratio than existing models tested on the same dataset. Despite lacking explicit handwriting analysis, our model achieves a level of content recall comparable to previous methods while using fewer key-frames. However, precision is lower because our method cannot remove many non-textual objects from the final binary frames. Our analysis of the speaker actions regions helps us to mitigate this issue by generating a foreground mask which improves the final precision.

In the future, we would like to get a better temporal analysis of the pose estimations, in particular, to track better the hands of the speaker, and to get more consistent skeleton predictions in general. Besides the speaker skeleton, an exact mask of the speaker pixels per frame can make our model very precise. In addition, while our current action classifier produces reasonably accurate results (83.06% in cross-validation), there is room for improvement. To avoid the trade-off between action segment lengths and frame-level accuracy, we would like to explore a pyramidal approach to improve the overall frame-wise accuracy. We also anticipate that more robust features and sequence-oriented machine



Authorized licensed use limited to: University at Buffalo Libraries. Downloaded on August 31,2020 at 14:02:45 UTC from IEEE Xplore. Restrictions apply.

Table I
LECTURE VIDEO SUMMARIZATION RESULTS FOR DIFFERENT METHODS FOR THE ACCESSMATH TESTING VIDEOS.

| METHOD | Frames MEAN ($\sigma$) | MEAN GLOBAL REC. | MEAN GLOBAL PREC. | MEAN GLOBAL F1 | MEAN PER FRAME REC. | MEAN PER FRAME PREC. | MEAN PER FRAME F1 |
|---|---|---|---|---|---|---|---|
| Conflict Minimization [3] | 17.29 (4.54) | 96.28 | 93.56 | 94.90 | 95.73 | 92.21 | 93.94 |
| Maximum Content Sums [3] | 34.42 (10.15) | **96.49** | 94.51 | **95.49** | **96.13** | 91.95 | **93.99** |
| TextBoxes Text Detection (no Fine Tuning) [5] | 17.00 (4.62) | 88.29 | **95.39** | 91.70 | 88.41 | **94.22** | 91.22 |
| TextBoxes Text Detection (w. Fine Tuning) [5] | 19.43 (5.32) | 92.33 | 94.16 | 93.23 | 91.69 | 93.45 | 92.56 |
| Speaker Actions (Binary) | **12.29 (2.14)** | 95.91 | 78.10 | 86.09 | 94.37 | 77.07 | 84.85 |
| Speaker Actions (Speaker Removal) | **12.29 (2.14)** | 95.91 | 78.75 | 86.49 | 94.34 | 77.66 | 85.19 |
| Speaker Actions (Background Removal) | **12.29 (2.14)** | 95.89 | 85.59 | 90.45 | 94.21 | 84.46 | 89.07 |
| Speaker Actions (Speaker and Background Removal) | **12.29 (2.14)** | 95.89 | 86.28 | 90.83 | 94.18 | 85.15 | 89.44 |

learning algorithms (e.g. LSTM) could better exploit the temporal nature of the speaker actions.

We also want to extend our method to work on other lecture videos which include multiple shots from different cameras and zoom levels. Most of our action classification features should be applicable to such lecture videos. Finally, if we combine our key-frame selection model with explicit text detection [5, 13], the precision of the resulting model should be much higher while also keeping our high compression ratio. Text detection results might also help us avoid selecting bad key-frames.

## ACKNOWLEDGMENT

This material was partially supported by the National Science Foundation under Grants No. 1640867 (OAC/DMR), and No. 1651118 (SBE).

## REFERENCES

[1] P. J. Guo, J. Kim, and R. Rubin, "How video production affects student engagement: An empirical study of mooc videos," in *Proceedings of the first ACM conference on Learning@ scale conference*. ACM, 2014, pp. 41–50.

[2] Z. Cao, G. Hidalgo, T. Simon, S.-E. Wei, and Y. Sheikh, "OpenPose: realtime multi-person 2D pose estimation using Part Affinity Fields," in *arXiv preprint arXiv:1812.08008*, 2018.

[3] K. Davila and R. Zanibbi, "Whiteboard video summarization via spatio-temporal conflict minimization," in *2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*, vol. 1. IEEE, 2017, pp. 355–362.

[4] K. Davila, A. Agarwal, R. Gaborski, R. Zanibbi, and S. Ludi, "Accessmath: Indexing and retrieving video segments containing math expressions based on visual similarity," in *2013 IEEE Western New York Image Processing Workshop (WNYIPW)*. IEEE, 2013, pp. 14–17.

[5] B. U. Kota, K. Davila, A. Stone, S. Setlur, and V. Govindaraju, "Automated detection of handwritten whiteboard content in lecture videos for summarization," in *2018 16th International Conference on Frontiers in Handwriting Recognition (ICFHR)*. IEEE, 2018, pp. 19–24.

[6] B. T. Truong and S. Venkatesh, "Video abstraction: A systematic review and classification," *ACM transactions on multimedia computing, communications, and applications (TOMM)*, vol. 3, no. 1, p. 3, 2007.

[7] J. Meng, H. Wang, J. Yuan, and Y.-P. Tan, "From keyframes to key objects: Video summarization by representative object proposal selection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 1039–1048.

[8] W. Hu, N. Xie, L. Li, X. Zeng, and S. Maybank, "A survey on visual content-based video indexing and retrieval," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 41, no. 6, pp. 797–819, 2011.

[9] H. Yang and C. Meinel, "Content based lecture video retrieval using speech and video text information," *IEEE Transactions on Learning Technologies*, vol. 7, no. 2, pp. 142–154, 2014.

[10] K. Li, J. Wang, H. Wang, and Q. Dai, "Structuring lecture videos by automatic projection screen localization and analysis," *IEEE transactions on pattern analysis and machine intelligence*, vol. 37, no. 6, pp. 1233–1246, 2015.

[11] N. Otsu, "A threshold selection method from gray-level histograms," *IEEE transactions on systems, man, and cybernetics*, vol. 9, no. 1, pp. 62–66, 1979.

[12] J. Sauvola and M. Pietikäinen, "Adaptive document image binarization," *Pattern recognition*, vol. 33, no. 2, pp. 225–236, 2000.

[13] B. U. Kota, K. Davila, A. Stone, S. Setlur, and V. Govindaraju, "Generalized framework for summarization of fixed-camera lecture videos by detecting and binarizing handwritten content," *International Journal on Document Analysis and Recognition (IJDAR)*, pp. 1–13, 2019.

[14] C. Choudary and T. Liu, "Summarization of visual content in instructional videos," *IEEE Transactions on Multimedia*, vol. 9, no. 7, pp. 1443–1455, 2007.

[15] K. Davila and R. Zanibbi, "Visual search engine for handwritten and typeset math in lecture videos and latex notes," in *2018 16th International Conference on Frontiers in Handwriting Recognition (ICFHR)*. IEEE, 2018, pp. 50–55.

[16] J. R. Zhang and J. R. Kender, "Identifying salient poses in lecture videos," in *2011 18th IEEE International Conference on Image Processing*. IEEE, 2011, pp. 2353–2356.

[17] Y. Tian and M.-L. Bourguet, "Lecturers' hand gestures as clues to detect pedagogical significance in video lectures," in *Proceedings of the European Conference on Cognitive Ergonomics*. ACM, 2016, p. 2.

[18] D. S. Cheng, H. Salamin, P. Salvagnini, M. Cristani, A. Vinciarelli, and V. Murino, "Predicting online lecture ratings based on gesturing and vocal behavior," *Journal on Multimodal User Interfaces*, vol. 8, no. 2, pp. 151–160, 2014.

[19] J. Ude, B. Schüller, R. Wegener, and J. Cassens, "A pipeline for extracting multi-modal markers for meaning in lectures," in *Proceedings of the Tenth International Workshop on Modelling and Reasoning in Context*, vol. 2134, 2018, pp. 16–21.

[20] S. Herath, M. Harandi, and F. Porikli, "Going deeper into action recognition: A survey," *Image and vision computing*, vol. 60, pp. 4–21, 2017.

[21] L. Breiman, "Random forests," *Machine Learning*, vol. 45, no. 1, pp. 5–32, Oct 2001. [Online]. Available: https://doi.org/10.1023/A:1010933404324

[22] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.



Authorized licensed use limited to: University at Buffalo Libraries. Downloaded on August 31,2020 at 14:02:45 UTC from IEEE Xplore. Restrictions apply.