Multimedia Tools and Applications (2024) 83:6731–6753
https://doi.org/10.1007/s11042-023-15565-w

AutoTag: automated metadata tagging for film post-production

Marcelo Sandoval-Castañeda[1] · **Scandar Copti**[2] · **Dennis Shasha**[3]

Received: 9 August 2021 / Revised: 11 October 2022 / Accepted: 18 April 2023 /
Published online: 16 June 2023

Abstract
Film post-production can be time- and money-inefficient. The reason is that a lot of the work involves a person or group of people, called metadata taggers, going through each individual piece of media and marking it up with relevant tags, such as the scene number, transcripts, and the type of shot for video footage. Such a task is particularly time-consuming for films with high shooting ratios (i.e., footage shot/footage shown). AutoTag automates much of the tagging process across 16 languages, saving both time and money. We describe the algorithms and implementation of AutoTag and report on some case studies.

Keywords Post-production · Film editing · Speech recognition · Image classification · Google speech-to-text

1 Introduction

An average high production movie has a shooting ratio between 100:1 and 1000:1. A metadata tagger's task is to watch all of these media and tag it. Assuming the most popular runtime of a movie, 101 minutes, and a conservative shooting ratio of 100, the time it would take for a single person to achieve this task is 168 hours, without even taking into account the audio recording associated with each piece of video. This is at least 21 days of a person

✉ Marcelo Sandoval-Castañeda
marcelo.sc@nyu.edu

Scandar Copti
scandar.copti@nyu.edu

Dennis Shasha
shasha@cims.nyu.edu

[1] Department of Computer Science, New York University Abu Dhabi, Saadiyat Island, Abu Dhabi, 129188, Abu Dhabi, UAE

[2] Department of Film and New Media, New York University Abu Dhabi, Saadiyat Island, Abu Dhabi, 129188, Abu Dhabi, UAE

[3] Courant Institute of Mathematical Sciences, New York University, 251 Mercer St, New York, 10012, NY, USA

working full-time, both delaying the post-production process significantly and placing a significant burden on independent filmmakers who often have a small budget.

AutoTag automates metadata tagging for Adobe Premiere Pro. This paper discusses the algorithms, implementation and user experiments. See Fig. 1 for an overview of the proposed workflow.

The main contributions of AutoTag are to automate the following tasks:

1. **Tag video footage with shot type (from close-up to long).** This feature applies machine learning techniques [4, 5, 20] to cinematic shot identification. Unlike previous work, we use an unsupervised learning approach that relies on a ResNet SSD for facial recognition[15]. This process is described in Section 5.2. Sections 7.2 and 7.3 present experiments.
2. **Tag each video footage and audio file with its corresponding scene number and associated portions from a screenplay.** Our approach, as explained in Section 2.4, tags video footage with its corresponding screenplay portion either by matching the footage's audio track or audio files to the linear timecode. Section 7.5 presents a case study.
3. **Create metadata for each media type and make it available within Adobe Premiere Pro as well as provide a search tool capable of filtering through our metadata more efficiently than Premiere.** This is described and in Section 6.

The source code is freely available at the GitHub repository https://www.github.com/mudtriangle/autotag.

Section 2 introduces film terminology and tools, such as screenplay formatting, editing software, and linear timecode. Section 3 establishes the technological tools and frameworks used in AutoTag. Section 4 compares AutoTag to existing work. Section 5 summarizes the primary use case and workflow of AutoTag. Section 5.1 describes the algorithm necessary to associate each media file to its corresponding scene. Section 5.2 describes the algorithm used for identification of cinematographic shot types. Section 6 shows the interface provided by AutoTag within Adobe Premiere Pro. Section 7 describes the experimental results obtained from four film projects. Section 8 proposes potential improvements and future work.

2 Background

2.1 Relevant film-related definitions

To ensure this paper is self-contained, we present a brief description of the critical film terminology.

- **Non-linear editing:** A form of editing that does not modify the original content, as opposed to classical editing of movies, which involved cutting and stitching physical film. Adobe Premiere Pro is an example of non-linear editing (NLE) software.
- **Raw footage:** The media that has been shot or recorded but has not yet been edited.
- **Shooting ratio:** The amount of raw footage shot divided by the final duration of the movie.
- **Aspect ratio:** The ratio of width to height. The most common aspect ratios include 21:9 (2.33:1), 16:9 (1.77:1), and 4:3 (1.33:1).
- **Screenplay:** The written script for a movie, divided in scenes.
- **Scene:** Within the context of a Screenplay, a scene is a collection of events that happen within the same time frame and location.

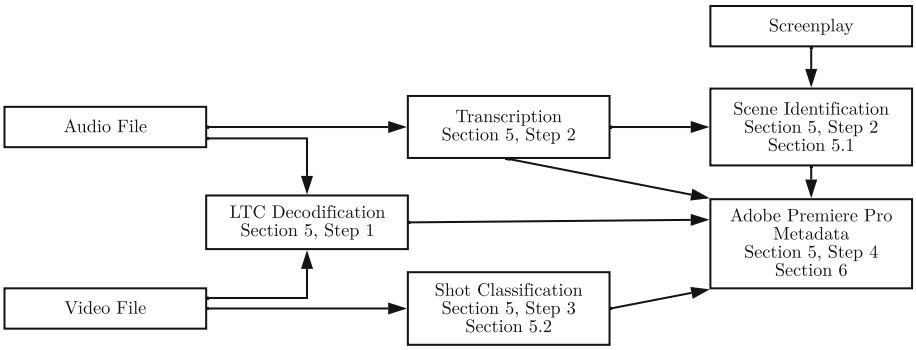

Fig. 1 The workflow of AutoTag, assuming the audio and video files contain linear timecode (LTC) information and that there is an associated screenplay

- **Timecode:** A sequence of numbers that contain the time information of media. These times are used for synchronization and to identify material in recorded media. It usually includes hour, minute, second, and frame, in the format of hh:mm:ss:ff.
- **Frame:** A single image that is part of a video. Typically, there are between 23.97 and 30 frames per second of video. However, the number can be arbitrarily smaller or larger.
- **Focus:** The sharp area of an image, as opposed to the blur generated by the lens for objects closer to or farther away from the focus distance.
- **Motion blur:** Streaks generated by objects moving faster than what the camera can capture, which is determined by the frames per second in the camera.
- **Shot type:** The relationship between the camera and the object(s) being shot. Examples are close-ups and wide shots. See Fig. 2.

2.2 Screenplay formatting software

One step in automatic metadata tagging process for fictional filmmaking is parsing the relevant screenplay and identifying the scenes in it. Generally, every screenplay follows relatively strict formatting rules (Fig. 3), allowing it to be parsed automatically. For this particular project, the screenplays parsed are Final Draft (.fdx) files, which is the industry standard screenplay writing software.

2.3 Non-linear video editing software

Adobe Premiere Pro is one of the three most important non-linear editing (NLE) software products and has become the leading software for independent filmmakers. It has also been used in high-tier productions such as *Mindhunter* (2017) and *Deadpool* (2016). In addition, its large community of third-party plug-in developers and the accessibility of documentation for plug-in development make it the ideal platform for a tool like AutoTag, though the functionally of AutoTag could benefit Adobe Premiere Pro's two main competitors: Avid Media Composer and Blackmagic's Davinci Resolve.

Adobe Premiere Pro's project structure consists of:

- **Project:** A directory, into which files can be imported. Its extension is .prproj.
- **Media file, Original file, or, sometimes, just file:** The file that contains the audio, visual, or audiovisual information.

(a) A **close-up shot**. Usually includes the subject's head and neck in a tight frame.

(b) A **medium close-up shot**. Usually includes the subject's head and shoulders.

(c) A **medium shot**. Usually goes from the subject's torso or hips to the top of the head.

(d) An **American shot**. Usually goes from the subject's knees to the top of the head.

(e) A **long (or wide) shot**. Usually includes the full body of the subject.

Fig. 2 Most common shot types in filmmaking, extracted from the movie *Breakfast at Tiffany's*

- **Media Item:** An entity that consists of a symbolic link to a media file.
- **Bin:** A directory-like entity inside a project. It can contain media items, other bins, and sequences.
- **Smart Bin:** A sub-type of bin that contains copies of all the media items that satisfy a given condition about their metadata. For instance, if a user wants to see all the media items that have "Scene 3" written in their metadata, they can create a smart bin that contains only media items in which "Scene 3" is a substring of their metadata tags.
- **Sequence:** A timeline object that contains references to media items in a sequence. These are used to do most of the actual editing: cutting and pasting videos together, including multiple tracks, handling transitions, etc.

All of these entities and directories contain metadata that can be edited and referenced. These include, but are not limited to, descriptions, log notes, scene numbers, and shot numbers. AutoTag's objective is to generate such tags automatically, which then can be searched using smart bins by the post-production crew.

2.4 Linear timecode

Linear Timecode (LTC) is the industry standard to synchronize different devices (e.g. an audio recording device and a camera). The format was established by the Society of Motion

```
INT. MESSY OFFICE - DAY

READER sits at their computer, wondering what a screenplay
looks like. MARCELO helps them.

                    READER
          So, what are we doing?

                    MARK
          We're having an imaginary
          conversation to showcase the
          formatting of a screenplay.
```

Fig. 3 The typical layout of a screenplay

Picture and Television Engineers (SMPTE), and consists of 80 bits encoded in audio frequencies. When translated, it contains all the time information needed for synchronization, as well as custom fields defined by the user in the setup.

3 Technological tools

The following subsections describes the technologies AutoTag uses and adapts to achieve its goals.

3.1 Dialogue transcriptions

Machine Learning-based speech-to-text models are frequently used in a variety of fields, such as translation[2, 3] and chatbot development [16]. In addition, there are many speech-to-text tools, each of them promising high-quality results. Kepuska and Bohouta [8] performed tests using three of the most popular models for speech recognition: one developed by Microsoft, one by Google, and one by Carnegie Mellon University. Their results suggest that Google's Speech Recognition Tool is the best, with a word error rate (WER) of 9%, compared to Microsoft's 18% and CMU's 37%.

AutoTag has two uses for transcripts: to identify media that correspond to a scene in a screenplay or to search for media elements (audio or video) by keyword. This is challenging for our application, because, as Khilari [9] points out, the main challenge of any speech-to-text tool is to provide coherent texts. Any word out of place can potentially create unintelligible sentences, especially in film where the actors sometimes deviate from the script.

3.2 Approximate text matching

Multiple techniques have been developed to construct algorithms that can determine whether a piece of text is similar enough to another. Ukkonen [21] presents an edit distance algorithm that runs in time $O(|A||B|)$ in the worst case, for two strings A and B. In addition, they propose a technique they call q-gram distance (or n-gram distance). An n-gram is simply a set of consecutive substrings of length n of a given string s: $s[1..n]$, $s[2..n+1]$, $s[3..n+2]$, ... Two similar strings will have a large number of common n-grams.

Neculoiu, Versteegh, and Rotaru [12] suggest a more modern approach to this problem. They use a Siamese recurrent neural network to assess whether two strings are similar enough. Their experiments yielded particularly impressive results when handling typos and annotations. This model involves substantial pre-processing, accounting for typos, synonyms, compound words, and annotations in text.

Because our task is to associate media with scenes which is easier than understanding a full transcript, AutoTag uses the simpler n-gram similarity with a few alterations that account for a screenplay's structure. AutoTag constructs n-grams of lengths 1 to 5 for each scene, and for the transcription of a given media file. Jaccard similarity scores based on n-grams are computed for each video and for each scene, using these n-grams of variable lengths. As we will see later, the Jaccard score plays the role of normalizing for length. The scene assigned to each media file will be the one with the highest n-gram Jaccard similarity.

3.3 Facial recognition

Facial recognition is the problem of finding faces in images and, in some applications, identifying them. For AutoTag, the most relevant metrics in facial recognition is to find the faces and speed.

Single shot detection (SSD) is currently the best performing model available. In Jang, Gunes, and Patras [6]'s experiments, a SSD model performs at 39.11 frames per second (FPS), which is significantly faster than the recorded frames-per-second in the video files. In an experiment conducted by Yang, Xiong, Loy, and Tang [23], however, a SSD model achieved a runtime of 110 milliseconds per frame, which translates to only 9.09 FPS. Nonetheless, it is faster than other models, such as FastRCNN's 140 ms (7.14 FPS), HR's 1600 ms (0.625 FPS), ScaleFace's 270 ms (3.70 FPS), and ScaleFace-Fast's 160 ms (6.25 FPS).

In addition, SSDs manage to achieve high accuracy even for difficult shots. Wang, Xu, Li, and Sun [22] test SSDs on images with a shallow depth of field or with motion blur. They explore methods through which to enhance the performance of SSDs and achieve near-state-of-the-art results, with a mean average precision of 0.852. Another possible approach proposed by Tang, Du, He, and Liu [18] involves context-based detection. This is particularly helpful when the faces to be detected are blurry or too small, and uses other elements within an image to identify the potential presence of a face. This model yielded a precision of 0.887 on their dataset.

Based primarily on speed considerations, AutoTag uses a ResNet-based single shot detector model developed by Adrian Rosebrock [15]. Our results indicate a range from 35.2 to 38.9 FPS with 0.743 mean average precision.

3.4 Shot identification

Few papers have dealt with identifying the types of shots in film. Most of them deal with reconstructing the geography of the scene from the image, considering aspects like depth and distance from the camera. AutoTag seeks only to identify the shot type.

Canini, Benini, and Leonardi[4] used the geometric composition of a shot and its motion distribution to build a support vector machine (SVM) classifier for shot types (e.g., close-up, medium, long etc as in Fig. 2). Their system yielded an accuracy of up to 0.89 on identifying medium shots in natural settings, and as low as 0.80 when identifying natural long shots.

Tsingalis, Vretos, Nikolaidis, and Pitas [20] propose an alternative approach to classifying shots. Their model proposes grabbing single frames from a video and extracting the locations of faces with respect to the rest of the frame. Using a support vector machine, this approach yields an average 0.99 accuracy when extrapolating multiple single frame classifications to the entire video file. This model works under the assumption that there are faces to be found within the frame, which, although very common, is not always the case.

A third approach, from Hasan, Xu, He, and Xu [5], proposes the use of motion detection to classify shots. Instead of using the traditional classifications, they focus on camera motion descriptors, such as static, pan, zoom, and tilt. Their model comes up with an average precision of 0.95 in these classifications. Despite this, their approach might have limited application to film, because their dataset, and subsequently, their model, does not account for more complex movements, like those produced by dollies, steadicams, and cranes.

AutoTag builds on the shot classification work of Tsingalis, Vretos, Nikolaidis, and Pitas [20] which is based on support vector machines. Our method differs slightly in that we

use k-means clustering. Specifically, after identifying the faces in a million frames, AutoTag runs k-means clustering with five centroids, to classify shots into one of the five shot types.

4 Related work

Several tools that automate certain metadata and files management tasks in Adobe Premiere Pro have recently appeared:

- PluralEyes[1], by Red Giant, allows for automated synchronization of audio and video files without the use of timecodes within Premiere Pro. As a tool exclusively focused on audio synchronization, PluralEyes lacks AutoTag's ability to match audio files to the screenplay. This matching is particularly useful in cases where multiple audio and video files contain recordings of the same scene. AutoTag's benefits do however require a timecode device at production time.
- Digital Anarchy's PowerSearch https://digitalanarchy.com/premiere-search/powersearch.html provides an alternative metadata searching tool to Premiere Pro's standard search workflow, that allows for searches in all forms of metadata at once. This includes user-generated metadata, markers, and other file metadata such as framerate, aspect ratio, video dimensions, date of creation, etc. AutoTag is complementary to and compatible with PowerSearch, because AutoTag generates new metadata tags (e.g., shot type and transcriptions as markers), which could then be used by PowerSearch when searching across the entire project.
- Transcriptive [19], also developed by Digital Anarchy, is a tool that integrates with Premiere Pro and provides transcriptions of media found within a project. This overlaps with AutoTag's functionality of transcribing video and audio. However, it does not include AutoTag's ability to match these transcriptions with a screenplay, or to leverage a screenplay for corrections after transcribing. On the other hand, it offers transcript-based editing, which could be integrated with AutoTag transcriptions to create quick "rough" edits. Thus, it is compatible and complementary.
- Starting in its 22.2 version, Adobe Premiere Pro now offers a transcription tool that generates transcriptions natively [17]. It provides automatic caption generation and speaker identification, in 14 languages. This tool does not include support for matching transcriptions and screenplay (as AutoTag does), or transcript-based editing (as Transcriptive does). However, this tool and AutoTag could be used in a complementary and compatible manner.
- Pro Media Tools [14], by Digital Rebellion, is a media and workflow management tool external to non-linear editing software. It allows for easy management of bins and markers, as well as providing an easy framework for timecode matching. AutoTag provides similar tools, but AutoTag is capable of generating its own metadata based on characteristics of the media itself or found in the screenplay. As with other tools in this list, AutoTag's metadata is easily accessible through Premiere Pro's metadata management tools, including third-party tools like Pro Media Tools.
- Pymiere [11] provides integration of Premiere Pro with Python code. This enables the user to manipulate metadata and manage files using Python (instead of ExtendScript) as a programming language. This tool is complementary to AutoTag for users who prefer Python.

In summary, these tools are for the most part complementary to and compatible with AutoTag, usually in the sense that AutoTag can provide data that those tools can use or simply that those tools offer functionality that AutoTag does not (non-LTC audio synchronization, transcript-based editing, and Python support). In addition, there is no other tool that integrates shot classification to the Premiere Pro pipeline. AutoTag is also free.

5 AutoTag workflow and algorithms

Users will benefit most from using the AutoTag toolset as filming takes place. At the end of each day of production, the day's footage can be imported into Adobe Premiere Pro and have its metadata tagged right away. The time to do this is proportional only to the length of that day's footage.

The media is imported into a Premiere project. Then, within Adobe Premiere Pro, AutoTag will execute the following steps.

1. The linear timecode (LTC) intervals represented in each piece of media are translated into readable timecode. This helps to link various files to the same scene. For example, if it is unclear after transcription which scene corresponds to a given video, the video's corresponding audio file(s) can be used to correctly identify the scene. The conversion from sound waves to readable timecodes is performed using the LTCTools framework, which uses Fast Fourier Transforms to decode the corresponding frequencies. Figure 4 shows a video file that overlaps with two audio files, and therefore, are contained inside the same scene.
2. Generate a transcript from each piece of media. This serves two purposes: (i) to match the media to the corresponding scene in the screenplay, and/or (ii) to be able to associate the text with the time markers in that media piece. As mentioned above, AutoTag uses the Google Cloud Speech-to-Text API, which employs a proprietary (to Google) deep learning model.
3. Run all video footage on the shot identification model, which uses the location of faces within a shot to classify the video in up to two out of five relevant categories,

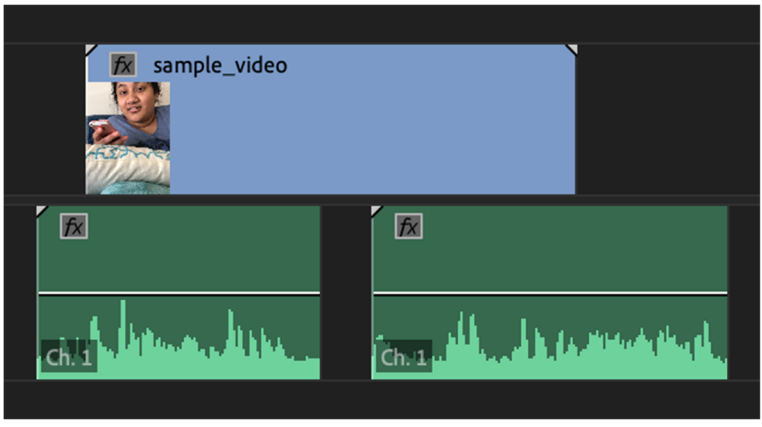

Fig. 4 Two audio files (green) and a video file (blue) in an Adobe Premiere Pro timeline. The contents of these files were recorded at the same time, making them part of the same scene

close-up shot, **medium close-up shot**, **medium shot**, **American shot**, and **long shot** Fig. 1. Facial recognition is performed by a ResNet-based Single Shot Detector provided by Adrian Rosebrock[15] and originally developed in Caffe[7]. (An alternative approach by (AUTHOR) would also have been possible. In our hands it gave the same performance as Rosebrock's model.) The k-means model used to identify the different types of shots in media was trained using frames from commercial feature films from several different countries and cinematographers, totalling approximately one million individual frames.

4. After these are finished, AutoTag writes the relevant metadata information into the Premiere project, in the form of metadata tags or media markers.

The following two subsections describe the algorithms used to associate audio files to scenes and to identify shot types, thus elaborating the brief discussions from the related work section.

5.1 Associating audio files with scenes in a screenplay

When a screenplay is present (typically for fiction), the dialogue is extracted from the text and compared to the transcripts obtained in the speech-to-text step of AutoTag in order to find the most likely scene a media item belongs to. To extract the dialogue from the industry standard software Final Draft, AutoTag makes use of the fact that Final Draft creates files with extension .fdx. Its structure is similar to that of a .xml file, which makes parsing for specific sections of a screenplay simple. Fortunately, many other common formats for screenplay writing can be converted to Final Draft files, including Celtx and Fountain files.

5.1.1 N-Gram Jaccard similarity search

AutoTag parses each scene in the fdx files to create a single string containing all of the words tagged as dialogue, separated by spaces. Next, AutoTag removes punctuation and stems the remaining words and removes stop words using the NLTK library in Python [10]. The result are collections of n-grams of lengths 1 through 5 for each scene in the screenplay.

AutoTag takes the output of the Speech-to-Text module and processes them in the same way: they are stripped of special characters and stop words, stemmed, and tokenized. This process enhances the robustness of scene identifications even when actors make mistakes or go off script. (Even in professional settings, directors and actors may change the wording without making changes to the text of the screenplay.) AutoTag takes the stemmed and tokenized string and constructs n-grams as above.

Specifically, for each transcript i, corresponding to an audio or video file, AutoTag first creates a set of n-grams T_i for n = 1 to 5. Next, AutoTag evaluates the Jaccard score between T_i and the n-grams S_j of each scene j. The j whose Jaccard score is greatest is then associated with i. The Jaccard score between T_i and S_j is computed as usual as follows:

$$J(T_i, S_j) = \frac{|T_i \cap S_j|}{|T_i \cup S_j|} \tag{1}$$

Both the numerator and the denominator grow in proportion to the number of n-grams, so the scores tend to be comparable whether a scene is long or short.

The average duration of a scene is 3 minutes, but it can be arbitrarily long. This implies that, even in long movies, the number of scenes is limited to under 100. In an experiment on a single film having 100 scenes, each comprising roughly 2.5 pages of script, matching the

transcript of all 100 files (once the transcript is found) with their corresponding scenes took 10 seconds. In practice therefore, the matching of transcript to scene takes negligible time.

Input: transcript extracted from file T_i, list of n-grams for each scene in a screenplay S
Result: scene number
1. jaccardScores $\leftarrow$ array of zeros of length equal to length(S)
2. **for all** list of n-grams $S_j \in S$ **do**
3. jaccardScores[j] $\leftarrow J(T_i, S_j)$
4. **end for**
5. **return** argmax_j(jaccardScores)

Algorithm 1 Get Scene from Transcript This algorithm calculates Jaccard similarity scores between recorded shots and screenplay scenes. Then it takes the scene that has the highest such score.

5.2 Identification of shot types

Shot type depends on the relative size of the largest face in a frame. So the key problem is to find faces.

Adrian Rosebrock from PyImageSearch developed an open source Single Shot Detector for face identification [15]. AutoTag runs Rosebrock's face identification for every frame of a single video sequence. If there are multiple faces identified in the frame, AutoTag selects the one with the largest surface area. For that face, AutoTag considers four different characteristics: the percentage area of the frame it occupies, the percentage width of the frame it occupies, the percentage height of the frame it occupies, and the position of the center of the face where the bottom right corner is (1, 1) and the top left corner is (0, 0). In the encoding of the center, x is relative to the height of the frame and y is relative to the width of the frame. See Fig. 5 for an example frame where these features have been extracted. As we explain below, we use a five-way clustering approach (for the five shot types) in order to characterize each shot type.

Fig. 5 An example still from the movie *Breakfast at Tiffany's*, where a face was identified. It occupies 0.019 of the area, 0.1125 of the width, 0.176 of the height, and its center is located at the point (0.289, 0.5125)

5.2.1 Training data for shot types

Because there is no dataset of cinematographic shots, AutoTag uses an unsupervised approach to characterize shot types. We extracted one million frames from relevant commercial films with diverse cinematographic styles, from four different countries and several time periods, summarized in Table 1. For each movie, we extract the frames where our SSD was able to identify at least one face. Our dataset is available at this web address https://bit.ly/3CqrKWI.

The movies came mostly from the United States, but also includes movies from India, South Korea, and the United Kingdom. Although they are mostly fiction, there is not necessarily a cinematographic difference, at least in terms of shot styles, between fiction and documentaries.

The dataset also reveals a potential weakness in the Rosebrock Single Shot Detection model. The model extracts fewer frames in cases where the cast consists of people of with dark complexions. This is especially true in the case of movies whose cast primarily consists of black actors, as is the case, for example, in *12 Years a Slave* (2013) and *Get Out* (2017). This seems to be less the case when considering other people of color: *3 Idiots* (2009) is one of the movies from which the highest number of frames was extracted.

5.2.2 Shot classification by clustering

Because there are five primary shot types in filmmaking, we used the k-means clustering method with k set to 5 from the Python library Scikit-learn[13].

From each frame, we extracted the faces and the values corresponding to the characteristics mentioned before: the percentage area of the frame it occupies, the percentage width of the frame it occupies, the percentage height of the frame it occupies, and the position of the center of the face. We next translated the values of each feature to z-score. That is, for each feature f and each value v, we transformed v to $(v - mean(f))/std(f)$ where $mean(f)$ is the mean of all values of feature f and $std(f)$ is the standard deviation of the values for feature f. This is done across all frames from all films included in the dataset described in Section 5.2.1.

Table 1 Movies used for the dataset, including their country of origin, cinematographer in charge, and the number of frames used in the dataset

Movie	Country	Cinematographer	Frames
12 Years a Slave	USA	Sean Bobbitt	59751
3 Idiots	India	C.K. Muraleedharan	150128
Boys Don't Cry	USA	Jim Denault	68908
Breakfast at Tiffany's	USA	Franz Planer	104052
Get Out	USA	Toby Oliver	68311
Memories of Murder	South Korea	Kim Hyeong-gu	93309
Oldboy	South Korea	Chung-hoon Chung	62574
Sholay	India	Dwarka Divecha	119479
The Big Sick	USA	Brian Burgoyne	100715
The Shawshank Redemption	USA	Roger Deakins	112402
The Thin Blue Line	USA	Stefan Czapsky	88343
Trainspotting	UK	Brian Tufano	53983

When the k-means clustering completed for the five centroids, we manually associated a shot type with each centroid. When processing a new scene, AutoTag associates each frame with its closest centroid. The features associated with each centroid can be found in Table 2.

Because our classifier for shot types is meant to characterize a video rather than a single frame, AutoTag averages over the frames. Every frame in the video is classified based on which centroid the largest face in the frame is closest to. Each shot label is mapped into a value in ordinal fashion: long is assigned to a 0, American to a 1, medium to a 2, medium close-up to a 3, and close-up to a 4. AutoTag computes an average value from the classifications on all the frames. If the value v of this average is within 0.1 of a specific classification value c, then c is assigned to the video file. Otherwise, if v is between two values c and $c + 1$, then AutoTag assigns both classifications to the video file.

The averaging tends to eliminate outliers. For example, in many of the videos, there are several frames having different shot types. In Fig. 6, we show the relationship between the number of videos and the number of shot types in that video. The x-axis represents the number of shot types and the height of the ith histogram is the number of videos that contain frames with i distinct shot types.

6 Integration with Adobe Premiere Pro

To make AutoTag useful and practical for filmmakers, we have integrated AutoTag with Adobe Premiere Pro. Media files have to first be imported into a project, from which

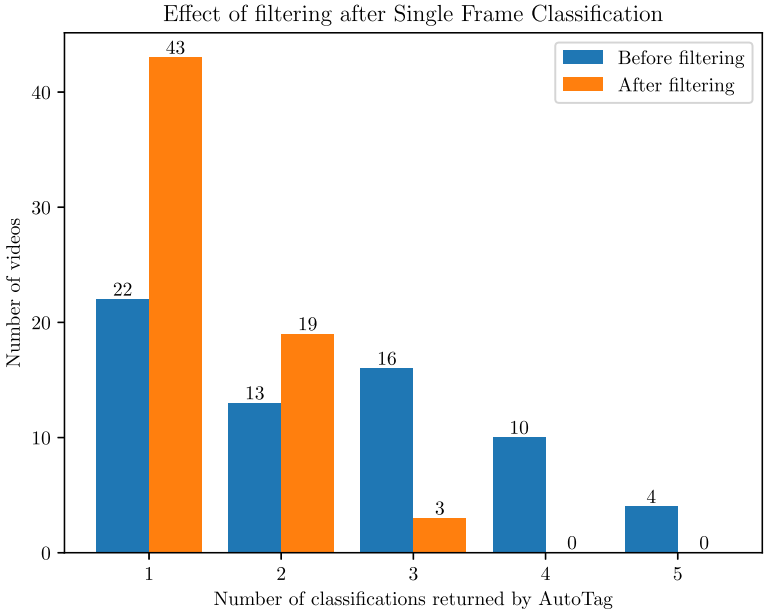

Fig. 6 (i) The blue bar plots indicate the number of shot types per video returned by AutoTag before eliminating shot types that occur in fewer than 20% of the frames. The plurality of videos have just one shot type. (ii) The orange bar plots indicate the number of shot types per video after the elimination. The elimination step reduces the number of shot types per video to one or two. Videos average about one minute in length. The videos come from the 65 videos from the case study of Section 7.5 where at least one shot type was identifiable

Input video file F, array of centroids in ascending order of size of face C
Result classification
1. classCounts ← array of 0s of length equal to length(C)
2. **for all** frame $F_i \in F$ **do**
3. C_i ← classifyByDistance(F_i, C)
4. classCounts[C_i] ← classCounts[C_i] + 1
5. **end for**
6. minFrameCount ← length(F) ∗ 0.2
7. **for all** classification counts $c_j \in$ classCounts **do**
8. **If** $c_j <$ minFrameCount **Then**
9. $c_j \leftarrow 0$
10. **end if**
11. **end for**
12. meanClass ← mean(classCounts)
13. **if** $(\lfloor(meanClass)\rfloor + 0.1) <$ meanClass $< (\lceil(meanClass)\rceil - 0.1)$ **then**
14. **return** ⌊meanClass⌋, ⌈meanClass⌉
15. **else**
16. **return** round(meanClass)
17. **end if**

Algorithm 2 Shot Classification The Shot Classification algorithm proceeds frame by frame based on a distance calculation. It drops classifications that appear infrequently and takes the average of the remaining classifications. If that average lies within 0.1 of a single classification (i.e., an integer), the classifier assigns that classification to the shot. Otherwise, the classifier assigns the neighboring classifications to the shot.

Premiere Pro creates their corresponding media items and bins. Next, AutoTag runs, usually for several hours. The result is a set of tagged bins within Adobe Premiere Pro.

6.1 Task automation in Adobe's ExtendScript

To integrate with Adobe Premiere Pro, we make use of Adobe's scripting tool, called ExtendScript. ExtendScript is based on ECMAScript 3, which is closely related to JavaScript. We use ExtendScript to extract the metadata from the media files related to a project and put them into a JSON file called `<PREMIERE PROJECT FILE>.json`, provide calculations for estimated time of completion for each task, and execute AutoTag's main processes from within Adobe Premiere Pro. If filmmakers use AutoTag incrementally, say by putting in each day's footage at the end of shooting, then AutoTag simply appends the metadata of the new media files into `<PREMIERE PROJECT FILE>.json`.

Table 2 Labels and their respective resulting centroid values from training, after being transformed back into their real values

Label	% Area	% Width	% Height	Center x	Center y
Long	0.025	0.095	0.248	0.341	0.296
American	0.027	0.098	0.256	0.499	0.305
Medium	0.032	0.106	0.373	0.506	0.396
Medium Close	0.086	0.179	0.472	0.514	0.463
Close-Up	0.201	0.277	0.715	0.650	0.560

6.2 User interface

Adobe Premiere Pro provides easy integration of HTML and CSS user interface development for its plugins. The user interface can be seen in Fig. 7. It includes the following:

- A drop-down list to choose the language(s) of the project, for when screenplays are used and/or transcriptions are needed.
- An option to use linear timecode (when a timecode device has been used in production, audio recordings inside the video file is unrecognizable noise).
- An option to have AutoTag classify shots in the project, from close-up to long.
- An option to have AutoTag generate a transcript of the media files. Options that are dependent on obtaining transcripts include:
  - Upload a screenplay.
  - Associate the transcript from each audio recording and each video to a scene in the screenplay.
  - Generate transcript markers every 10 seconds containing the text in the transcript.

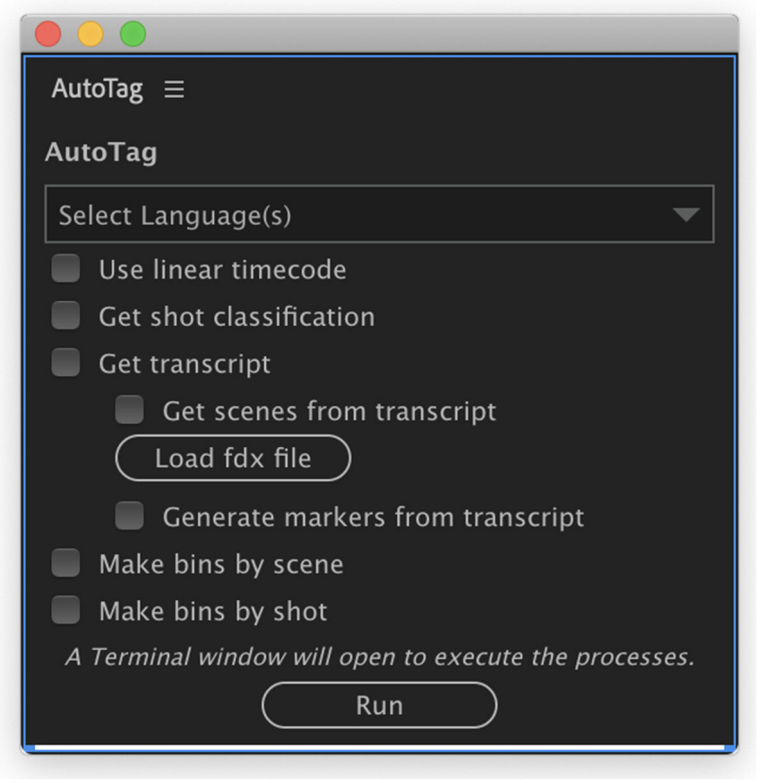

Fig. 7 The user interface for AutoTag inside a Premiere Pro panel. The general options available are to choose a language to be used, use linear timecode to identify audio and video files that correspond to one another, get shot classification, generate a transcript from each media file, create 10-second searchable markers, and create smart bins by shot. When a screenplay is provided, AutoTag also includes options to identify scenes for each media file and to create smart bins by scene

- Make bins by scene, which creates a smart bin for each scene with references to the files from that scene.
- Make bins by shot, which creates a smart bin for each shot type with references to the files using that shot type.
- Run button, when pushed, AutoTag will execute the entire process based on the above options.

In addition, AutoTag includes a tool called AutoTag Search (Fig. 8), which provides an interface to load footage onto the Source Monitor based on AutoTag metadata searches. It includes the following conjunctive options for its search:

- Choose files from some scene (e.g. scene 3).
- Restrict based on shot classification (e.g. close-ups, long shots).
- Restrict to files that contain a string within its transcript.

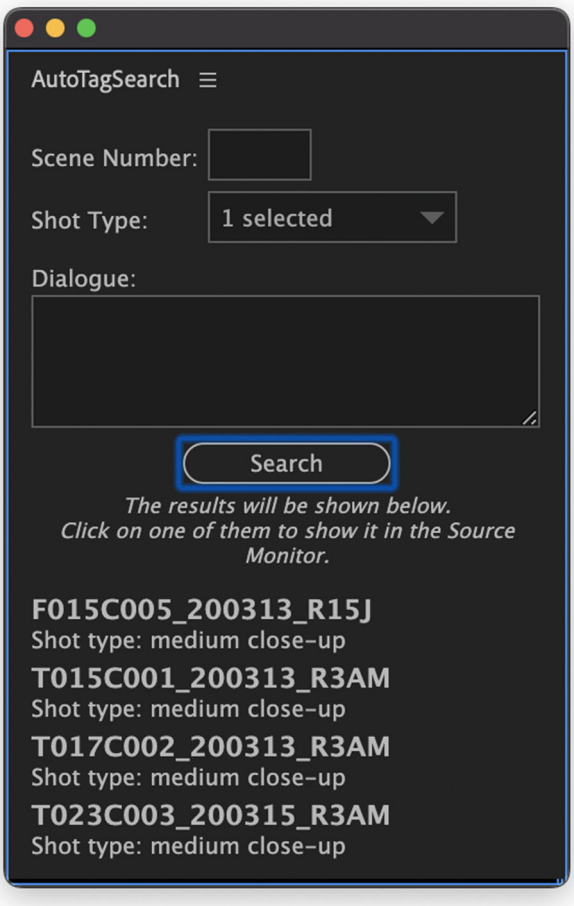

Fig. 8 The user interface for AutoTag Search inside a Premiere Pro panel. Each query returns a list of media shots based on AutoTag's scene number (3 in this case) and shot classification (medium close-up), along with a string search on the transcript ("Look out!")

6.3 Generated tags and markers

Because the generated tags are meant to be searchable and not obstruct the editor's workflow, AutoTag puts them into the 'Log Note' subsection of the metadata panel in Adobe Premiere Pro. The generated tags follow the structure `AutoTag_<tag type>_<tag value>`, with six padding zeroes for the scene number. They are also separated by semicolons, which allows for searches to be more specific. The markers generated in the media items are all named `AutoTag_Transcript` and contain the searchable transcription as a description of the marker.

7 Experiments

7.1 Experimental setting

Sections 7.2 and 7.3 detail the process and results of two experiments we conducted to measure AutoTag's ability to classify cinematographic shots. We asked five filmmakers to independently classify stills and videos extracted from films not in our training dataset into one of the five shot types of our model. After this, these stills and videos were run through AutoTag to generate their classifications. We then compared AutoTag's classifications to those of the filmmakers. The metrics used for comparison are the following: rate of exact agreement,

$$A_{exact} = \frac{\sum_n [c_n = \hat{c}_n]}{N} \tag{2}$$

and rate of within-one agreement,

$$A_{w1} = \frac{\sum_n [|c_n - \hat{c}_n| \leq 1]}{N} \tag{3}$$

where c_n is the shot type determined by one filmmaker for still or video n, $\hat{c}_n$ is the AutoTag label for the same still or video, and N is the total number of media used in the experiment.

7.2 Shot classification on stills

For this experiment, we ran AutoTag's shot classification on 200 stills selected at random from 5 movies not in our training dataset: *Amores Perros* (2000), *Bringing Up Baby* (1938), *Into the Wild* (2007), *The Assassination of Jesse James by the Coward Robert Ford* (2007), and *The Godfather* (1972). We also asked five filmmakers to classify these stills into one of the cinematographic shot types that we identified (long, American, medium, medium close-up, and close-up).

Out of these 200 stills, our model proposed a classification that matched at least one of the filmmakers in 159 of them, yielding an accuracy of 79.5%. Out of the remaining 41 stills that were mislabeled, AutoTag managed to predict a classification within one step of the filmmakers' options in 32 of them, representing 16% of the data. We define one step as one position in the ordering of cinematographic shots, from long shot to close-up shot. Our model mislabeled the remaining 9 stills beyond this acceptable range (e.g. classifying a medium close-up shot as a long shot).

Table 3 shows the rate of exact agreement between filmmakers and AutoTag. Filmmakers among themselves agree within a range of 0.335 and 0.65; AutoTag agrees with each filmmaker within a range of 0.365 and 0.565, well within the margins of the agreement found for filmmakers.

Table 3 Rate of exact agreement (two subjects agree on a single shot type, out of five) between subjects (S1 to S5) and AutoTag (AT) for stills

	S1	S2	S3	S4	S5	AT
S1	1					
S2	0.65	1				
S3	0.51	0.515	1			
S4	0.53	0.645	0.49	1		
S5	0.465	0.435	0.335	0.65	1	
AT	0.44	0.53	0.485	0.565	0.365	1

The human subjects agree with one another between 33.5% and 65% of the time. AutoTag agrees with the humans between 36.5% and 56.5% of the time

Table 4 shows the rate of agreement within one step of a classification. In this case, the rate of agreement for filmmakers among themselves is between 0.905 and 0.99. The rate of agreement between AutoTag and the filmmakers ranges between 0.85 and 0.88.

Overall, AutoTag agrees with other filmmakers roughly as much as they do with one another, though slightly less in the one step experiment.

7.3 Shot classification on video

We repeated the process in Section 7.2 with 20 clips selected at random from two movies, also not in our training dataset: *Still Alice* (2014) and *T2 Trainspotting* (2017).

Out of 20 videos, AutoTag output a classification that matched at least one of the filmmaker's classifications in 16 of them, yielding an accuracy of 80%. AutoTag matched classifications within one step in 3 other videos, representing 15% of the data.

Table 5 shows the rate of exact agreement between filmmakers and AutoTag. Filmmakers among themselves agree within a range of 0.35 and 0.75; AutoTag agrees with each filmmaker within a range of 0.5 and 0.65. AutoTag is still within the range of agreement rate among filmmakers.

Table 4 Rate of agreement within one step (two subjects agree within one step in the shot type ordering) between subjects (S1 to S5) and AutoTag (AT) for stills

	S1	S2	S3	S4	S5	AT
S1	1					
S2	0.99	1				
S3	0.905	0.97	1			
S4	0.96	0.985	0.96	1		
S5	0.91	0.99	0.935	0.99	1	
AT	0.865	0.875	0.85	0.87	0.88	1

The human subjects agree with one another between 90% and 99% of the time. AutoTag agrees with the humans around 86% of the time

Table 5 Rate of exact agreement (two subjects agree on a single shot type, out of five) between subjects (S1 to S5) and AutoTag (AT) for video clips

	S1	S2	S3	S4	S5	AT
S1	1					
S2	0.6	1				
S3	0.45	0.65	1			
S4	0.4	0.6	0.35	1		
S5	0.65	0.75	0.6	0.55	1	
AT	0.5	0.65	0.5	0.55	0.65	1

The human subjects agree with one another between 35% and 75% of the time. AutoTag agrees with the humans between 50% and 65% of the time

Table 6 shows the rate of agreement within one step of a classification. In this case, the rate of agreement for filmmakers among themselves is between 0.95 and 1. The rate of agreement between AutoTag and the filmmakers ranges between 0.85 and 0.9.

The findings in this experiment are consistent with the findings in the experiment with stills. Nonetheless, it performs better using the exact agreement metric in videos than in stills, likely because it is able to smooth out outlier frame classifications.

7.4 Case study: *challenging erasure*

The documentary short film is called *Challenging Erasure* by Katarina Holtzapple. The raw footage consists of 50 different videos, consisting mostly of interviews as medium shots, found footage from other media sources, and a variety of other shots to be used as a B-roll (e.g., footage to associate images with the words that the interviewee is speaking). It includes videos shot at different aspect ratios, such as 16:9, 21:9, and 9:19.5, which come from a Blackmagic Pocket 4K and from an iPhone X shooting in portrait mode. Its audio consists mostly of interviews recorded in a H5 Zoom recorder working on one channel. The total audio to be processed lasts 8.23 hours long, including video and audio, and the total video to be processed is 5.11 hours long. The goal was to compress this into a cut of roughly 10 minutes which gives a shooting ratio of 80.7:1, considering both audio and video.

Table 6 Rate of agreement within one step (two subjects agree within one step in the shot type ordering) between subjects (S1 to S5) and AutoTag (AT) for video clips

	S1	S2	S3	S4	S5	AT
S1	1					
S2	1	1				
S3	0.95	0.95	1			
S4	1	1	0.95	1		
S5	1	1	0.95	1	1	
AT	0.9	0.85	0.85	0.85	0.9	1

The human subjects agree with one another between 95% and 100% of the time. AutoTag agrees with the humans between 85% and 90% of the time

The median time taken to process these audio contents is 3.21 seconds for every 10 seconds subclip of audio. Only 5% of the subclips took more than 6.05 seconds to process, with 6.29 seconds being the maximum. This suggests that AutoTag provides a workable transcript of the audio at 62.9% of the duration of the audio files at worst, and about 32% of the duration of the audio files on average. This processing cost $11.86 USD in total, with the Google Cloud rate of $0.006 USD for every 15 seconds transcribed.

As for the shot classification, it took 3.63 hours to process 5.11 hours of video. This represents 71.03% of the duration of the video, achieving a speed of 35.2 frames per second. Only one video in the raw footage did not contain a face, and thus could not be classified by our model. The shot classifications were 91.8% accurate, failing to identify faces in 4.01% of the files and misclassifying 4.01% of the files. The files where faces could not be identified consisted of medium and long shots in which the faces were dark. As for the misclassifications, they occurred in an American shot, identified by the model as a long shot, and a long shot, identified as a medium shot. A large percentage of the footage consisted of medium shots, medium close-ups, and close-ups, making up 69.4% of all shots. Out of these, only one medium shot was misidentified as containing no faces. Misclassifications in this project seem to occur only for faces that are far away from the camera.

7.4.1 Testimonial from the director

The director, Katarina Holtzapple, stated that "this tool is particularly useful for the work I do, as I do not have a clear shotlist or someone on set that takes notes of all of these things, because so much of my work is found footage and shots taken for over a year and that can get very disorganized". For this particular project, she estimated that about 25% of her time during post-production was spent looking through footage and extracting the information that AutoTag makes readily accessible and searchable.

7.5 Case study: *how we leave*

Our second case study is a fiction film directed by Liene Magdalēna called *How We Leave*. This project is composed primarily of videos shot in 16:9 aspect ratio, which have been linked to their audio counterparts through a linear timecode device. There are, in total, 85 video files and 196 audio files. The total duration of the media is 9.76 hours for its expected runtime of 10 minutes, thus giving a shooting ratio of 58.6:1. This project is a particularly interesting challenge for AutoTag, because some files from other projects got mixed up in the process. In this case, AutoTag was used partly to identify which media files do not belong to any scene, and therefore, do not belong in this project.

In this case, the median time taken to process the audio contents was 4.05 seconds for every 10 seconds of audio. However, 5% of the subclips took more than 13.208 seconds to be processed, with 14.64 being the maximum. The high processing times all come from the other projects that got mixed in. These files consist mostly of interviews performed both in English and a second unidentified language, and are not part of *How We Leave*. Although this skews our performance statistics, it can be seen that the model processing time was still only 40.5% of the duration of the clips on average.

21 out of those 24 files were identified as outside the scope of the *How We Leave* screenplay, due to their having zero n-gram Jaccard similarity to any scene within the screenplay, which results in a 87.5% accuracy when identifying unrelated media within a Premiere project. The three files that were identified as part of the screenplay were identified as belonging to scene 4, which is the longest and most dialogue-heavy one in the screenplay.

Out of the remaining 172 files that do belong to *How We Leave*, 79 contained dialogue and belong in a scene in the screenplay. Of these 91.1% were classified correctly. Misclassification occurred between scenes 2 and 5, where some of the dialogue overlaps. AutoTag had trouble with the files from *How We Leave* that contained no dialogue, giving them a zero score and therefore asserting that they came from another movie.

Shot classification took 4.23 hours to process 6.07 hours of video. This represents 69.7% of the duration of the video, achieving a speed of 35.87 FPS. This is consistent with our findings in the first case study. In this project, our shot classification model achieved an accuracy of 88.2%. The misclassifications are largely related to the lighting of each shot, as the face recognition model seems to rely heavily on well-exposed footage. At the same time, another weakness is revealed in the face recognition model, which is that it cannot handle extremely long shots because it cannot detect some smaller faces. These all were tagged as containing no identifiable faces, when they in fact did. Another interesting characteristic of the misclassifications is that, when the faces were identified properly, and thus, a shot was assigned, it is only one step off at most, and it always overestimated the distance from the camera. One likely cause for this is that the actors of this movie were children, and thus, have smaller faces than adults.

7.5.1 Testimonial from the director

Liene Magdalēna described the many technical difficulties she and her team had to go through during post-production. Particularly, having to identify which audio belonged to which video by hand, and then not being able to keep track of these in an organized manner afterwards. There were other issues, such as the handling of linear timecode, the naming of files, and sharing information across editors. The director said that she had to go through each piece of footage by hand to identify scenes, shots, and whether a file belonged or not in the project. The post-production took about five months, much of which involved keeping track of unorganized metadata.

Regarding AutoTag, she stated that "a tool like this allows us filmmakers to focus more on the art involved in post-production, not having to worry much about the more technical and repetitive tasks, like keeping track of metadata and linear timecode".

7.6 Case study: *reminiscencia*

The third case study is a fiction film directed by Renato Corpancho called *Reminiscencia*. It is still being edited at the time of this writing. The editor is using AutoTag during the initial steps of post-production. The shots were taken using a Canon C300 Mark III camera. The project consists of 288 video files and 342 audio files. The total duration of media is 19.87 hours.

AutoTag worked on the ProRes proxy files which are smaller and lower resolution than the original files. This case study did not consider transcripts because this movie only loosely followed its screenplay.

Shot classification took 12.77 hours for 19.87 hours of video, which translates to a speed of 38.89 FPS. This number is slightly higher than in other case studies, likely due to the use of the lower resolution proxy files. This project contained 41 files that did not contain any faces, and thus, could not be classified by AutoTag. Out of the remaining 247 video files, 222 files were correctly classified with their corresponding shots, equivalent to a 89.8% accuracy, which is consistent with our other case studies. The files most

prone to misclassifications were videos that involved multiple faces in an underexposed environment.

7.6.1 Testimonial from the director

The director, Renato Corpancho, said "I work very closely with my editor, and we often find ourselves going through the files looking for the exact shot I have in mind for a sequence in the film. AutoTag makes that search process a lot faster".

7.6.2 Testimonial from the editor

The editor for *Reminiscencia* is Carolina Lominiczi. She mentioned that the most tedious step of the editing process is the organization of files inside the software, especially in a project without a screenplay. In her work, "organizing the files to the degree provided by AutoTag regularly takes several weeks for a project without a screenplay, but AutoTag does it overnight." She also commented on finding misclassifications, where she stated that "they are annoying, but the trade-off is worth it, becuase there are not that many of them".

7.7 Case study: *happy holidays*

The fourth case study is a feature-length fiction film directed by Academy-award nominee Scandar Copti called *Happy Holidays*. The camera used is an Arri ALEXA Mini, shot in Raw, and with two resolutions: 4K and 2.7K. For editing, the original files were converted into ProRes proxy files. He ran AutoTag on a small sample of the proxy footage, specifically nine video files that belonged to the same scene. Unfortunately, not much footage is available due to the COVID-19 pandemic, as production of this film has been temporarily halted.

In these files, AutoTag was 100% accurate with the director's choices for shot types. However, this project has some specific limitations: *Happy Holidays* was filmed using long-lasting shots (several minutes). These typically included movement of the camera or individuals from one shot type to another. AutoTag's shot classifier correctly identified the one or two shot types.

7.7.1 Testimonial from the director

Mr. Copti described his process as follows: "the way we usually do this classification of media is through memory, logging, or having an assistant do it. I can easily spend several weeks working on this for each project. AutoTag saves me all of that time, money, and effort. When editing a sequence, you use the technical aspects of a file and the writing to convey an emotion. Cutting from a wide shot to a close-up provides a specific feeling, so you need to have all of your shots properly classified. AutoTag can do this for you".

Regarding search, he said that he finds it most useful to search by combinations of dialogue and scene. "Just like with Mac's Spotlight, there is no need to thoroughly organize my media if I can just search for it this way. That would save a lot of time in the editing process."

He added that he would like to see AutoTag work on Avid Media Composer as well, as he does much of his work on that software.

8 Conclusions and future work

AutoTag automates the time-consuming task of assigning metadata to raw footage. Specifically,

- For every clip, AutoTag will infer the shot type (from close-up to long) and the scene numbers by reference to the screenplay.
- It will put this information into Adobe Premiere, thus supporting search and assembly.

AutoTag thus replaces hours of human time by an overnight computation. AutoTag has the potential to make film post-production less time consuming and less expensive for all filmmakers. As far as we know, this is the first system to offer such functionality.

Code for this project is available at https://www.github.com/mudtriangle/autotag. Setup instructions are provided.

Future work One improvement to AutoTag is to include character identification through vocal or visual properties. This would allow scene selection (both video and audio) by character.

A second technical improvement is to characterize how the camera is being held. For example, the camera can be mounted on a dolly, be held using a steadicam device, be handheld, or be at a fixed position (static). This is useful to filmmakers who seek aesthetic continuity. For example, they might want to avoid a dolly-held scene followed by a handheld scene.

Acknowledgements The user interface of the Adobe Premiere Pro plugin was designed and developed by María Paula Calderón. Shasha's work has been partly supported by (i) the New York University Abu Dhabi Center for Interacting Urban Networks (CITIES), funded by Tamkeen under the NYUAD Research Institute Award CG001 and by the Swiss Re Institute under the Quantum Cities initiative, (ii) NYU Wireless, and (iii) U.S. National Science Foundation grants 1934388 and 1840761. That support is greatly appreciated.

Data Availability The datasets generated during and/or analysed during the current study are available in the following Google Drive repository, https://bit.ly/3CqrKWI.

Declarations

Conflict of Interests The authors declare that they have no conflict of interest.

References

1. Audio Video Sync Software Red Giant - PluralEyes by Maxon. https://www.maxon.net/en/red-giant/pluraleyes
2. Bansal S, Kamper H, Livescu K, Lopez A, Goldwater S (2018) Low-resource speech-to-text translation. Computing Research Repository
3. Berard A, Pietquin O, Servan C, Besacier L (2016) Listen and translate: a proof of concept for end-to-end speech-to-text translation. Computing Research Repository

4. Canini L, Benini S, Leonardi R (2011) Classifying cinematographic shot types. Multimedia Tools and Applications
5. Hasan MA, Xu M, He X (2014) Xu, C. Camera motion histogram descriptor and its application to cinematographic shot classification. IEEE Transactions on Circuits and Systems for Video Technology, Camhid
6. Jang Y, Gunes H, Patras I (2019) Registration-free face-ssd: single shot analysis of smiles, facial attributes, and affect in the wild Computer Vision and Image Understanding
7. Jia Y, Shelhamer E, Donahue J, Karayev S, Long J, Girshick R, Guadarrama S, Darrell T (2014) Caffe: convolutional architecture for fast feature embedding. Proceedings of the 22nd ACM international conference on Multimedia
8. Kepuska V, Bohouta G (2017) Comparing speech recognition systems (microsoft api, google api and cmu sphinx). International Journal of Engineering Research and Application
9. Khilari P, Bhope VP (2015) A review on speech to text conversion methods. International Journal of Advanced Research in Computer Engineering & Technology
10. Loper E, Bird S (2002) Nltk: the natural language toolkit. Proceedings of the ACL-02 workshop on effective tools and methodologies for teaching natural language processing and computational linguistics - vol 1
11. Masingarbe Q (2019). https://github.com/qmasingarbe/pymiere
12. Neculoiu P, Versteegh M, Rotaru M (2016) Learning text similarity with siamese recurrent networks Proceedings of the 1st Workshop on Representation Learning for NLP
13. Pedregosa F, Varoquaux G, Gramfort A, Michel V, Thirion B, Grisel O, Blondel M, Prettenhofer P, Weiss R, Dubourg V, Vanderplas J, Passos A, Cournapeau D, Brucher M, Perrot M, Duchesnay E (2011) Scikit-learn: machine learning in python. Journal of Machine Learning Research
14. Pro Media Tools. https://www.digitalrebellion.com/promedia/
15. Rosebrock A (2018) Face detection with opencv and deep learning. PyImageSearch
16. Shakhovska N, Basystiuk O, Shakhovska K (2019) Development of the speech-to-text chatbot interface based on google api. Modern Machine Learning Technologies
17. Speech to text in premiere pro. https://helpx.adobe.com/premiere-pro/using/speech-to-text.html
18. Tang X, Du DK, He Z, Liu J (2018) Pyramidbox: a context-assisted single shot face detector. European Conference on Computer Vision
19. Transcribe, search and edit in premiere pro. https://transcriptive.com/products/transcriptive-for-premiere-pro/
20. Tsingalis I, Vretos N, Nikolaidis N, Pitas I (2012) Svm-based shot type classification of movie content. IEEE Mediterranean Electrotechnical Conference
21. Ukkonen E (1992) Approximate string-matching with q-grams and maximal matches. Theoretical Computer Science
22. Wang S, Xu T, Li W, Sun H (2017) Cssd: cascade single shot face detector. IEEE International Conference on Image Processing
23. Yang S, Xiong Y, Loy CC, Tang X (2017) Face detection through scale-friendly deep convolutional networks. Computing Research Repository