Intro and Recap Detection for Movies and TV Series

Xiang Hao, Kripa Chettiar, Ben Cheung, Vernon Germano and Raffay Hamid
Amazon Prime Video
{xianghao,ksivakum,cheungb,germanov,raffay}@amazon.com

Abstract

Modern video streaming service companies offer millions of video-titles for its customers. A lot of these titles have repetitive introductory and recap parts in the beginning that customers have to manually skip in order to achieve an uninterrupted viewing experience. To avoid this unnecessary friction, some of the services have recently added "skip-intro" and "skip-recap" buttons to their video players before the intro and recap parts start. To efficiently scale this experience to their entire catalogs, it is important to automate the process of finding the intro and recap portions of titles. In this work, we pose intro and recap detection as a supervised sequence labeling problem and propose a novel end-to-end deep learning framework to this end. Specifically, we use CNNs to extract both visual and audio features from videos, and fuse these features using a B-LSTM in order to capture the various long and short term dependencies among different frame-features over time. Finally, we use a CRF to jointly optimize the sequence labeling for the intro and recap parts of the titles. We present a thorough empirical analysis of our model compared to several other deep learning based architectures and demonstrate the superior performance of our approach.

1. Introduction

The growth of video streaming services has given rise to the practice of *marathon-watching*, where users watch a series of video content in a single extended session [40]. Given the popularity of this behavior, it is important for streaming services to provide interfaces that facilitate watching content in an uninterrupted and frictionless manner.

One source of such friction is the introductory (*intro*) content which is usually played in the beginning of a video and goes over various title-credits *e.g.* information about the actors, director and producers. Besides *intros*, TV episodes can also contain *recap* content that summarizes what happened in the previous episodes in the series. Watching the *intro* and *recap* parts of a title can be repetitive and therefore can result in customer fatigue. If the customers want to skip these content-segments, they have to manually seek the content forward, which creates unnecessary friction.

To minimize this source of friction, some of the video streaming companies have recently added "skip-intro" and "skip-recap" buttons to their video players which appear a few seconds before the *intro* or *recap* parts start. If these buttons are clicked, the player jumps to the end of *intro* or *recap* parts respectively. These features create a better playback experience, increase customer satisfaction, and can have a positive impact on the long-term user engagement.

To support this experience, one needs to know the timestamps where the *intro* and *recap* parts of a video-title start and end such that the "skip-intro" and "skip-recap" buttons could be placed accordingly. Scrubbing titles for finding *intro* and *recap* timestamps is a labor-intensive manual process and requires highly trained human annotators. Therefore efficiently scaling-up this experience to a streaming service's entire catalog with millions of video-titles requires leveraging Machine Learning to maximally automate this process.

Choosing a Practical Learning-Based Approach:
There are several learning based approaches that could be applied to tackle our problem. On the unsupervised end of the spectrum, one could model *intro* and *recap* as different types of frequently repeating content and then try to use content based matching [41] to find them.

Although an advantage of this approach is that it does not require labeled data for training a parametric model, it does pose several practical challenges. First, its false positive rate can be high due to multiple segments of shared content between two or more episodes. While one could apply *ad-hoc* rule-based filtering scheme to remove some of these false positives, such solutions only address a subset of these errors and are difficult to generalize. Second, this approach requires multiple episodes before the content in different episodes can be matched. When there is only one episode available in a TV series, this approach is not applicable. Similarly, this approach cannot be applied to Movies given their singleton nature. Third, this approach does not work well for titles where the *intro* content changes for each episode. Game of Thrones is a popular example of such

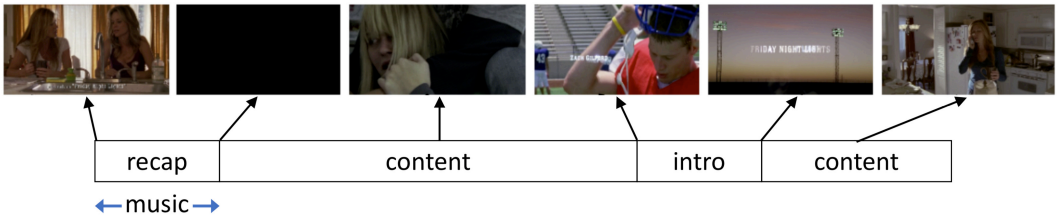

Figure 1: Example of discriminable *intro* and *recap* patterns. Note that there is background music playing in *recap* and title-credit information is displayed in the frames of *intro*.

TV series where the *intro* content changes depending on the story-arch of a particular episode. Similar challenge exists for *recaps* since their constituent shots are not always from the most immediate previous episode, but can come from any episode from the season (*e.g.* in Fox studios show *24*). In this case, it is not clear which episodes should be compared with the current episode to detect *recaps*. One may try to compare to *all* prior episodes, but that poses scalability and maintainability challenges.

An alternative approach that does not pose the aforementioned limitations is to view the problem from a supervised learning perspective. An underlining assumption in supervised methods is that there are some frequently occurring patterns between the features (video content) and the labels (*intro* and *recap* timestamps). The fact that humans can identify whether the content is *intro*, *recap* or the actual content suggests that there are discriminable *intro* and *recap* patterns in the titles. Looking closely at the title data gives some clues regarding the following common patterns:

• For *intros*, as shown in Figure 1, the title-credits information *e.g.* producers, distributors, and director's information is usually displayed on plain (usually black) background with high-contrast (usually white) text. When there is a consecutive series of video frames containing such text, it is a good indicator that these frames are title-credits. In addition, there are usually a few seconds of silent transition period right after the credits where the frames usually have completely dark background and no foreground content during this transition time. Moreover, music usually plays without any speech when credits are showing. Therefore, music signals could be used as indicators for credits.

• For *recaps*, usually a series of rapidly changing scenes are shown, providing a repetitive detectable pattern. Similarly, many *recaps* start with an opening sentence such as "previously on", either in audio or as text on the video frames, along with music playing in the background. In addition, as shown in Figure 1, there are sometimes a few seconds of silent transition period with dark frames right after the *recaps* similar to *intros*.

The presence of these discriminative patterns make supervised model a promising choice for our problem. Furthermore, such supervised models could be applied to a variety of previously mentioned practical scenarios where unsupervised models are not readily applicable. Particularly for cloud-based services, since it is not practical to access clients' entire video-catalog, it is all the more important to use supervised models that can find *intro* and *recap* in each video uploaded by the customers without requiring any additional dependencies. We therefore focus in this work on exploring supervised learning based approaches to build *intro* and *recap* detection model.

Technical Challenges:
There are several key challenges in building a supervised learning based solution for our problem.

• First, it is unclear how to best formulate the *intro* and *recap* detection into a supervised learning problem. For instance, the more obvious choice of formulating it as a regression problem with video as input and *intro* and *recap* timestamps as regression targets is likely to give sub-optimal results since the transition among *intro*, *recap*, and the actual content would not be explicitly modeled.

• Second, while there are some patterns to discern which parts of a title are *intros* and *recaps*, it is unclear what is the best way to characterize these patterns. One way is to build an individual detection model for each of these patterns, *i.e.*, a model to detect white text on black background, while another to detect if music is playing, *etc*. However, to ensure high coverage of *intros* and *recaps*, such an approach requires a large list of known patterns which is time-consuming to identify, and thus, such a list is not likely to be comprehensive. Even if there was such a comprehensive list available, it would be expensive to build a model for each enlisted pattern and practically difficult to maintain them.

• Third, these discriminating patterns can exist in different data modalities, *e.g.* the dark background exists in the visual signal, while music exists in the audio signal. We need to use multiple data modalities effectively, and it is unclear how to optimally do that for our problem.

Contributions:
To solve these challenges, we propose a deep learning based solution that makes the following key contributions:

• First, our detection approach is trainable end-to-end and therefore easy to maintain at scale. We use convolution neural networks (CNN) to extract discriminable visual and audio features from video-titles. These features are fed into a bidirectional long short term memory (B-LSTM) network to capture the various long and short term dependencies among different frame-features. Finally, for optimal temporal smoothing, we use a conditional random field (CRF) to jointly optimize the labels of all video frames, *i.e.*, *recap*, *intro*, or the actual content.

• Second, our approach allows us to model multiple information channels in a unified manner. We present a detailed empirical analysis of how multi-modal fusion performs for the problem at hand with both early and late fusion schemes.

• Third, our framework is applicable to both the *intro* and *recap* detection problems and can be extended to detect other types of video-segments, *e.g.* end-credits. Our solution is easy to put into production since its inference does not depend on preceding episodes, thereby reducing its memory requirements and maintenance overhead.

We present a thorough empirical analysis of our model compared to several other deep learning based architectures and demonstrate its superior performance. We begin in Section 2 by discussing the related work and explaining our approach in more detail in Section 3. We present our dataset, evaluation metrics, experiment settings, and a thorough set of comparative empirical results in Section 4 and discuss the conclusion and future work in Section 5.

2. Related Work

Automatic *intro* and *recap* detection is a relatively new problem, and closely relates to temporal action localization, named-entity recognition, and multi-modal fusion. Below, we briefly go over previous approaches in these three domains and distill how our approach is different from them.

Temporal Action Localization. The problem of finding semantically meaningful clips or segments within a video is closely related to the domain of temporal action localization where the goal is to identify action classes in untrimmed videos as well as the corresponding start and end timestamps of each action [35, 5, 43, 42, 2, 22]. Modern solutions of temporal action localization use a two-stage approach, where the first stage focuses on generating region proposals while the second stage focuses on action classification and boundary refinement [44, 35, 43]. There exist several end-to-end methods that combine the proposal generation and action classification steps [2, 42, 22]. A key difference between temporal action localization and our problem is that in our case each video only contains at most one *intro* and one *recap* while there could be multiple instances of the same action in the general temporal action localization problem. Another key difference is the tolerance of

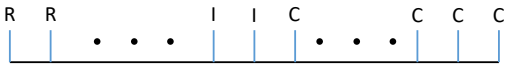

Figure 2: Illustration of our sequence labeling for *intro* and *recap* detection problem. Each of the two consecutive blue bars represent a time interval of a title. The labels above demonstrate the type of the interval, where R represents *recap*, I represents *intro*, and C represents actual title content.

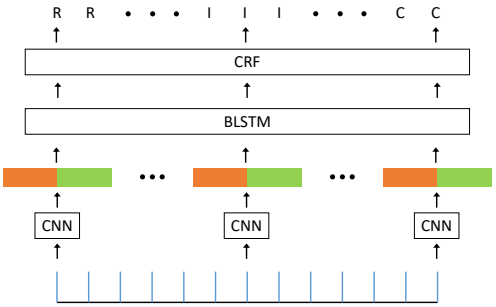

Figure 3: Overview of our network architecture. Here blue vertical lines represent the video frames, while the orange and green bars represent the visual and audio features of the corresponding frames respectively. Letters R, I and C correspond to *recap*, *intro* and actual title content respectively.

action boundary error. In temporal action localization, a detected region is generally considered as a correct detection when it significantly overlaps with the ground truth where the significance of overlap is determined by a metric such as intersection-over-union (IoU) [23] ranging anywhere from 0.1 to 0.9 in general. In contrast, our use-case is significantly more sensitive to the detected region boundary. We can only treat a detected region as a correct detection when it almost completely overlaps with the ground truth. Thus, we introduce a metric based on the absolute difference between identified boundary and ground truth in Section 4.2.

Named Entity Recognition. Another related field to our problem is Named Entity Recognition (NER) [26, 18, 6], where the goal is to classify each token in a sentence into predefined categories, such as name, location, *etc*. Many different type of classifiers have been used to recognize named entities, including Hidden Markov models [25], maximum entropy Markov models [24], and Conditional Random Fields (CRFs) [17] for sequence learning [18]. Our problem is related to NER in the sense that each frame of a video can be treated as a token and then each token can be classified as one of the categories, *i.e.*, *intro*, *recap*, or the actual content. Unlike previous solutions for NER, we adopted bidirectional LSTMs to encode temporal dependencies for video frames and use CRF to optimize the labels of all video frames.

Multimodal Fusion. Our problem is also related to multi-

modal fusion. Multi-model fusion has been studied for various tasks to leverage different types of features, such as image and text fusion for product classification [12], visual and audio fusion for video description [14], and temporal multimodal fusion for learning driving behavior [28]. Various type of strategies have been studied including Boltzmann machines [29], attention [14], and gated recurrent fusion [28]. Fusion of different modalities can often be achieved using two general mechanisms: early and late fusion [27, 29, 11]. In early fusion, the features from different modalities are combined together first before feeding them to a model, while in late fusion the features from each modality are first fed into a separate model and the model outputs are fused subsequently. While early fusion allows the model to explore the interaction of different features from each modality [12], the choice of early or late fusion mostly depends on model and sample complexity of a problem. In this work, we study the effectiveness of early and late fusion as a function of our model complexity.

3. Model Architecture

We now explain our formulation of *intro* and *recap* detection as a supervised learning problem and introduce the various components of our model architecture.

We frame *intro* and *recap* detection as a sequence labeling problem (see Figure 2 for an illustrative example). The input for our problem is a time series of video intervals and the goal is to label each time interval with one of three possible labels, *i.e.*, *intro* (I), *recap* (R), or actual content (C). Note that while in Figure 2 *recap* starts from the very beginning of the title, followed by *intro* and then content, this order is not necessarily always the case. For example, there are video-titles with *cold-openings* that jump directly into the story at the beginning before *intros* or *recaps* are shown. This order-variance in which *intro*, *recap* and content can appear in titles further adds complexity to our problem.

The architecture of the network we propose to overcome these technical challenges is shown in Figure 3. It consists of the following three main components:

- CNN for visual and audio feature extraction (§ 3.1).
- B-LSTM for capturing long and short term temporal dependencies among various frame-features (§ 3.2).
- CRF for joint label optimization of video frames (§ 3.3).

We now present these components in more detail.

3.1. CNN for Visual and Audio Features

CNNs are deep networks that leverage the spatial nature of images to enable extraction of discriminative image features. They have been shown to outperform traditional shallow-learning or hand-crafted Computer Vision techniques and have demonstrated excellent performance in many vision tasks (*e.g.* object recognition [31] and handwritten digit classification [20]). Moreover, it has been shown that CNNs can be a powerful way to process audio signals. For instance, 1D ConvNet [19] can be applied to raw audio features to accurately capture the frame-level audio transitions resulting in better audio representations for applications like rare sound event detection [21].

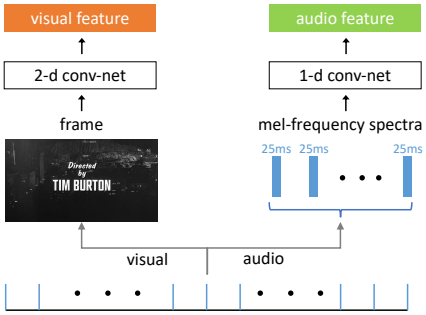

Figure 4: Detailed illustration of the CNN components for the visual and audio feature extraction.

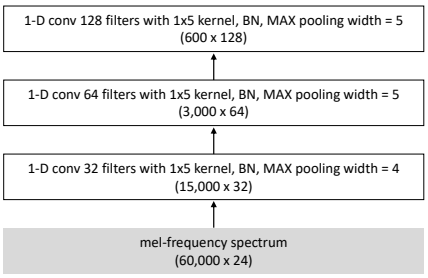

Figure 5: Our 1D ConvNet component: three steps of 1D convolution, BN, and MAX pooling

The CNN based feature extraction component we use in this work has three main steps to compute both visual as well as audio features (see Figure 4 for illustration). First, it extracts visual features from video-frames by feeding the frames to a pre-trained CNN. Second, it extracts audio features by applying 1D ConvNet [19] on the log mel-frequency spectrum of the audio signals. Third, it combines the audio-visual features using multi-modal data fusion.

Pre-trained CNN for Frame Representation. We use the Inception V3 deep network [37] pre-trained on ImageNet [32] to extract our frame level visual features. Specifically, we extract image frames from video titles at one frame per second (1 FPS), feed these frames into the Inception-V3 network, and then extract the ReLu activations of the last hidden layer (2048 dimension) as the frame feature. The choice of using 1-second temporal granularity was made due to its suitability for our use case as well as its effectiveness from a computational complexity perspective.

1D Convolution for Audio Representation. Following the work in [21], for the audio signal of videos, we first extract

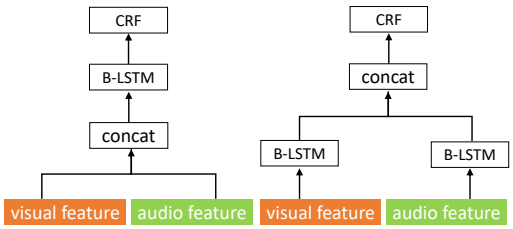

Figure 6: (a): Early fusion; (b): Late fusion.

log-power mel-frequency spectrum [9] with 24 mel-scale filters from every 25ms window with 10ms shift between two consecutive windows. The extracted log-mel spectrogram is passed through our 1D ConvNet component to get the audio features. The 1D ConvNet component has three steps, where each step consists of a 1D convolution, a batch normalization (BN), and a MAX pooling layer. The three steps gradually aggregate the audio signal and results in a 128-d vector for each second of signal as shown in Figure 5.

Audio-Visual Fusion. To leverage the video and audio features jointly in a unified way and use them in a signal neural network model, we explored early and late fusion technique to combine the two data modalities. For early fusion, as shown in Figure 6a, we concatenate the video features (2048 dimension) and audio features after 1D ConvNet (128-d). Recall that both these feature spaces are constructed over 1 second temporal scale. This results in a 2176 dimension vector for each second of a title and those vectors serve as the input to the B-LSTM units (note that the network architecture overview shown in Figure 2 is illustrated with early fusion). For late fusion, as shown in Figure 6b, instead of fusing the visual-audio features before B-LSTM units, we feed the visual and audio feature separately to their own B-LSTM units and then concatenate the outputs of the B-LSTM units to serve as the input to the CRF units.

3.2. Bi-directional LSTM

Recall that Recurrent Neural Networks (RNNs) are designed to capture the dynamic changes between consecutive data points along sequences. LSTMs [13] are a variant of RNNs that are designed to alleviate shortcomings of RNNs, particularly the gradient vanishing problem [30].

LSTMs however only capture dependencies from the past, whereas for many sequence labeling problems, it is beneficial to know both the past and the future dependencies. This is especially true for our *intro* and *recap* detection problem. For example, since *intro* is usually followed by the actual title content, knowing the actual title content has already started will help the model to reaffirm that the *intro* has already ended. An elegant solution to systematically take the future information into consideration

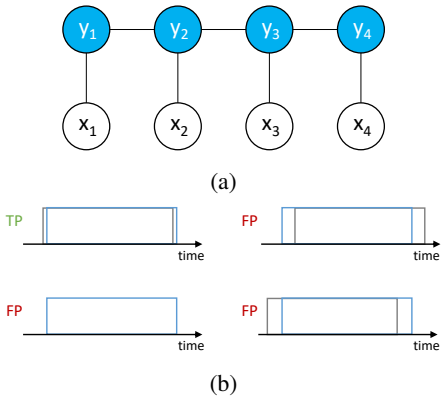

Figure 7: (a): A Linear-Chain CRF Graph; (b): Examples of true positive (TP) and false positive (FP). Gray boxes indicate the labeled *intro* or *recap* if they exist and blue boxes indicate the corresponding model detections.

in addition to the past information is to use bi-directional RNNs [33]. Therefore, we use B-LSTM to consume our features in our network structure. In addition, since it is observed that deep RNNs work better than shallower ones for many problems [38, 15, 16] including sequence labeling tasks like NER [38] and option mining [15], we stacked two layers of B-LSTM (with 512 and 128 neurons respectively).

3.3. Conditional Random Field

While B-LSTMs can model the dependencies among input information, they do not explicitly account for the dependencies among the output labels. To infer the labels for an input sequence, it is better to consider the transition between labels in neighborhoods and jointly infer the optimal sequence of labels. For example, *intro* is mostly likely to be followed by actual title content in a video title, and this information should be more explicitly modeled to reach optimal detection performance. As CRFs [17] provide a way to model such dependencies, we use a CRF (shown in Figure 7a) to jointly learn the sequence labeling instead of decoding each label independently.

More specifically, let $x = \{x_1, ..., x_n\}$ represents an input sequence, where x_i is the output from the i^{th} timestamp in B-LSTM. Let $y = \{y_1, ..., y_n\}$ represents a sequence of labels for x. Then the conditional probability $p(y|x)$ in CRF is defined as [36]

$$p(y|x) = \frac{1}{Z} \prod_{i=1}^{I} f_i(y_{i-1}, y_i, x),$$

where $Z = \sum_y \prod_{i=1}^{I} f_i(y_{i-1}, y_i, x)$. Similarly, the the i^{th} potential function $f_i(y_{i-1}, y_i, x)$ is defined as $\exp(W^T_{y_{i-1},y_i} x + b_{y_{i-1},y_i})$. Here W and b are the weight matrix and bias term corresponding to label pair (y_{i-1}, y_i).

For CRF training, we use the maximum likelihood estimation to train CRF by maximizing the following log-likelihood loss function over all M training samples:

$$L(W, b) = \sum_{m=1}^{M} \log p(y|x; W, b),$$

During training, it is required to compute the normalization constant Z for the likelihood, which can be computed efficiently using forward-backward algorithm [7]. For inference, we search for the optimal label sequence y^* with highest conditional probability $y^* = \arg\max_y p(y|x; W, b)$ efficiently using the Viterbi decoding algorithm [10].

4. Experiments

4.1. Data Deep Dive

We manually collected a data-set containing 46,946 video titles with 91.4% of the titles being TV episodes and the remaining ones being movies. Each title runs at 24 or 25 frames per second and is manually reviewed by an operator. The operators checked the whole video for a list of elements including *intro* and *recap*. The corresponding start and end timestamps were recorded for both *intro* and *recap* when they existed. Each title could only have one *intro* and one *recap* at most. When the *intro* or *recap* did not exist, the corresponding timestamps fields were left empty. To ensure label quality, all titles went through a mandatory QA pass with a maximum of 1.6 speed to verify information against the provided timestamps.

In our data-set, all of the tagged *intro* and *recap* are within the first 10 minutes of titles. Therefore, we process each title up to the first 10 minutes (600 seconds) for both visual and audio feature extraction. Among all of the 46,946 titles, 15,934 (33.9%) titles have *recaps*, 45,027 (95.9%) titles have *intros*, and 14,015 titles have both *recaps* and *intros*. Please refer to the supplementary material for more details of the labeled data-set. For model training, the whole data-set is randomly split into training (33,717), validation (8559), and test (4,670) sets based on the normalized series and movie names. This way, all titles in the same TV series end up either in one or in the other split to prevent information leakage between any pair of the split sets.

4.2. Evaluation Metrics and Training Setup

To quantify the performance of our detection models, we need to first define how to measure if a detection is valid (true positive) or not (false positive). Inspired by the true positive (TP) definition in object detection problem, such as COCO detection [23], where TP is defined as detections with Intersection Over Union (IOU) above some threshold, we define TP as detections where both the start and the end timestamps are within T seconds from the corresponding ground truths, where T is the generally small (1-3 secs.) tolerance of the timestamp boundary distance. Any detection that does not meet this criterion is considered as false positive (FP). Figure 7b shows some illustrative examples of TP and FP, where gray boxes indicate the labeled *intro* or *recap* ground truths and blue boxes indicate model detections. Anytime the start and end timestamps of the detection are very close to the ground truth is considered as a TP, while all other possibilities are considered as FP.

With the above definition of TP, we measure our model for *intro* and *recap* independently using precision and recall. Precision is the percentage of correctly detected *intros* and *recaps* among all detections and measures the ability of a model to identify correct *intros* and *recaps*. Recall is the percentage of ground-truth *intros* and *recaps* that are correctly detected and measures the ability of a model to identify all ground truth *intros* and *recaps*. We also calculate the F_1 score to facilitate model comparison. Parameter optimization is performed using mini-batch SGD [1] with RMSProp [8], batch size of 128, and learning rate of 0.001. We use early stopping [4] based on the performance on the validation set (please see the supplementary material for more details on experiment setup).

4.3. Results

We now summarize the key results from our experiments. Table 1 shows the performance of our proposed model on the validation data-set with 1-second tolerance and compares it with the results from three types of localization models using visual features, namely Loc-LSTM, Loc-1D CNN, and Loc-TSN. The first two models are implemented based on temporal action localization [22, 2], where the overall loss function is a combination of the classification loss of *intro* and *recap* existence and regression loss of *intro* and *recap* region. The model Loc-LSTM has two layers of LSTM (with 512 and 128 neurons respectively) before the loss layer to learn the video representation, while Loc-1D CNN uses 4 layers of 1D ConvNet (with 1024, 512, 256, and 128 neurons respectively) to replace the LSTM layers. The third model is based on temporal segment network (TSN) [39], where an input video is divided into segments with 1-second length, and the corresponding temporal segment supervision is used to train a two-layer fully connected neural network (with 512 and 128 neurons respectively) to classify the segments to *intro*, *recap*, or actual content.

We can see from Table 1 that our proposed method has significantly better performance than the alternative approaches that frame *intro* and *recap* detection as a temporal action localization problem using visual feature. Note that in our data-set, the average lengths of *recap* and *intro* are 54 and 40 seconds respectively, which results in a substantially high IOU value of 0.981 for *recap* and 0.975 for *intro* using 1-second tolerance of timestamp boundary.

Architecture Name	Intro			Recap		
	Precision	Recall	F_1	Precision	Recall	F_1
Loc-LSTM	0.56%	0.56%	0.56%	0.55%	0.55%	0.55%
Loc-1D CNN	0.41%	0.41%	0.41%	1.20%	1.20%	1.20%
Loc-TSN	8.68%	9.05%	8.86%	0.01%	0.03%	0.02%
Bi-LSTM+ CRF	63.15%	56.79%	59.80%	70.08%	64.15%	66.99%

Table 1: Model comparison using visual feature. For discussion of the results, refer to the 2^{nd} paragraph of section 4.3

#	Architecture Name	Feature Type	Intro			Recap		
			Precision	Recall	F_1	Precision	Recall	F_1
1	1-layer-LSTM	Audio	13.83%	13.34%	13.58%	27.10%	31.32%	29.05%
		Visual	31.90%	32.03%	31.97%	21.33%	26.50%	23.64%
		Early Fusion	28.61%	28.81%	28.71%	33.97%	40.18%	36.81%
		Late Fusion	38.43%	39.33%	38.87%	40.33%	46.49%	43.19%
2	LSTM	Audio	16.56%	16.52%	16.54%	22.92%	19.41%	21.02%
		Visual	37.94%	33.62%	35.65%	46.48%	49.15%	47.78%
		Early Fusion	41.23%	41.64%	41.44%	42.66%	48.96%	45.59%
		Late Fusion	40.03%	41.01%	40.51%	44.35%	51.99%	47.87%
3	B-LSTM	Audio	31.94%	29.54%	30.69%	49.59%	54.05%	51.73%
		Visual	45.81%	46.05%	45.93%	58.46%	61.94%	60.15%
		Early Fusion	59.87%	60.68%	60.27%	67.05%	68.43%	67.73%
		Late Fusion	49.36%	50.51%	49.92%	58.66%	56.43%	57.52%
4	Bi-LSTM+Viterbi	Audio	36.92%	28.24%	32.00%	51.64%	53.45%	52.53%
		Visual	47.68%	42.13%	44.73%	59.30%	60.67%	59.98%
		Early Fusion	62.25%	58.66%	60.40%	68.03%	67.92%	67.98%
		Late Fusion	51.36%	49.18%	50.25%	62.90%	57.88%	60.29%
5	Bi-LSTM+ CRF	Audio	55.22%	39.32%	45.93%	67.04%	51.77%	58.42%
		Visual	63.15%	56.79%	59.80%	70.08%	64.15%	66.99%
		Early Fusion	68.56%	64.90%	66.68%	74.68%	70.96%	72.77%
		Late Fusion	65.96%	62.18%	64.01%	73.27%	59.97%	65.96%

Table 2: Model performance of different architectures

The particularly low accuracy results we observe for the alternative approaches we tried are similar to those previously reported in temporal action localization litrature [43] at similarly high values of IOUs. In particular, [34] and [5] report accuracy of 0.20% and 1.30% respectively on ActivityNet v1.3 data-set [3] at IOU of 0.95. The Loc-TSN approach performs better than the other two localization based methods, but since the TSN segments only capture local information inside each segment, it does not capture the long term dependencies over time. In contrast, our proposed method captures the long and short term dependencies and models the temporal transitions at the boundaries of *intro* and *recap* more explicitly. This makes it easier for the model to capture signals such as start and end of title-credits as well as opening voice like "previously on" thereby resulting in significantly better model performance at 1-second tolerance.

Architecture 5 (Bi-LSTM+ CRF) in Table 2 shows the performance of our proposed on the validation data-set with 1-second tolerance using different types of feature, namely, audio, visual, early fusion, and late fusion of audio and visual features. We can see that the early fusion results in the best model performance among the four types, indicating that our architecture has the capacity to model both the visual and audio feature well when they are fed to the model as inputs. Moreover, the early fusion model is computationally efficient and given the input features only requires 70 ms on average to infer the *intro* and *recap* of each video-title on a GPU machine with one NVIDIA Tesla V100.

We also calculate the performance of our model with early fusion for different tolerance values on the test data-set. Similar to metrics on the validation data-set, our model achieved high accuracy with 73.22% precision, 68.64% recall, and 70.85% F_1 score for *intro* and 73.46% precision, 67.89% recall, and 70.57% F_1 score for *recap* on the test data with 1-second tolerance. In addition, all metrics including precision, recall, and F_1 score increased by more

than 8% as the tolerance value is relaxed to 3 seconds. More detail can be found in the supplementary material.

4.4. Ablation Study

To investigate the contribution of several key components in our proposed architecture, we compare our model architecture with four other deep learning based architectures. For all these architectures, the CNN feature extraction component is fixed to be the same as the one described in Section 3.1. In order to distill the effectiveness of using the B-LSTM and CRF components, we set up the architecture comparison to be incremental where only one change is made for a pair of consecutive architectures considered.

First, we start with a simpler deep learning architecture with one single LSTM layer (512 neurons) and a dense layer with softmax activation function (architecture 1 in Table 2). No CRF layer is used and the cross entropy loss is applied during model training. During inference, the label with highest probability is chosen as the prediction at each timepoint. Second, we add an additional LSTM layer with 128 neurons on top of the existing LSTM layer in architecture 1 to analyze the benefit of RNN stacking (architecture 2). Third, we replace both LSTM layers in architecture 2 with B-LSTM layers to study the effectiveness of B-LSTM (architecture 3). Fourth, instead of always choosing the mostly likely label for each timepoint, we use Viterbi decoding during inference compared to architecture 3 to test whether it is enough to consider label smoothness only during inference (architecture 4). Finally, we replace the dense layer in architecture 3 with a CRF layer and use CRF loss during training compared to architecture 3 (architecture 5) to analyze the importance of considering label smoothness with CRF loss in training. The model performances are shown in Table 2, and we summarize the key findings as follows.

a: B-LSTM and CRF Matters. When comparing the performance of the five architectures, there is a significant improvement at each incremental change between each pair of consecutive architectures. The magnitude of the improvement is especially big when LSTM is replaced with B-LSTM in architecture 3 and when CRF layer is added in architecture 5. For example, among all models trained with different features using LSTM architecture 2, the best one (in terms of F_1 score) achieves 41.23% precision and 41.64% recall with early fusion for *intro* and 44.35% precision and 51.99% recall with late fusion for *recap*. When LSTM layers are replaced with B-LSTM layers in architecture 3, the best one achieves 59.87% precision and 60.68% recall for *intro* and 67.05% precision and 68.43% recall for *recap* with early fusion. In addition, we can see that while there is a marginal improvement in F_1 score of the early fusion model when adding Viterbi decoding in architecture 4 compared to architecture 3, the magnitude of improvement is significantly bigger when CRF loss is used during training in architecture 5, Specifically, when CRF is added in architecture 5, the best one achieves 68.56% precision and 64.90% recall for *intro* and 74.68% precision and 70.96% recall for *recap* with early fusion.

b: Feature Fusion is Beneficial. When visual and audio feature are fused together, the fused features generally perform significantly better than audio or visual features used independently especially when B-LSTM and CRF are used. There are cases where early or late fusion performs worse than using visual signal only such as the early fusion for *recap* in architecture 2 and the late fusion for *recap* in architecture 3, but when B-LSTM and CRF components are added in architecture 5, both the early and late fusion work significantly better than audio or visual features used separately. It is also observed that early fusion results in better performance as the complexity of architecture increases, which suggests that more complex architecture can model both the visual and audio feature better. In particular, the architecture 5 with early fusion achieves the best model performance with 68.56% precision and 64.90% recall for *intro* and 74.68% precision and 70.96% recall for *recap*.

c: Stacking Helps. When comparing different data modalities, visual signal has more predictive power than audio signal for both *intro* and *recap* in almost all cases. The only exception is for the *recap* in architecture 1 where audio signal performs better than visual signal, but when LSTM layers are stacked in architecture 2, visual features start to outperform audio for *recap* detection. In addition, stacking LSTM improves the performance of models with fused features when comparing architecture 1 and architecture 2.

5. Conclusion and Future Work

In this work, we presented a novel end-to-end deep learning prototype for automated *intro* and *recap* identification by leveraging CNN, B-LSTM, and CRF in an unified framework. We used CNNs to extract visual and audio features, fused the features into B-LSTM to capture the context changes among consecutive frames, and then used CRF to jointly optimize the sequence labeling for *intros* and *recaps*. We demonstrated that our model can achieve high accuracy and perform much better compared to several other deep architectures. We showed that B-LSTM and CRF components were critical to get high accuracy and the fused features generally perform significantly better than audio or visual features used independently, especially when B-LSTM and CRF are used. The approach can be extended to detect other video content, such as end credits, and can also be applied to other sequence learning applications. Going forward, we plan to experiment other ways of visual and audio feature fusion to improve detection accuracy. Moreover, caption information is not incorporated into our model due to its limited coverage, and we plan to explore ways to overcome this data limitation to enhance our model.

References

[1] Léon Bottou. Large-scale machine learning with stochastic gradient descent. In *Proceedings of COMPSTAT'2010*, pages 177–186. Springer, 2010.

[2] Shyamal Buch, Victor Escorcia, Bernard Ghanem, Li Fei-Fei, and Juan Carlos Niebles. End-to-end, single-stream temporal action detection in untrimmed videos. In *BMVC*, volume 2, page 7, 2017.

[3] Fabian Caba Heilbron, Victor Escorcia, Bernard Ghanem, and Juan Carlos Niebles. Activitynet: A large-scale video benchmark for human activity understanding. In *Proceedings of the ieee conference on computer vision and pattern recognition*, pages 961–970, 2015.

[4] Rich Caruana, Steve Lawrence, and C Lee Giles. Overfitting in neural nets: Backpropagation, conjugate gradient, and early stopping. In *Advances in neural information processing systems*, pages 402–408, 2001.

[5] Yu-Wei Chao, Sudheendra Vijayanarasimhan, Bryan Seybold, David A Ross, Jia Deng, and Rahul Sukthankar. Rethinking the faster r-cnn architecture for temporal action localization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1130–1139, 2018.

[6] Jason PC Chiu and Eric Nichols. Named entity recognition with bidirectional lstm-cnns. *Transactions of the Association for Computational Linguistics*, 4:357–370, 2016.

[7] Aron Culotta and Andrew McCallum. Confidence estimation for information extraction. In *Proceedings of HLT-NAACL 2004: Short Papers*, pages 109–112, 2004.

[8] YN Dauphin, H De Vries, J Chung, and Y Bengio. Rmsprop and equilibrated adaptive learning rates for non-convex optimization. arxiv 2015. *arXiv preprint arXiv:1502.04390*, 2015.

[9] Steven Davis and Paul Mermelstein. Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences. *IEEE transactions on acoustics, speech, and signal processing*, 28(4):357–366, 1980.

[10] G David Forney. The viterbi algorithm. *Proceedings of the IEEE*, 61(3):268–278, 1973.

[11] Ignazio Gallo, Alessandro Calefati, and Shah Nawaz. Multimodal Classification Fusion in Real-World Scenarios. *Proceedings of the International Conference on Document Analysis and Recognition, ICDAR*, 5:36–41, 2018.

[12] I. Gallo, A. Calefati, S. Nawaz, and M. K. Janjua. Image and Encoded Text Fusion for Multi-Modal Classification. *2018 International Conference on Digital Image Computing: Techniques and Applications, DICTA 2018*, 2018(December):10–13, 2019.

[13] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[14] Chiori Hori, Takaaki Hori, Teng Yok Lee, Ziming Zhang, Bret Harsham, John R. Hershey, Tim K. Marks, and Kazuhiko Sumi. Attention-Based Multimodal Fusion for Video Description. *Proceedings of the IEEE International Conference on Computer Vision*, 2017-Octob:4203–4212, 2017.

[15] Ozan Irsoy and Claire Cardie. Opinion mining with deep recurrent neural networks. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 720–728, 2014.

[16] Rafal Jozefowicz, Oriol Vinyals, Mike Schuster, Noam Shazeer, and Yonghui Wu. Exploring the limits of language modeling. *arXiv preprint arXiv:1602.02410*, 2016.

[17] John Lafferty, Andrew McCallum, and Fernando CN Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. 2001.

[18] Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. Neural architectures for named entity recognition. *arXiv preprint arXiv:1603.01360*, 2016.

[19] Colin Lea, Rene Vidal, Austin Reiter, and Gregory D Hager. Temporal convolutional networks: A unified approach to action segmentation. In *European Conference on Computer Vision*, pages 47–54. Springer, 2016.

[20] Yann LeCun, Bernhard Boser, John S Denker, Donnie Henderson, Richard E Howard, Wayne Hubbard, and Lawrence D Jackel. Backpropagation applied to handwritten zip code recognition. *Neural computation*, 1(4):541–551, 1989.

[21] Hyungui Lim, Jeongsoo Park, and Y Han. Rare sound event detection using 1d convolutional recurrent neural networks. In *Proc. of DCASE*, pages 80–84, 2017.

[22] Tianwei Lin, Xu Zhao, and Zheng Shou. Single shot temporal action detection. In *Proceedings of the 25th ACM international conference on Multimedia*, pages 988–996, 2017.

[23] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014.

[24] Robert Malouf. Markov models for language-independent named entity recognition. In *COLING-02: The 6th Conference on Natural Language Learning 2002 (CoNLL-2002)*, 2002.

[25] Sudha Morwal, Nusrat Jahan, and Deepti Chopra. Named entity recognition using hidden markov model (hmm). *International Journal on Natural Language Computing (IJNLC)*, 1(4):15–23, 2012.

[26] David Nadeau and Satoshi Sekine. A survey of named entity recognition and classification. *Lingvisticae Investigationes*, 30(1):3–26, 2007.

[27] Katsuyuki Nakamura, Serena Yeung, Alexandre Alahi, and Li Fei-Fei. Jointly learning energy expenditures and activities using egocentric multimodal signals. *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, 2017-Janua:6817–6826, 2017.

[28] Athma Narayanan, Avinash Siravuru, and Behzad Dariush. Gated Recurrent Fusion to Learn Driving Behavior from Temporal Multimodal Data. *IEEE Robotics and Automation Letters*, 5(2):1287–1294, 2020.

[29] Jiquan Ngiam, Aditya Khosla, Mingyu Kim, Juhan Nam, Honglak Lee, and Andrew Y. Ng. Multimodal deep learning. *Proceedings of the 28th International Conference on Machine Learning, ICML 2011*, pages 689–696, 2011.

[30] Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. On the difficulty of training recurrent neural networks. In *International conference on machine learning*, pages 1310–1318, 2013.

[31] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, pages 91–99, 2015.

[32] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115(3):211–252, 2015.

[33] Mike Schuster and Kuldip K Paliwal. Bidirectional recurrent neural networks. *IEEE Transactions on Signal Processing*, 45(11):2673–2681, 1997.

[34] Zheng Shou, Jonathan Chan, Alireza Zareian, Kazuyuki Miyazawa, and Shih-Fu Chang. Cdc: Convolutional-deconvolutional networks for precise temporal action localization in untrimmed videos. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5734–5743, 2017.

[35] Zheng Shou, Dongang Wang, and Shih-Fu Chang. Temporal action localization in untrimmed videos via multi-stage cnns. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1049–1058, 2016.

[36] Charles Sutton, Andrew McCallum, et al. An introduction to conditional random fields. *Foundations and Trends® in Machine Learning*, 4(4):267–373, 2012.

[37] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2818–2826, 2016.

[38] Quan Tran, Andrew MacKinlay, and Antonio Jimeno Yepes. Named entity recognition with stack residual lstm and trainable bias decoding. *arXiv preprint arXiv:1706.07598*, 2017.

[39] Limin Wang, Yuanjun Xiong, Zhe Wang, Yu Qiao, Dahua Lin, Xiaoou Tang, and Luc van Gool. Temporal segment networks: Towards good practices for deep action recognition. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 9912 LNCS:20–36, 2016.

[40] Kelly West. Unsurprising: Netflix survey indicates people like to binge-watch tv. In *Cinema Blend*, 2014.

[41] Xiao Wu, Alexander G Hauptmann, and Chong-Wah Ngo. Practical elimination of near-duplicates from web video search. In *Proceedings of the 15th ACM international conference on Multimedia*, pages 218–227. ACM, 2007.

[42] Serena Yeung, Olga Russakovsky, Greg Mori, and Li Fei-Fei. End-to-end learning of action detection from frame glimpses in videos. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2678–2687, 2016.

[43] Runhao Zeng, Wenbing Huang, Mingkui Tan, Yu Rong, Peilin Zhao, Junzhou Huang, and Chuang Gan. Graph convolutional networks for temporal action localization. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 7094–7103, 2019.

[44] Yue Zhao, Yuanjun Xiong, Limin Wang, Zhirong Wu, Xiaoou Tang, and Dahua Lin. Temporal action detection with structured segment networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2914–2923, 2017.