Driver Behavior Extraction from Videos in Naturalistic Driving Datasets with 3D ConvNets

Hanwen Miao · Shengan Zhang · Carol Flannagan

Received: 26 May 2021 / Revised: 23 January 2022 / Accepted: 18 May 2022 / Published online: 24 June 2022
© The Author(s), under exclusive licence to Springer Nature Singapore Pte Ltd. 2022

Abstract
Naturalistic driving data (NDD) is an important source of information to understand crash causation and human factors and to further develop crash avoidance countermeasures. Videos recorded while driving are often included in such datasets. While there is a large amount of video data in NDD, only a small portion of it can be annotated by human coders and used for research. In this paper, we explored a computer vision method to automatically annotate behaviors in videos. More specifically, we developed a 3D ConvNet algorithm to automatically extract cell phone behaviors from videos. The experiments show that our method can extract chunks from videos, most of which (∼85%) contain the automatically labeled cell phone behaviors. Importantly, we discuss and evaluate two use cases: (1) using algorithm labels without subsequent human review, and (2) using algorithm labels with subsequent human review. We find that even a 99% accurate algorithm will produce statistics that are appreciably biased towards the null, relative to ground truth, when labels are used without review. Thus, while the algorithm is not accurate enough to support the direct use of its labels in analysis, in conjunction with a human review of the extracted chunks, this approach can find cell phone behaviors much more efficiently than simply viewing a video.

Keywords Cell phone behaviors · Video processing · 3D ConvNets

Introduction
Research and data on automotive safety have been a high priority for many decades. The Fatality Analysis Reporting System (FARS), a census of all fatal crashes in the U.S., began in 1975 (National Highway Traffic Safety Administration 2018). This national dataset enabled analysis and tracking of the most severe crashes and began to lead to improvements in vehicle design. The National Automotive Sampling System Crashworthiness Data System (NASS-CDS) was launched in 1979 (Radja 2016) and included detailed crash investigations of towaway crashes involving light vehicles. This dataset provided key information about causes of injury and death that enabled research and development into the design and mandate of seatbelts, airbags, crush zones, and many other safety systems that have saved tens of thousands of lives.

In more recent years, automotive safety research has advanced understanding of crash causation, which has led to the development of crash avoidance countermeasures. A key source of data to inform this development has been naturalistic driving data (NDD), where participants’ vehicles are equipped with a variety of sensors, including cameras (interior and exterior facing), accelerometers, and GPS’s, among others. The largest such dataset in the world was collected through the Second Strategic Highway Research Program (SHRP2) Naturalistic Driving Study (NDS) (Hankey et al. 2016).

While NDD has supported hundreds of research papers and advances in crash countermeasures and understanding human factors of drivers, the video data are generally grossly underused. The standard approach to obtaining analyzable information from video data is to use human coders, who watch hours of video and annotate such things as driver activities (including secondary tasks), Safety–Critical Events (SCEs) (which are identified using kinematic triggers and verified using video), presence/absence of passengers,
use of seat belts or car seats, weather and road conditions, etc. (Precht et al. 2017; Bharadwaj et al. 2019; Bao et al. 2020). Video-coded information is key to understanding driver risk factors (e.g., texting while driving) and precrash scenarios and kinematics (through identification of SCEs). This limitation means that, for example, of the over 1 million hours of driving in the SHRP2 NDS, the most-used video-based data is 200 h that were annotated in the 32,500 20-s clips of baseline driving and approximately 1500 SCEs (i.e., crashes and near-crashes) (Hankey et al. 2016).

One potential way to dramatically increase the usability of the video data is by having computer vision algorithms automatically annotate activities or objects of interest. The field of computer vision has advanced significantly in the last decade. Notably, the simultaneous development of neural network methods and hardware for computational capacity have supported major improvements in algorithm performance. Initially, convolutional neural networks (ConvNets) (Ciregan et al. 2012) were applied to images and achieved improvements on some tasks like image classification compared to older methods. With the success of algorithmically labeling static images, researchers extended the approach to label activities in videos.

Activity labeling introduces challenges for computer vision because of the added temporal dimension. In the computer vision domain, many algorithms have been proposed to localize actions in untrimmed videos. Shou et al. (2016) proposed a method with three segment-based 3D ConvNets. Chao et al. (2018) proposed a TAL-NET, which is inspired by the Faster R-CNN object detection framework. Zeng et al. (2019) proposed a Graph Convolutional Networks to exploit the proposal-proposal relations to better improve localization. Gleason et al. (2020) proposed a method to localize activities via classifying and aggregating fixed-length chunks.

In general, the dominant approaches are either 2D-ConvNets or 3D Convnets (Chen et al. 2021), where the former analyzes space and time separately, and the latter analyzes them together. 2D ConvNets either use two streams (one for RGB and one for optical flow) that are fused late in upper layers (e.g., Karpathy et al. 2014) or two algorithms in sequence, the first of which identifies features or objects in each frame and the second of which analyzes their movements over time (e.g., Zhou et al. 2018). In contrast, 3D ConvNets use three-dimensional features in the convolutional layers (Tran et al. 2015). More recent algorithms have built on these basic approaches using additional features such as using both slow and fast frame-rates to capture different levels of temporal patterns (Feichtenhofer et al. 2019).

A 2021 paper Chen et al. (2021) presented a standardized approach to benchmarking activity-recognition algorithms and found that once the backbone network, input sampling approach and temporal pooling are equated, the previous state-of-the-art inflated 3D ConvNet (I3D) (Carreira and Zisserman 2017) produces accuracies comparable with newer models such as SlowFast (Feichtenhofer et al. 2019) that had demonstrated improved accuracy but was trained on a different backbone. The original I3D ConvNet is based on ImageNet-pretrained Inception-V1 (Szegedy et al. 2015), which is a neural network designed for 2D image processing. All the filters and pooling kernels in Inception-V1 were inflated to 3D to add an additional temporal dimension.

While the state of the art is advancing in computer science, in the application area of driving, many current algorithms to detect activities such as driver behaviors are based on hand-crafted features, which are modeled with Support Vector Machines (SVM), Hidden Markov Models (HMM), or Recurrent Neural Networks (RNN) (e.g., see discussion in Gebert et al. (2019)). However, there is increasing use of deep learning and more state-of-the-art approaches in the field. For example, Gebert et al. (2019) sought to improve upon hand-crafted approaches by developing a three-step end-to-end process where the steps included (1) optical flow estimation, (2) classification/feature extraction, and (3) temporal localization. Mandal and Adu-Gyamfi (2020) used existing state-of-the-art object detection algorithms combined with object tracking algorithms to evaluate deep learning as an approach to traffic counting. They found that although the algorithms generally tended to overestimate the number of vehicles, certain combinations of algorithms (e.g., CenterNet (Duan et al. 2019), Detectron2 (Wu et al. 2019) or YOLOv4 (Bochkovskiy et al. 2020) combined with Deep SORT (Wojke et al. 2017)) produced promising results.

A comprehensive meta-analysis of research on the use of deep learning in transportation research evaluated how different methods, topic areas, datasets, and other factors influence performance metrics, notably accuracy (Sayer et al. 2011). They showed that a wide variety of deep learning approaches, including Deep Neural Networks (DNN), ConvNets, Long-Short Term Memory (LSTM), and Stacked Auto-Encoders (SAE) outperformed traditional methods in accuracy. Interestingly for our application, they also showed that accuracy for driver behavior prediction was, on average, almost 10 percentage points lower than when these methods are used for other topic areas (e.g., travel demand).

However, in the general research domain, there are few papers describing the use of computer vision algorithms to enhance the usability of research datasets such as NDD. While existing work shows the effectiveness of machine learning and ConvNets for information extraction from video and image data, our study is concerned specifically with enhancing Big Data utility in researc, in this case, in the transportation domain.

The research-dataset use case presents somewhat different requirements for the performance of computer vision...
algorithms compared to their use in ADAS and ADS. Nota-
bly, ADAS and ADS require real-time processing, whereas
algorithms used on research datasets do not need to operate
in real time. That said, research datasets are often quite large
and thus faster-than-real-time processing is needed if whole
datasets are to be processed in a reasonable timeframe.

While computer vision holds great promise for increasing
the usability of Big Data, labeling errors can propagate into
research results. In particular, we envision two potential use
cases. First, algorithm-based labels may be introduced into a
dataset such as SHRP2 and used without review. This would
maximize the quantity of data that are labeled, but labeling
errors would be incorporated into the research. Alternatively,
computer vision algorithms could be used to identify can-
didate video clips where a behavior might be present, and
human review would be used to verify labels. This approach
is still limited by human resource, but it could be far more
efficient than reviewing randomly-selected video clips, as
is the case for the baseline video in SHRP2. We note, how-
ever, that random selection of clips enables estimation of
the general prevalence of behaviors and thus has other sig-
nificant benefits, but for studies of particular behaviors, this
approach is inefficient.

To further the usability of video data in NDD, this paper
explores the possible use of 3D ConvNets to detect cell
phone behaviors on the part of drivers. We developed a 3D
ConvNet algorithm using NDD from the Integrated Vehi-
cle-Based Safety Systems (IVBSS) Field Operational Test
(FOT) (Varghese et al. 2020). This approach was selected
because it is currently at the leading edge of existing algo-
rithms. However, we recognize that new approaches are
constantly under development and will outperform this
method in a short timeframe. Our primary goal is to then
explore the potential use of this algorithm in two scenarios:
(1) where computer-vision-based labels are used directly in
analysis, and (2) where the algorithm’s labels are reviewed
by humans.

In the remainder of this paper, we present the develop-
ment and performance of our algorithm, which is trained to
label videos for cell-phone-related behaviors on the part of
drivers in an NDD. Following this, we evaluate the algo-
rithm’s performance in two simulated use cases and exam-
ine both the consequences of the performance of our algo-
rithm as well as better algorithm performance that might be
achieved in the future.

**Video Data**

For this analysis, we used videos from the Integrated Vehi-
cle-Based Safety System (IVBSS) Study (Varghese et al.
2020). The IVBSS FOT, conducted in 2009, was targeted at
observing drivers’ responses to multiple warning systems
in an integrated package. There were 108 subjects who each
drove one of eight identical vehicles for 40 days each. The
first 12 days consisted of baseline driving during which the
systems were turned off. This was followed by 28 days of
driving with the system.

The IVBSS data are ideal for this study because the entire
baseline dataset for all drivers was previously annotated for
cell phone behaviors for secondary data analysis purposes
(e.g. (Xiong et al. 2014)). Cell phone events were divided
into three categories: cell phone interaction, dialing a cell
phone, and talking on the cell phone. The remaining video
was labeled “no cell phone behavior”. Start and end frames
for each cell phone behavior were identified.

There are two camera views in the IVBSS videos: a face
camera that centers on the face and the upper shoulders of
the driver, and a cabin camera that includes a view of the
driver’s hands on or around the steering wheel. The face
video frame rate is 10 Hz and the cabin video frame rate is
2 Hz. Examples of these two views are shown in Fig. 1. It is
worth noting that because of the time of data collection for
IVBSS, cell phones are generally an older “flip” or “candy
bar” style rather than a smartphone style.

In our study, we used the face and cabin video views
from 96 drivers throughout the whole baseline period. These
drivers all had at least one cell phone behavior observed
and gave permission for their video to be used for viewing
and secondary data analysis. The original frame size was
720 \times 240$ pixels. We resized frames so that their widths were 256 pixels. We further cropped the frames at a randomly chosen location (same for all frames in a given video clip) to get $224 \times 224$ patches as inputs for training data augmentation. For the validation and test samples, we used a centered $224 \times 224$ crop for all clips.

We divided the videos into a training set (70%), a validation set (20%) and a test set (10%) based on driver ID. Each driver appeared in only one dataset to prevent the model from benefiting from driver-specific learning. In principle, an algorithm to be used on a specific dataset (e.g., IVBSS or SHRP2) could be allowed to learn the particular drivers in the dataset to improve accuracy on implementation, but for this study, we followed this more challenging protocol. The training set was used to train the models; the validation set was used to test the model for the purpose of stopping the training to avoid overfitting, and the test set was used for the evaluation of the final models.

From the 70% of drivers selected for the training dataset, we extracted 6934 video clips for the “uneven” training dataset. These included 1044 “dialing the phone”, 2853 “interacting with the phone”, 2399 “talking on the phone”, and 638 “no cell phone behavior”. The “no cell phone behavior” clips were much more common in the dataset than the cell phone activity clips, so while all cell phone behavior clips were used, only a small subsample of these baseline samples was used.

The baseline subsample was first sub-selected to provide controls for the cellphone-behavior clips such that the baseline clips occurred within the same trip and near in time to the cell phone behaviors (just before or just after). These clips were then down-sampled to make up close to 10% of the total clip sample. This training dataset contained cellphone-behavior clips in proportion to their appearance in the original dataset. As described in the results, the “dialing the phone” and “interacting with the phone” labels were later combined into a single “interacting” category with 3897 clips. We should note that other activities such as eating or drinking could occur in the baseline clips but were not labeled as such.

The “even” training dataset was constructed using only the three labels and was designed to have the same overall size but an even distribution of labels. This dataset consisted of 6933 clips with 2311 each of “interacting with the phone”, “talking on the phone” and “no cell phone behavior”. This dataset was constructed because unequal label occurrence tends to cause algorithms to be biased towards common categories, whereas an algorithm trained on an evenly distributed dataset should make errors that are symmetrical (i.e., equally likely to mislabel Category A with label B as to mislabel Category B with label A).

### Method

#### 3D Neural Network

Our method is composed of two steps: clip classification and temporal localization. The clip classification step predicts the following items for each clip: the category the clip belongs to and whether the start or the end point of a cell phone behavior falls into the clip. The temporal localization step recombines consecutive clips labeled with the same behavior to produce whole events of differing lengths with the same behavior label, including start and end points.

We chose this approach over a single-step approach because activity-classification methods at the clip level are more mature than identifying particular start/end points within the clip. This is analogous to the development of image classification first at the image level and later using bounding boxes within an image. With the two-step approach, we can change each part separately as the methods mature, rather than completely retraining the full model each time. In addition, with the 3D ConvNet approach, each clip must have the same dimensions, even though the behaviors may occur over any length of time. Thus, the first-stage model labels each standard-length clip and the second-stage model refines those results to more precise durations. As noted, these two algorithms can be separately improved to improve the performance of the overall model. This approach is similar to Gebert et al. (2019).

#### Data Preparation

We first extracted short clips from videos. This process was somewhat different for the training stage and the inference stage, as illustrated in Fig. 2. During the training stage, for each video, we extracted five kinds of clips: (1) clips only containing the start point of a behavior; (2) clips only containing the endpoint of a behavior; (3) clips containing both the start point and the end point; (4) clips between the start point and the end point; (5) clips containing no cell-phone-related behavior. Because of their different frame rates, we downsampled the face-view clips to 16 frames, so that we could stack the features from two feature extractors along with the dimension corresponding to the number of filters.

We then selected 8 s, or 16 frames, as the clip size. This size was selected to ensure that there was enough information in the clips for the ConvNet to use without producing clips that were too long for most behaviors. Events shorter than 8 s (e.g., at the start or end of a whole video) were discarded. For each clip, we provided five labels: class label, start inclusion label and end inclusion label, start regression parameter, end regression parameter.
During the inference stage, we divided each video into overlapping clips of 8 s (16 frames) length. A sliding window of the first 8 s was designated as the first clip. Then the window moved by 2 s (4 frames) to generate the next clip, and so on. In this way, every 2 s of video is part of up to four clips. This allowed for the possibility that behaviors might be shorter than 8 s and because an 8-s clip with mixed behaviors might be more difficult to label than one with only one behavior. Overlapping the clips allows the algorithm to evaluate the same two seconds of video in more than one context.

**Clip Classification**

The clip classification module was built based on the two-stream Inflated 3D ConvNet (I3D). The structure of the module is shown in Fig. 3. The detailed structure of I3D can be seen in Fig. 3 in Carreira and Zisserman (2017). In the original paper, the I3D model receives RGB inputs or flow inputs and predicts the activity class. We used all of the layers except the last “1 × 1 × 1 Conv” layer as a feature extractor. Images from both cabin view and face view were fed into two separate feature extractors. Then we stacked the extracted features and fed them into a new classifier layer. For this classifier layer, in addition to the behavior classification branch, we add two more branches. One is a clip inclusion branch, which predicts whether the start or end point of a cell phone behavior falls into the clip, and the other is a localization branch, which predicts the location of the start or the end point away from the center of the clip.

We trained the behavior classification branch with cross entropy loss (Eq. 1).

\[
\mathcal{L}_{cls}^i = -\log \left( \frac{\exp(p_{cls}^i[\text{class}_i])}{\sum_j^N \exp(p_{cls}^i[j])} \right)
\]

where \(\mathcal{L}_{cls}^i\) is the behavior classification loss of the \(i\)th clip, \(p_{cls}^i\) is a vector of class scores, \(\text{class}_i\) is the actual class label of the \(i\)th clip, and \(N\) is the total number of classes.

We trained the clip inclusion branch with binary cross-entropy loss (Eqs. 2, 3).

\[
bce(x, y) = -(y \log x + (1 - y) \log(1 - x))
\]

\[
\mathcal{L}_{inc}^i = \frac{1}{2} (bce(p_{st}^i, y_{st}^i) + bce(p_{end}^i, y_{end}^i))
\]

where \(\mathcal{L}_{inc}^i\) is the inclusion loss of the \(i\)th clip, \(p_{st}^i\) and \(p_{end}^i\) correspond to the predicted scores of the inclusion of the start point and end point in the \(i\)th clip, \(y_{st}^i\) and \(y_{end}^i\) are ground truth labels, which are set to 1 if the start (or end) point falls into the \(i\)th clip and are set to 0 otherwise.

We trained the localization branch with smooth \(\mathcal{L}_1\) loss (Eq. 4).
where $r_{st}^i$ and $r_{end}^i$ are ground truth regression parameters, which are calculated using Eq. (5), and $v_{st}^i$ and $v_{end}^i$ are predicted regression parameters.

\[
L_{\text{loc}}^i = y_{st}^i \cdot \text{smooth}_{L_1}(r_{st}^i - v_{st}^i) + y_{end}^i \cdot \text{smooth}_{L_1}(r_{end}^i - v_{end}^i)
\]  

(4)

where $n'$ is half of the clip length and $t_c^i$ is the center of the clip. Our final loss can be written as Eq. (6),

\[
L_{\text{total}} = L_{\text{cls}} + \lambda \mathbb{1}_{n' \geq 1} (L_{\text{inc}} + L_{\text{loc}})
\]  

(6)

where $\mathbb{1}_{n' \geq 1}$ is an indicator function that is equal to 1 when the ground truth action class label is greater than 1, and is equal to 0 otherwise. The class label for "no cell phone behavior" is always set to 0, so $\mathbb{1}_{n' \geq 1}$ is an indicator of whether a target action is present or not. $\lambda$ is a parameter controlling the weight of the class inclusion loss and the localization loss in the total loss and is set to 0.2 in our experiment.

### Temporal Localization

After the clips were classified, we used the outputs to identify longer activities with start and end points. This is also a two-step process, starting with rough aggregation, followed by refined temporal labeling. Figure 4 illustrates the two steps in the localization process. First, rough aggregation joins adjacent clips with the same behavior label and assigns a crude start and end from the beginning of the first aggregated clip to the end of the last aggregated clip. Refined temporal localization identifies a more precise start/end point within the aggregated clips because Event labels (e.g., Event1 and Event2) may be contained in (or partially contained in) several of the overlapping clips.

#### Rough Aggregation

The simplest way to aggregate is to merge adjacent clips of the same class. The following describes specific steps:

1. Aggregate adjacent clips with the same label into a longer clip and save the longer clips with the same label into a list.
2. Within each list, find the start frame and the end frame of each longer clip, if two adjacent clips are overlapped, they are further merged.
3. For longer clips in different lists, if a shorter clip is covered by a longer clip, only the longer clip is kept. If two clips have a small overlapping part at the boundary, to avoid the conflict, the overlapped part is discarded for both clips. If the overlapped part covers more than half of a clip, the covered clip is discarded.

#### Refined Temporal Localization

Simply merging all neighboring clips can achieve satisfactory performance, but we can also use the predicted scores for the inclusion branch and predicted regression parameters for the localization branch from the clip classification module to achieve better performance in endpoint localization. First, we borrowed the idea of non-maximum suppression

![Fig. 4 Temporal localization module](Image)
from object detection and perform it on \( p^i_{st} \) and \( p^i_{end} \) to select start and end clips. Specific steps are shown in Appendix A.

We used these start and end clips to refine the rough aggregation. In Step 2 of rough aggregation, if the boundary of a longer clip contains a start clip or an end clip, we utilized the corresponding predicted regression parameters \( v_{st} \) and \( v_{end} \) to further narrow the boundary using Eqs. (7) and (8).

\[
\text{start} = v_{st} \ast n' + c \quad (7)
\]
\[
\text{end} = v_{end} \ast n' + c \quad (8)
\]

Training

We built and trained the clip classification module with PyTorch. We first fine-tuned the pretrained I3D model on our videos with only the cabin view and class classification branch. Then we trained the whole model using stochastic gradient descent (SGD) with learning rate 1e−2, momentum 0.9, and weight decay 1e−5. We trained the model for a total of 30 epochs, lowering the learning rate by 0.1 after every 10 epochs. We used a batch size of 8.

Hardware

We trained and tested the 3D ConvNets on a single server. The server was equipped with a 12 GB Nvidia Titan X GPU (GP102-400-A1), a 2.40 GHz 2X Intel Xeon CPU (E5-2630 v3), 128 GB of memory (8 × 16 GB 2133 MT/S DDR4), and a 4 TB hard drive.

Use Case Simulation

To evaluate the use of our algorithm as well as potential future algorithms on research datasets, we set up a simulation of a potential use case. This simulation illustrates the use of the algorithm to label data without review. We assume that the performance of the algorithm is known for a small validation sample but that the algorithm is then run on an entire dataset such as SHRP2 and the results are used in subsequent analysis. The analysis we illustrate is an estimation of the crude odds ratio (OR) of crashing while texting vs. not texting.

The ground truth for our simulation is based on (Dingus et al. 2016), which used SHRP2 data to estimate ORs for a wide variety of secondary behaviors. They analyzed 905 crashes and 19,732 baseline (non-crash) driving epochs. Texting occurred in 1.9% baseline events and the reported OR for texting was 6.1. We used these values as the primary ground truth for our simulation, though other values were also explored.

For the simulations, we populated four cells of the 2 × 2 table of texting (yes/no) × crash (yes/no) as follows:

1. Select baseline sample size \( (n_b) \), crash sample size \( (n_c) \), true OR \( (\text{OR}_{true}) \), algorithm confusion matrix and texting prevalence \( (p_b) \).
2. Select the number of texting events in baseline driving by drawing from a binomial distribution with parameters \( n = n_b \) and \( p = p_b \).
3. Select the number of texting events in crashes by drawing from a binomial distribution with parameters \( n_c = \) crash sample size and \( p_c = \) probability of texting in the crash sample, where \( p_c \) is given with Eq. (9).

\[
p_c = \frac{p_b \ast \text{OR}_{true}/(1 - p_b)}{1 + p_b \ast \text{OR}_{true}/(1 - p_b)} \quad (9)
\]
4. For each cell in the 2 × 2 ground truth table created in steps 2 and 3, relabel events by drawing from a Binomial distribution with \( n = \) number of events in that cell and \( p = \) probability of mislabeling based on the confusion matrix.
5. Using the algorithm labels (both correct and incorrect), populate the algorithm’s 2 × 2 table.
6. Compute the OR for the ground truth sample and the algorithm-labeled sample.

Using this method, we ran 32 simulations made up of four baseline sample sizes (10,000; 50,000; 100,000; 500,000) by two crash sample sizes (1000; 10,000) by four confusion matrices. The confusion matrices were two observed from our algorithms, one developed on an unevenly distributed training sample (across classes) and one on an evenly distributed training sample, using only the “no activity” and “interacting/dialing” categories. In addition, we constructed two symmetrical confusion matrices, one with 95% accuracy and one with 99% accuracy for each class. We ran each simulation 5000 times.

To explore the effects of true OR and texting prevalence, we conducted a second set of simulations using a fixed sample size of 20,000 baselines and 1000 crashes (similar to SHRP2) and using the even 3D ConvNet confusion matrix. We crossed three ORs (0.5, 0.8, 3.0) with four prevalences (0.019, 0.2, 0.5, 0.75).

Results

Algorithm Performance

We first trained our model on the “uneven” dataset with 4 classes and 2 views. We observed the confusion matrix and found that it was difficult for the model to distinguish
“dialing the phone” from “interacting with the phone”. We then retrained the model on the same dataset with 3 classes. We also trained the model using the same method but with each of the individual views separately and 3 classes to identify the contributions of each view. Finally, we trained a fifth model on the “even” training dataset with three classes. All of these algorithms were developed using the same approach (but in some cases with feature extraction from only one view) and tested on the same (unequally distributed) test dataset.

### Activity Classification Accuracy

We first evaluated our model with the activity classification accuracy on those clips in the test dataset. As shown in Table 1, using both cabin view and face view achieved better classification accuracy compared to using just one view, indicating that the model uses information from both views to detect cell phone behaviors. Classifying clips with 3 classes also achieved better overall classification accuracy compared to classifying clips with 4 classes. Finally, the “even” training dataset produced nearly equivalent overall accuracy on the test data to the “uneven” training dataset, in spite of the test data also being unevenly distributed.

### Confusion Matrix for Activity Classification

We used confusion matrices to further visualize the performance of classification. From Fig. 5a, it is evident that the model did not distinguish the behavior of dialing the phone from interacting with the phone well. This led us to collapse dialing and interacting into one “interacting/dialing” category. From Fig. 5c, we observe that the model often misclassifies “no activity” into “talking on the phone” with only the cabin view. Comparing Fig. 5b, d, we can see that if we train the model on an unevenly distributed dataset, the model tends to misclassify the clip into “interacting with the phone” when it misclassifies a clip. That is, it learns to select the most common category more readily and thus produces asymmetrical confusion matrices.

### Start/End Prediction

We tested start/end inclusion branches on the test set and drew Receiver Operating Characteristic (ROC) and Precision-Recall (P-R) Curves, shown in Fig. 6. Curves for the start inclusion prediction are relatively good, but performance for end inclusion prediction is only marginally above chance.

### Temporal Prediction Accuracy

The final goal of our method is to predict cell phone behaviors temporally in a continuous video. To evaluate our method, we selected 71 videos from the validation and test datasets. Each video contained more than two cell phone behaviors. For each video, we produced several predicted chunks aggregated from short clips. For each chunk, we calculated temporal prediction accuracy as in Eq. (10).

$$\text{ACC}(C_i) = \frac{\sum_{j=1}^{N} L(\text{intersection}(GT_j, C_i))}{L(C_i)}$$

where $C_i$ is a predicted chunk, $GT_j$ is a ground truth chunk of the same class as $C_i$, intersection($GT_j, C_i$) is the overlapped part between two chunks, and $N$ is the total number of ground truth chunks of the same class. For this accuracy measure indicates, for example, what percentage of each chunk of video labeled “talking”, is, on average, actually talking.

As shown in Table 2, with rough aggregation, an average of 67% of predicted chunks labeled “interacting with the phone” and 80% of those labeled “talking on the phone” are correctly labeled. The statistics using refined aggregation with a threshold of 0.3 (see Appendix A for implementation details) are 79% and 94%, respectively. The use of start-and end-point prediction improves the temporal prediction accuracy slightly.

### Runtime

To calculate average runtime with our computing setup, we selected a batch size of 16 and processed a group of 500 videos using the 3D ConvNet trained on the evenly-distributed training dataset using both face and cabin views. We then divided the total runtime by the total duration of the videos to estimate the average runtime per second of video. Using our particular hardware setup, the runtime was 0.137 s per second of video, or 7.3× speed.

### Simulations

As shown in Fig. 5, the confusion matrix for the uneven 3D ConvNet had an error rate of 24% when the ground truth
was baseline and 6\% when the ground truth was texting. For the even 3D ConvNet confusion matrix, the corresponding values were 13\% and 10\%. These values were used in the simulation, along with the two constructed high-accuracy symmetrical matrices.

Table 3 shows the results of simulations using 1.9\% texting prevalence and 6.1 OR. For the even and uneven confusion matrices from our 3D ConvNet, the estimated ORs across all baseline and crash sample sizes averaged approximately 1.6 and 1.4, respectively. For the 95\% and 99\% confusion matrices, the corresponding values were 2.4 and 4.4. Changes in sample size had no appreciable effect on the mean estimates.

Standard deviations of the OR estimates were lowest for the even 3D ConvNet confusion matrix, followed by the uneven 3D ConvNet matrix, the 95\% matrix, and finally the 99\% confusion matrix. Standard deviation decreased slightly with increasing baseline sample size. However, increasing the crash sample size had a much larger effect on the standard deviation of the estimates, cutting them approximately in half across all confusion matrices.
In all cases, mean OR estimates were closer to 1 than ground truth. However, when prevalence was closer to 50%, mean OR estimates were closest to ground truth. Standard deviations were similar across levels of texting prevalence, though they were somewhat small the closer prevalence was 50%. True OR had a greater effect on standard deviation such that ORs farther from 1 produced higher standard deviations in the estimates.

Table 3 Mean and standard deviation of estimated OR as a function of baseline and crash sample size

| Confusion matrix | Baseline sample size | Crash sample size = 1000 | Crash sample size = 10,000 |
|------------------|----------------------|--------------------------|----------------------------|
| Even             | 10,000               | 1.593 0.132              | 1.589 0.060                 |
| Even             | 20,000               | 1.590 0.127              | 1.588 0.050                 |
| Even             | 100,000              | 1.590 0.124              | 1.587 0.041                 |
| Even             | 500,000              | 1.588 0.122              | 1.588 0.039                 |
| Uneven           | 10,000               | 1.352 0.098              | 1.351 0.042                 |
| Uneven           | 20,000               | 1.354 0.096              | 1.351 0.037                 |
| Uneven           | 100,000              | 1.353 0.094              | 1.351 0.031                 |
| Uneven           | 500,000              | 1.353 0.094              | 1.351 0.029                 |
| 95%              | 10,000               | 2.364 0.234              | 2.368 0.117                 |
| 95%              | 20,000               | 2.373 0.227              | 2.367 0.095                 |
| 95%              | 100,000              | 2.367 0.213              | 2.366 0.073                 |
| 95%              | 500,000              | 2.371 0.217              | 2.364 0.069                 |
| 99%              | 10,000               | 4.373 0.510              | 4.370 0.295                 |
| 99%              | 20,000               | 4.364 0.478              | 4.359 0.234                 |
| 99%              | 100,000              | 4.364 0.446              | 4.354 0.161                 |
| 99%              | 500,000              | 2.371 0.217              | 2.364 0.069                 |

True OR for all cases is 6.1

Table 4 Mean and standard deviation of estimated OR as a function of true OR, baseline and crash sample size, and texting prevalence

| True OR | Texting prevalence | Mean | Std |
|---------|--------------------|------|-----|
| 0.5     | 0.019              | 0.944| 0.090|
| 0.8     | 0.019              | 0.978| 0.090|
| 3       | 0.019              | 1.233| 0.103|
| 0.5     | 0.2                | 0.692| 0.054|
| 0.8     | 0.2                | 0.879| 0.065|
| 3       | 0.2                | 2.151| 0.139|
| 0.5     | 0.5                | 0.594| 0.040|
| 0.8     | 0.5                | 0.844| 0.055|
| 3       | 0.5                | 2.283| 0.161|
| 0.5     | 0.75               | 0.601| 0.039|
| 0.8     | 0.75               | 0.856| 0.060|
| 3       | 0.75               | 1.935| 0.166|

All simulations used the 3D ConvNet even confusion matrix with 20,000 baseline samples and 1000 SCE samples.

The results of the simulations varying true OR and behavior prevalence are shown in Table 4.
Discussion and Future Work

This work explores the development and use of a 3D ConvNet to identify instances of cell-phone-related behaviors by drivers in a large naturalistic driving dataset (NDD). In particular, we are interested in the potential for using such algorithms to label large amounts of unlabeled video present in NDDs such as the SHRP2 dataset (Hankey et al. 2016), which contains millions of hours of unlabeled video.

Using the IVBSS dataset (Varghese et al. 2020), which was available to us with labels, we developed a set of 3D ConvNet algorithms and then evaluated them in two use cases: with and without human review. The former ensures that he labels are correct, while the latter maximizes the amount of data that can be labeled by the algorithm. We also measured run time to estimate the relative performance of different versions of the algorithm in conjunction with review time on the part of human coders (in the with-review use case).

Our algorithm uses a two-step approach where the first step classifies clips by activity and whether or not a start point or an endpoint is present in the clip. The second step aggregates the clips and identifies the more specific start and end locations for each detected behavior. We adapted an existing feature set from the I3D model (Carreira and Zisserman 2017) for the first-step algorithm and developed a separate algorithm for the second step based on the results of the first.

Our two-step approach enables separate improvement of each step. In particular, behavior labeling is a better-developed capability in computer vision, meaning that technological advances could focus on the second step. Notably, approaches using attention (e.g., Rodriguez et al. 2020) or semantic segmentation (e.g., Xu et al. 2015) show particular promise in improving performance, especially in localizing behaviors. That said, Chen et al. (2021) showed that I3D accuracy improved by 4% when it was developed on the ResNet50 backbone rather than the InceptionV1 backbone, so we might be able to make similar gains in the first-step algorithm with only this simple strategy.

The baseline-driving portion of the IVBSS dataset was used for training, as it had previously been labeled for three cell phone behaviors (dialing, interacting, and talking) based on two camera views (cabin and face). However, the initial algorithm that was trained using four event categories did not distinguish well between the interacting and dialing categories. We noted that these two behaviors were labeled based on whether they were followed by the behavior of talking on the phone immediately. Because our model is just provided with short clips, it can lose this context information. This might be addressed in a future effort that would include some additional context (e.g., in the form of activity order and a secondary algorithm that distinguishes between dialing and interacting).

In our case, to address this issue, we combined the two confusable categories into a single interacting/dialing category, which improved the overall accuracy. However, in deciding to combine categories in the context of a specific research question such as the relationship between cell phone behaviors and driving, it would be important to consider whether the categories are likely to affect safety equally. That is, by combining categories, the algorithm-labeled data will combine two underlying behaviors into a single category and subsequent analysis would represent the combination. In the human-review use case, the categories could be disaggregated by the human coder, but not in the no-review use case.

We developed three additional algorithms as well using the same method. The first was trained on an equal-sized training sample but with even category representation rather than the natural representation in the dataset. This produced an algorithm with the same overall accuracy of 79% on the test as the uneven-distribution three-category algorithm but an even confusion matrix that did not favor one category over another. We used this algorithm for further explorations. Two other algorithms were built using only cabin view and only face view to explore simpler, faster options and determine how critical it was to use both.

While the algorithm does a reasonable job at identifying the activity for eight-second clips, we were interested in identifying longer instances of cellphone-related activities. This was achieved by the second-step aggregation algorithm based on the predicted class label, predicted start and end scores associated with regression parameters. However, predicted endpoint inclusion showed poor performance, in particular. As a result, we found that rough aggregation with only class labels can achieve satisfactory performance, whereas the more complex refined aggregation had only marginally better performance.

We considered two potential use cases for computer vision (CV) algorithms in the context of research datasets such as SHRP2: (1) labeling a very large number of events with the algorithm and then using the labels in subsequent analysis, and (2) using the labels to identify promising events (i.e., events with a high likelihood of containing the behavior of interest) for human review.

We simulated the first use case under a variety of conditions to estimate the crash OR for texting while driving. Our ground truth was primarily based on values from the SHRP2 dataset, but we simulated other ground truth values as well. On the face of it, this is the ideal use case for CV algorithms—the behavior in question is relatively rare (1.9% of baseline events), such that out of 20,000 baseline events, there are fewer than 400 cases of texting. However, our simulations show overwhelmingly that even an extremely
accurate algorithm (99% correct) produces enough mislabeled events to reduce the estimated OR from 6.1 to 4.4, a 28% reduction in the estimate.

The increase in baseline sample size enabled by the CV algorithm improved the standard deviation of the estimates. However, increases in labeled crash sample size resulted in the greatest improvements, cutting standard deviations in half. Interestingly, the ground truth estimates had the largest standard deviations because of the small numbers of texting events in crashes. OR variance is a function of the reciprocal of each cell’s sample size, so the size of the smallest cell will have the most influence on variance. Because there are so many more non-texting events than texting events, mislabeling is much more likely to incorrectly add texting events to the sample than incorrectly add non-texting events to the sample (in either crash or baseline samples). Thus, for the CV-labeled datasets, including very accurate ones, there are more events added to the smallest cells of the table, and thus lower standard deviations around OR estimates.

Our simulation also showed that the prevalence of the activity has a big effect on the accuracy of the estimates when labeled by an imperfect algorithm. For example, when texting prevalence is 50%, the even 3D ConvNet produced OR estimates that were closest to the ground truth, compared to any other prevalence. However, any behavior that occurs 50% of the time is easy to find by using human coders and obtaining actual ground truth labels is relatively easy.

Ironically, the best use case for CV algorithms is in labeling rare events so that sample sizes can be much larger than is feasible with human coding. Thus, in general, using CV algorithms to label data for analytical use without review will likely end up producing nullward-biased estimates to a degree that probably makes this use case inappropriate, at least under the conditions we simulated. We note, however, that this type of pattern was also found in Guo et al. (2010) when near-crashes were included with crashes in the outcome sample. In that case, near-crashes (rather than incorrect labels) added noise and resulted in nullward-biased OR estimates but smaller confidence intervals (because of the increase in crash sample size, as was seen here). The authors argued that such estimates can be seen as a “conservative risk estimate” and a lower bound on the true OR. The same argument holds in the use case simulated here. However, because of the importance of OR estimates in determining the prioritization of interventions as well as messaging, nullward-biased estimates could result in downplaying the importance of certain risk factors, especially when compared to estimates based on other methods. Even the OR ranking may not be preserved since both prevalence and true OR have an effect on the estimator bias in our use case.

In contrast, our algorithm and other CV algorithms could be readily used in the context of searching for specific events with human review. For example, an analytical goal might be to create an algorithm that uses kinematic data to detect when a driver is texting. For that analysis, a sample of texting events and a matched sample of non-texting events could be used to develop the algorithm. If a sample of 500 texting and 500 non-texting events were desired, a human would need to review 26,316 baseline events. However, with the even 3D ConvNet used to find candidate events for review, the human would only need to review 555 cases that were labeled as texting by the algorithm. Using our hardware, the two-view algorithm would take about 22.5 h to find the candidate clips. In exchange, human review time would be reduced by 98%. Also in this case, the rough aggregation approach will generally produce longer clips, but the main benefit of the algorithm is in finding the clips within the larger dataset in general. Having to review a slightly longer clip to find the start and end by hand is a small cost relative to viewing randomly selected videos in search of specific behaviors. Thus, the refined aggregation component of our algorithm does not add benefit unless its performance can be substantially improved.

We emphasized the symmetrical confusion matrix of the even-dataset algorithm, particularly with the no-review use case in mind. However, with review, in some cases it could be beneficial to prefer an uneven confusion matrix and/or a different arrangement of false positives and false negatives. False positives and negatives can be traded off by changing the label criterion at the decision layer of the algorithm. The optimal arrangement will depend on the research problem. For example, if the goal is to find a particular number of events involving a given behavior and there is no reason to believe that behaviors the algorithm finds are unique in any way, then lowering the false positive rate will increase the total run time of the algorithm but decrease the total review time for the humans. If, on the other hand, all (or nearly all) of a particular behavior must be found in a set of videos, it will be important to minimize the false-negative rate, which will come at the expense of false positives. For rare events, given the substantial benefit of using the algorithm over searching through video at random, even a high false-positive rate can still be more efficient. However, it is important to evaluate the specific research problem to identify whether computer vision is going to be helpful at all, and if so, how best to target the algorithm’s performance.

As a key example, algorithm-based labeling, even with a review, would not replace the effort put into coding more than 32,000 baseline cases in the SHRP2 case-control sample. That is, with a case-control (or case-crossover) sample, sampling is done on the basis of the outcome (crash/no crash) rather than the behavior, and all of the samples need to be labeled accurately, thus requiring human review of all of them. This type of sampling is crucial to crash-OR estimation in the traffic safety context. The human-supervised approach described above is ideally suited to sampling...
approaches based on the behavior of interest, which would typically be used to evaluate outcomes such as driving kinematics (e.g., lane-keeping, speed-keeping) and other behaviors (driving or otherwise) associated with the behavior of interest (i.e., more common outcomes than crashes and near-crashes).

In general, we see the algorithmic labeling of video in SHRP2 and other research datasets as promising. However, it may need to be limited to certain use cases. Based on our simulations, even an extremely accurate algorithm will bias OR estimates towards the null by a large enough degree to make the approach of simply using labels in a basic analysis infeasible. However, the 3D ConvNet we developed is good enough to drastically improve the efficiency of searching for rare events in conjunction with human review. Improvements to activity recognition models are being made at a very fast pace in computer science, and these gains can be used to further improve search efficiency. However, until human verification can be eliminated, incremental improvements in accuracy will produce incremental gains in search efficiency. Future research should investigate whether the types of events found by a CV algorithm are different from those not found (possibly influencing the results of analyses done on the resulting events). Moreover, further developments in computer vision will lead to even better algorithms that can be used for this purpose. While this approach is not yet fully scalable because of the need for human review, it could greatly increase the value of data produced by that review process. In addition, there may be statistical approaches to de-biasing estimates given a known confusion matrix, which could help make the fully scalable approach possible in the future.
Appendix A: Detailed Algorithm for Start and End Clip Selection

Algorithm 1 NMS algorithm for $p^i_{st}$

Input: a list of $p^i_{st}$ and a threshold $\theta$

1: Sort $p^i_{st}$ from highest to lowest and get a list of sorted indexes $indexes$
2: $selected = []$
3: $i \leftarrow 0$
4: $L \leftarrow \text{len}(p_{st})$
5: $visited = [0] \ast L$
6: while $i < L$ do
7:     $idx \leftarrow indexes[i]$
8:     if $p^i_{st} < \theta$ then
9:         break
10:     end if
11:     if $visited^{idx} == 1$ then
12:         $i \leftarrow i + 1$
13:     else
14:         if $p^i_{st} >= p^{\text{max}(0, idx-1)}_{st} >= \theta \& p^i_{st} >= p^{\text{min}(L-1, idx+1)}_{st} >= \theta$ then
15:             $visited^{idx} = 1$
16:             $visited^{idx-1} = 1$
17:             $visited^{idx+1} = 1$
18:             $selected.append(idx)$
19:         else
20:             $visited^{idx} = 1$
21:             $visited^{idx-1} = 1$
22:             $visited^{idx+1} = 1$
23:         end if
24:     end if
25:     $i \leftarrow i + 1$
26: end while


Acknowledgements The authors would like to thank Alex Cao for his help in obtaining data and debugging code. This material is based upon work supported by the Federal Highway Administration under contract number 69JJ319000009. Any options, findings, and conclusions or recommendations expressed in this publication are those of the authors and do not necessarily reflect the views of the Federal Highway Administration.

Declarations

Conflict of interest On behalf of all authors, Carol Flannagan states that there is no conflict of interest.

References

Bao S, Wu L, Yu B, Sayer JR (2020) An examination of teen drivers’ car-following behavior under naturalistic driving conditions: with and without an advanced driving assistance system. Accid Anal Prev 147:105762

Bharadwaj N, Edara P, Sun C (2019) Risk factors in work zone safety events: a naturalistic driving study analysis. Transp Res Rec 2673(1):379–387

Bochkovskyi A, Wang CY, Liao HYM (2020) Yolov4: optimal speed and accuracy of object detection. arXiv:2004.10934 (arXiv preprint)

Carreira J, Zisserman A (2017) Quo vadis, action recognition? A new model and the kinetics dataset. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 6299–6308

Chao YW, Vijayanarasimhan S, Seybold B, Ross DA, Deng J, Sukthankar R (2018) Rethinking the faster r-cnn architecture for temporal action localization. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 1130–1139

Chen CFR, Panda R, Ramakrishnan K, Feris R, Cohn J, Oliva A, Fan Q (2021) Deep analysis of cnn-based spatio-temporal representations for action recognition. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp 6165–6175

Ciregan D, Meier U, Schmidhuber J (2012) Multi-column deep neural networks for image classification. In: 2012 IEEE conference on computer vision and pattern recognition. IEEE, pp 3642–3649

Dingus TA, Guo F, Lee S, Antin JF, Perez M, Buchanan-King M, Hankey J (2016) Driver crash risk factors and prevalence evaluation using naturalistic driving data. Proc Natl Acad Sci 113(10):2636–2641

Duan K, Bai S, Xie L, Qi H, Huang Q, Tian Q (2019) Centernet: keypoint triplets for object detection. In: Proceedings of the IEEE/CVF international conference on computer vision, pp 6569–6578

Feichtenhofer C, Fan H, Malik J, He K (2019) Slowfast networks for video recognition. In: Proceedings of the IEEE/CVF international conference on computer vision, pp 6202–6211

Gebert P, Roitberg A, Haurilet M, Stiefelhagen R (2019) End-to-end prediction of driver intention using 3d convolutional neural networks. In: 2019 IEEE intelligent vehicles symposium (IV), pp 969–974. IEEE

Gleason J, Schwarz S, Ranjan R, Castillo CD, Chen JC, Challappa R (2020) Activity detection in untrimmed videos using chunk-based classifiers. In: Proceedings of the IEEE/CVF winter conference on applications of computer vision workshops, pp 107–116

Guo F, Klauser SG, Hankey JM, Dingus TA (2010) Near crashes as surrogate for naturalistic driving studies. Transp Res Rec 2147(1):66–74

Hankey JM, Perez MA, McClafferty JA (2016) Description of the SHRP 2 naturalistic database and the crash, near-crash, and baseline data sets. Virginia Tech Transportation Institute, Virginia

Karpathy A, Toderici G, Shetty S, Leung T, Sukthankar R, Fei-Fei L (2014) Large-scale video classification with convolutional neural networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 1725–1732

Mandal V, Adu-Gyamfi Y (2020) Object detection and tracking algorithms for vehicle counting: a comparative analysis. J Big Data Anal Transp 2(3):251–261

National Highway Traffic Safety Administration (2018) Fatality analysis reporting system analytical users manual 1975–2017. United States Department of Transportation

Precht L, Keinath A, Krems JF (2017) Identifying effects of driving and secondary task demands, passenger presence, and driver characteristics on driving errors and traffic violations—using naturalistic driving data segments preceding both safety critical events and matched baselines. Transport Res F Traffic Psychol Behav 51:103–144

Radja GA (2016) National automotive sampling system-crashworthiness data system, 2015 analytical user’s manual (No. DOT HS 812 321)

Rodriguez C, Marrrese-Taylor E, Saleh FS, Li H, Gould S (2020) Proposal-free temporal moment localization of a natural-language query in video using guided attention. In: Proceedings of the IEEE/CVF winter conference on applications of computer vision, pp 2464–2473

Sayer JR, Bogard SE, Buonarosa ML, Leblanc DJ, Funkhouser DS, Bao S, Winker CB (2011) Integrated vehicle-based safety systems light-vehicle field operational test key findings report [ITS JPO] (No. FHWA-JPO-11-082). United States. Joint Program Office for Intelligent Transportation Systems

Shou Z, Wang D, Chang SF (2016) Temporal action localization in untrimmed videos via multi-stage cnns. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 1049–1058

Szegedy C, Liu W, Jia Y, Sermanet P, Reed S, Anguelov D, Rabinovich A (2015) Going deeper with convolutions. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 1–9

Tran D, Bourdev L, Fergus R, Torresani L, Paluri M (2015) Learning spatiotemporal features with 3d convolutional networks. In: Proceedings of the IEEE international conference on computer vision, pp 4489–4497

Varghese V, Chikaraishi M, Urata J (2020) Deep learning in transport studies: a meta-analysis on the prediction accuracy. J Big Data Anal Transport 20:1–22

Wojke N, Bewley A, Paulus D (2017) Simple online and realtime tracking with a deep association metric. In: 2017 IEEE international conference on image processing (ICIP), pp 3645–3649. IEEE

Wu Y, Kirillov A, Massa F, Lo WY, Girshick R (2019) Detectron2

Xiong H, Bao S, Sayer JR (2014) Factors affecting drivers’ cell phone use behavior; implications from a naturalistic study. Transp Res Rec 2434(1):72–79

Xu C, Hsieh SH, Xiong C, Corso JJ (2015) Can humans fly? Action understanding with multiple classes of actors. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 2264–2273

Zeng R, Huang W, Tan M, Rong Y, Zhao P, Huang J, Gan C (2019) Graph convolutional networks for temporal action localization. In: Proceedings of the IEEE/CVF international conference on computer vision, pp 7094–7103

Zhou B, Andonian A, Oliva A, Girshick R (2018) Rethinking the faster r-cnn architecture for temporal action localization with a natural-language query in video using guided attention. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 2464–2473

Publisher’s Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.