Now You *Shake* Me: Towards Automatic 4D Cinema

Yuhao Zhou[1] Makarand Tapaswi[1,2] Sanja Fidler[1,2]
[1]University of Toronto [2]Vector Institute
{henryzhou, makarand, fidler}@cs.toronto.edu
http://www.cs.toronto.edu/~henryzhou/movie4d/

Abstract

We are interested in enabling automatic 4D cinema by parsing physical and special effects from untrimmed movies. These include effects such as physical interactions*,* water splashing*,* light*, and* shaking*, and are grounded to either a character in the scene or the camera. We collect a new dataset referred to as the* Movie4D dataset *which annotates over 9K effects in 63 movies. We propose a Conditional Random Field model atop a neural network that brings together visual and audio information, as well as semantics in the form of person tracks. Our model further exploits correlations of effects between different characters in the clip as well as across movie threads. We propose effect detection and classification as two tasks, and present results along with ablation studies on our dataset, paving the way towards 4D cinema in everyone's homes.*

1. Introduction

Fast progress in deep learning together with large amounts of labeled data has enabled significant progress in tasks such as image tagging [16], object detection [14], action recognition [10], and image captioning [43]. Neural networks have also proven themselves as surprisingly good artists by repainting images in different styles [12], writing poems [18], and synthesizing music [5, 42]. With the emerging market of virtual reality, simulated roller-coasters, and infotainment, machines might also help us reach a new level of the entertainment experience.

In 4D cinema, the audience is taken to a wild ride through the movie: their seats shake when a high speed car chase unrolls on the screen, water splashes on their faces when a boat cuts through the Perfect Storm, and smoke veils around them when Clint Eastwood lights up yet another cigarette. While entertaining for the audience, such effects are not so fun to annotate for the movie creators. They are time consuming and require careful annotation of what physical phenomena is occurring at every time instant in the film [23]. The strength of the effect, and possibly direction is also important to faithfully recreate the fast-paced dynamic world for the audience.

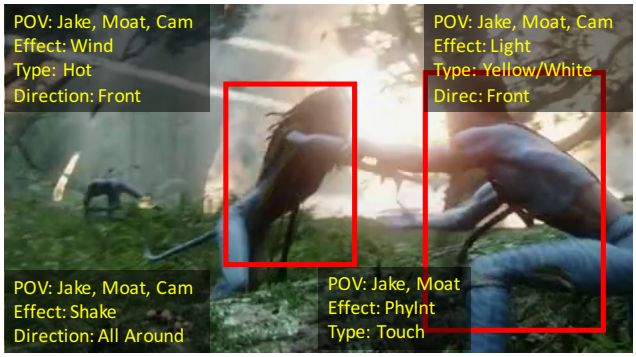

Figure 1: Movie4D: We aim to predict effects as experienced by characters and camera, their duration, and all details such as intensity, type, and direction, automatically in movies.

In this work, we take a step towards promoting creation of 4D cinema by automatically parsing detailed physical effects in movies. In particular, given a streaming video, we aim to detect both which effect is being applied to each of the characters in the scene (or camera), as well as to predict accompanying details such as the intensity of each effect, its duration and possibly direction. While inferring effects from videos has clear significance for the entertainment industry, we believe it also has value for building intelligent robots in the future. When faced with the real world, robots will need to foresee physical forces based on current visual or audio information in order to cope with them.

Due to unavailability of an existing dataset of this form, we first collect the *Movie4D dataset* containing rich annotations of physical effects in feature films (Fig. 1). Our dataset contains 9286 effect annotations with time stamps and accompanying details. The effects are also grounded to either the camera's point of view, or to a particular character in the clip. These effects take place in various scenes, ranging from heroic battlefronts to everyday lives.

We propose a model to parse effects from untrimmed videos and ground them to characters' tracks. We formalize the task as performing inference in a Conditional Ran-

dom Field, that exploits potentials extracted from multiple modalities (visual, audio, and semantic) via neural networks. Our model further profits from correlations between effects applied to different characters in the same clip (such as one character being exposed to a *shake* likely means that the other character experiences the same effect), as well as across clips (some effects like *water splashing* are long in duration). We showcase the model through ablation studies, and point to challenges of the task.

Our code and data will be released[1] in order to inspire more research in bringing 4D cinema to everyone's homes.

2. Related Work

Video analysis, especially movies and TV series have several research directions. Among them, some of the most popular tasks are automatic person identification [4, 7, 9, 27], pose estimation [8], describing short video clips using one sentence [29] or aligning videos with plots [38] and books [39, 47]. Video-based question-answering is also growing in popularity, and among these MovieQA [40] and PororoQA [19] are based on movies and TV series.

Actions and interactions have also been studied in the context of movies. Hollywood2 [21] aims at predicting a few action classes given short clips, while human interactions such as hugging, kissing, are studied in [26]. A few approaches aim at finding people looking at each other [24]. With deep learning, and the need for larger datasets, action recognition (not necessarily in movies) has grown via ActivityNet [17], the THUMOS challenge and related UCF dataset [36]. Moving away from classifying actions given a segmented clip, there is a drive to detect and classify actions in longer untrimmed videos.

In the related domain of audio analysis, AudioSet [13] is a large collection of audio events that range from human and animal sounds, musical instruments, to everyday environmental sounds. A recent audio-video dataset Flickr-SoundNet [3] has enabled training audio-visual models in an unsupervised manner [2, 3]. Movie effects are audio-visual too, and we exploit these modalities in our models.

4D effects. Over the years, movie budgets have increased and facilitated use of dazzling special effects [1]. In this paper, we propose classification and detection of such effects and physical interactions in movies, as experienced by the camera and characters. Inspired by semantic role labeling for images that requires to predict the verb and the corresponding role-noun pairs (imSitu [44]), our effect annotations (*e.g.* wind) come with a variety of details that determine the intensity (*e.g.* strong), direction (*e.g.* from left) and even sub-types (*e.g.* cold wind).

In the past, several attempts have been made to predict a similar range of effects, however, in different, and importantly, isolated contexts. Classification of weather conditions has been analyzed for driver-assistance [30], while detecting water [25], and especially rain [11] is of special interest. In the context of fire safety, work by [6] aims at detecting smoke.

There is work on real rendering of audio-visual signals to sensory devices and chairs. [22] aims to translate audio signals (movies, hand-held games, *etc.*) onto a vibro-tactile sensory device, with [34] focusing on rendering gunshots. Probably the most related to our work, [23] analyzes 10 real 4D films with about 2.2K effects and manually groups them based on viewer experience (*e.g.* motion, vibration). As motion forms a large chunk of 4D effects, [23] employs optical flow along with Kalman filtering to use video motion to control the chair. Our work is different on two key fronts: (i) we use audio-visual information to detect and classify effects in movies, and even those that were not originally made for 4D; and (ii) our dataset annotations and model reason about which characters experience the effects. We also collect a significantly larger dataset with 63 movies and over 9.2K effect annotations.

	TRAIN	VAL	TEST	TOTAL
Movies	50	7	6	63
# Effects	7283	816	1187	9286
# Instantaneous effects	1492	115	268	1875
Effect avg. duration	5.9	6.8	6.6	6.1
Per Movie				
Avg. # video clips	11.9	12.1	13	12.1
Avg. # characters	7.2	6.4	7.5	7.1
Per Clip				
Avg. # shots	79.9	79.2	78.9	79.7
Avg. # threads	9.6	9.9	8.3	9.5
Avg. # person tracks	56.8	62.5	61.2	57.9

Table 1: Summary of the Movie 4D dataset.

3. Movie4D Dataset

We first introduce our dataset, by describing the annotation process, statistics, and proposed tasks. In the next section, we propose a model that aims to solve these tasks.

We build the Movie4D dataset to analyze the detection and detailed parsing of effects in films. Our dataset consists of 654 five-minute clips obtained from 63 movies. Most of our movie genres are action/adventure, and sci-fi as they typically contain the highest number as well as diversity of effects. However, Movie4D also features films from drama, comedy, and romance, that could be used as a proxy to understanding effects in the real world.

3.1. Annotating Effects

We are interested in annotating effects in video clips, including the start and end timestamps, effect type, and effect-specific details such as intensity, direction, and sub-type.

[1] www.cs.toronto.edu/~henryzhou/movie4d/

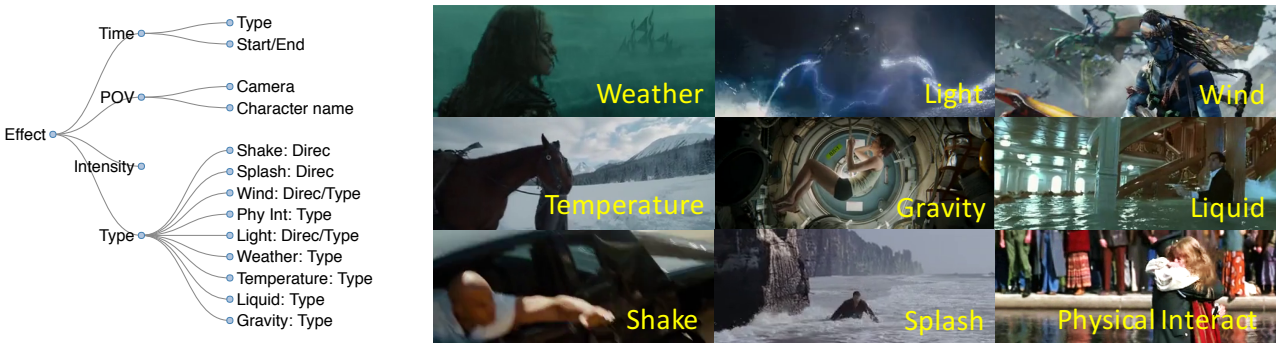

Figure 2: **Left**: The tree illustrates the structure of an effect annotation. Each effect consists of duration, point of view, intensity, and type, with additional effect-specific details. **Right**: Example frames corresponding to the different effect types in our dataset. While some effects can be easily detected from images alone, many of them require multiple sources of information (image, motion, audio and semantic cues).

We develop a web annotation interface, where the annotator is allowed to select a movie, and then browse through a set of clips. As preprocessing, we split the movie into 5-min clips and provide a few to the annotator. After selecting a clip, the annotators are presented with the video, and a dynamic questionnaire interface that allows them to add new effects (details in supp. material). While browsing (watching) the clip, the annotator first adds the type of the effect, such as *shake* or *wind*, and a few mandatory fields common to all effects: (i) start and end time; (ii) intensity; and (iii) point of view (POV). Effects that have a duration less than one second are referred to as *instantaneous*. For each effect type, we present additional effect-specific fields which the annotator can fill. Fig. 2 provides the list of effect types, details, as well as examples. We explain our effect annotations in more detail below.

Intensity. We provide three options: *mild*, *medium*, and *strong*. Mild effects are common on a daily basis, such as a light blow of wind. Medium effects are significantly more noticeable, but are still acceptable to most people. An example would be stronger wind or a pouring rain. Strong effects are not common in normal life and may cause pain or discomfort, such as severe shaking due to explosion or earthquakes, or strong winds due to hurricanes.

POV denotes the subject that experiences the interaction within the video. We allow annotators to choose from a cast of main characters that we provide (seven per movie on average), as well as the camera. A camera POV indicates that the effect is applied to the cameraman or the observer of a scene from a first-person perspective.

Our annotators were hired from the freelance website Upwork, that facilitates interaction through a message board. We trained the annotators for roughly two hours and gave them constant feedback for the first two movies they annotated, in order to ensure consistency. Each annotator was asked to annotate a full movie. The annotators were paid by the hour.

1. Shake. The POV experiences continuous or sudden spatial motion. Shake has detail *direction* with options: left-right, front-back, up-down, all-around, and other.

2. Splash is caused by water (or any liquid) splashing onto the subject. Splash has detail *direction* with options: front, back, left, right, top, bottom, all-around, and other.

3. Wind is a result of natural weather phenomenon or artificial manipulations by machines (such as standing on a fast moving boat). The detail *direction* for wind has the same options as that of *splash*. Wind also has detail *type* with the following options: hot, cold, and normal.

4. Physical Interactions are effects defined between two characters, such as fighting. Physical interactions have detail *type* with the following options: hit, pinch, twist, string, rub, drag, massage, impact, gunshot, and other. We also ask our annotators to select the source and target of the physical interaction (*e.g.* 'A drags B', where A and B are characters from the cast).

5. Light effects are defined only for those that involve an artificial light source. The detail *direction* has the same options as *splash* and *wind*. Light effects also have detail *type* with different colors: white, red, yellow, orange, green, blue, purple, and other.

6. Weather effects are usually subject to both the camera and all characters in the scene. Weather has detail *type* and comprises: extremely-sunny, rain, snow, fog, wind, snowstorm, other.

7. Temperature is annotated as an effect when the ambient temperature is not normal. It has *type* high and low.

8. Liquid Surrounding indicates that a large portion of the character/camera is submerged in water or other liquids. The detail *type* tells us the type of liquid: water, or other.

9. Gravity effects are annotated only when POV experiences unnatural low/high gravity forces. They have the detail *type* with options: high gravity (acceleration), low gravity, and zero gravity.

Person tracks. As our effect reasoning requires determining POV, we ground this information to character tracks in the clips. In particular, we perform person tracking in every

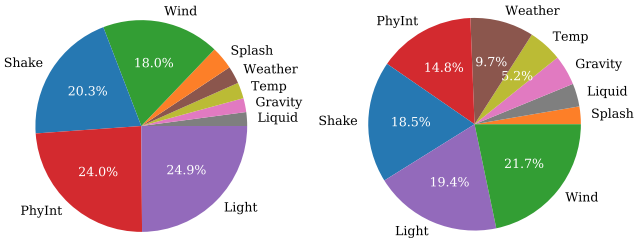

Figure 3: Effect distribution by count (left) and duration (right).

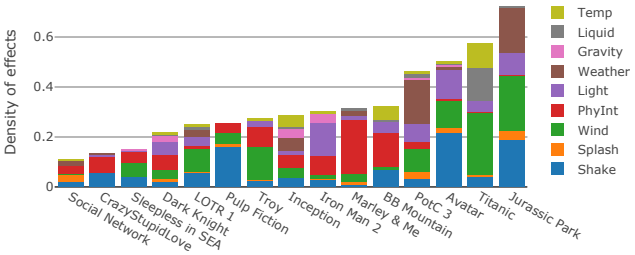

Figure 4: Density of effects in several movies. In general, action/adventure/sci-fi movies tend to have more effects, while drama/romance have a lower density and variety.

shot of the clip, and ask the annotators to assign a character name from the cast list to each track. We use person detections from the YOLO9000 object detector [28], and combine subsequent detections into person tracks based on spatial overlap. 41.4% of our tracks correspond to main characters and 30.3% to background characters. We obtain several false positives due to detections spaced at 3 frames per second.

3.2. Dataset Statistics

We collected a total of 9286 annotations from 654 clips each 5-min long. Table 1 provides a summary of different features about our data, along with train-val-test splits. We create splits with disjoint movies and achieve a balance between movie metadata and effect annotations.

The distribution of the number of effect classes and their durations is presented in Fig. 3, and we see that *light* and *wind* are dominant effects. Fig. 4 presents the density of effects in a few example movies. We select 15 movies from various genres and compute the effect duration within them. Dramatic effects such as *shake*, *liquid surrounding*, and *wind* are more pronounced in action-packed movies such as adventure and sci-fi.

In Fig. 5, we show the t-SNE [41] visualization of the top 40 characters based on amount of time spent with effects. The effect duration is used as a feature for clustering. We observe that characters from sci-fi movies such as *Interstellar*, *Gravity*, and *Iron Man* are grouped together at the bottom (due to the zero-gravity floating and flying). In contrast, characters from adventure/action films such as *Lord of the Rings* and *Hunger Games* that experience natural phenomenon (*wind*, *weather*) are grouped at the top.

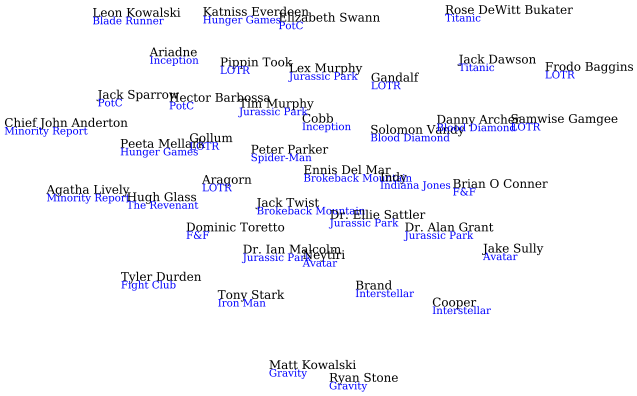

Figure 5: tSNE plot of characters (grouped using effect duration). Notice the clustering of characters by themes such as outer space (bottom), fighting (top left), and adventure (top right).

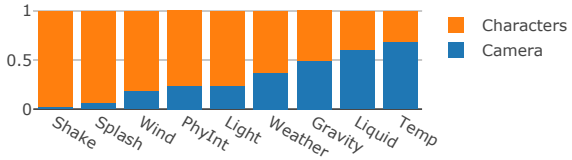

Figure 6: Ratios of camera and character POV annotations.

Fig. 6 presents the ratio of effects applied to characters or camera. Effects due to environmental conditions apply more frequently to the camera. On the other hand, *shake* and *splash* are very human-centric.

4. Effect Parsing in Videos

We propose models for effect detection and recognition. We first describe some preprocessing and introduce notation. Then, we present the neural architecture that is used to extract various features and perform classification. This acts as a baseline for our tasks. The classifier outputs are exploited as unary potentials in a Conditional Random Field (CRF) that performs joint reasoning about the effects within and across movie shots and threads. We address both *trimmed* effect recognition, and the more challenging *untrimmed* effect detection and parsing.

4.1. Video preprocessing

Careful consideration of shot boundaries and threading is important in order to compute features that are meaningful, as well as to exploit scene and filming priors. For example, *wind* effect typically applies to all characters as well as the camera in a shot, and possibly spans multiple neighboring shots. Similarly, if one character experiences a physical interaction it is highly likely that another character should exist and also undergo a physical interaction within the same shot or even thread.

Shots. Given a (5-min) clip from our dataset, we first detect shot boundaries using the motion-compensated difference between two consecutive frames [45].

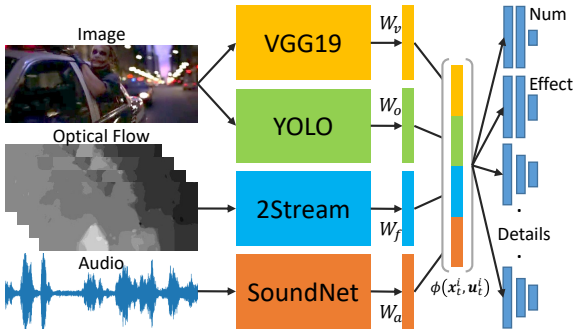

Figure 7: Neural network with pre-trained feature extractors.

Sub-Shots. Shots longer than 3s are further divided into *sub-shots* capped at a maximum duration of 3s. We adopt these as our primary unit of analysis. While the average shot duration is 3.65s, some shots in our movies are longer than 10s. Using sub-shots (instead of shots) reduces the noise from neighboring non-effect frames, while still providing enough audio-visual content to retain relevant effect information. Note that shots that are less than 3s are equivalent to their sub-shots. On average, our sub-shots are 2.49s long, and are assigned an effect label if they have a 10% or greater overlap with the effect annotation.

Threads. We *thread* shots taken from the same camera angle and viewpoint using SIFT-based matching [37].

4.2. Neural Architecture

Effects in movies are dominated by their audio-visual nature. In a video clip, we denote the audio content of a sub-shot as $\mathbf{u}_t$ and visual content as $\mathbf{x}_t$. We use $i = 1, \ldots, |\mathbf{x}_t|$ to index frames within the sub-shot, *i.e.* $\mathbf{x}_t^i$ denotes the visual feature of frame i in the sub-shot. We extract several features using networks pre-trained on different tasks:

(i) *Visual* features are the core of our model and intuitively are useful to detect all effect types. Within each second, we sub-sample three frames, and extract features from the `pool5` layer of the VGG19 model [35] pre-trained on ImageNet [31]. As a sub-shot is capped at 3s, it corresponds to a maximum of 9 visual frames and features. We denote each frame's representation as $\mathbf{v}_t^i = \phi_v(\mathbf{x}_t)$ and mean pool across the spatial grid to obtain $\mathbf{v}_t^i \in \mathbb{R}^{512}$. Optionally, we adopt a two-layer MLP (512-128-1) to compute self-attention weights to exploit spatial features (on the 7×7 grid) and replace the mean pool by a weighted average.

(ii) *Optical Flow* features form a visual representation that encode motion and are useful to detect effects such as *wind, splash, shake, etc*. We use the temporal stream of the two-stream network [10] trained for action recognition, and encode a stack of 10 optical flow images for each visual frame to obtain $\mathbf{f}_t^i = \phi_f(\mathbf{x}_t), \mathbf{f}_t^i \in \mathbb{R}^{512}$. Similar to *Visual* features, self-attention weights are used.

(iii) *Audio* features are complementary to images, and are being used successfully for unsupervised audio-visual learning [2, 3]. We extract audio features for 1s raw audio samples using the SoundNet8 model [3] from the `pool5` layer, $\mathbf{a}_t^i = \phi_a(\mathbf{u}_t), \mathbf{a}_t^i \in \mathbb{R}^{256}$. In conjunction with the image, we expect audio to help detect effects such as swooshing *winds*, *splashing* water, or a mechanical *shake*.

(iv) *Object detections* can play a complementary role to image features. For example, presence of people may indicate *physical interaction*, while a car suggests *shake*. Based on predictions from the YOLO9000 object detector [28], we choose 550 most significant classes and form a sparse feature vector corresponding to the object detection probabilities $\mathbf{o}_t^i = \phi_o(\mathbf{x}_t), \mathbf{o}_t^i \in \mathbb{R}^{550}$ for each frame.

Finally, we obtain a sub-shot representation by concatenating embedded features through linear layers:

$$\phi(\mathbf{x}_t^i, \mathbf{u}_t^i) = [W_v\mathbf{v}_t^i, W_f\mathbf{f}_t^i, W_a\mathbf{a}_t^i, W_o\mathbf{o}_t^i]\,, \qquad (1)$$

Each linear layer embeds features into a $D = 512$ dim space. Using pre-trained models as feature extractors was a crucial requirement to train good models on our dataset.

Classifiers. We build several two-layer MLP classifiers on the sub-shot representations to predict: (i) *count* (C^n): number of effects present in the current sub-shot (0, 1, 2, 3); (ii) *effect* (C^e): the effect labels (9 classes); (iii) effect *intensities* (C^i): three classes; and (iv) effect-specific *details* ($C^{d1}, \ldots, C^{dG}$) such as type of physical interaction or direction of wind (each with its own independent MLP).

The count, effect, and intensity classifiers have hidden layers with size corresponding to the input, *i.e.* $h_n, h_e, h_i \in \mathbb{R}^{2048}$. As detail classifiers have much less training data, we set the hidden layer $h_d \in \mathbb{R}^{512}$. The output layer computes predictions in one-of-K classes:

$$\hat{\mathbf{y}}_t^i = W_2 \cdot \mathrm{ReLU}(W_1 \cdot \phi(\mathbf{x}_t^i, \mathbf{u}_t^i)) = C(\phi(\mathbf{x}_t^i, \mathbf{u}_t^i))\,. \qquad (2)$$

Biases are ignored for notational brevity. We leverage the architecture (summarized in Fig. 7) to train task-specific models in the following.

4.3. Trimmed Video: Effect recognition

Our first task is to predict the effect type and the corresponding details when provided with a trimmed clip (known to contain an effect). We treat this prediction as tagging, *i.e.* we do not take into account POV information.

We first compute the set of sub-shots that overlap with the trimmed clip and use them to obtain predictions $\hat{\mathbf{y}}_t^i$. For a given effect with sub-shots $\mathbf{x}_1, \ldots, \mathbf{x}_T$, we compute the final prediction $\hat{\mathbf{y}}$ by

$$\hat{\mathbf{y}} = \max_t \frac{1}{|\mathbf{x}_t|} \sum_i \hat{\mathbf{y}}_t^i\,. \qquad (3)$$

Averaging results within frames of a sub-shot ($\sum_i$) improves robustness, and selecting the highest scoring sub-shot ($\max_t$) reduces noise.

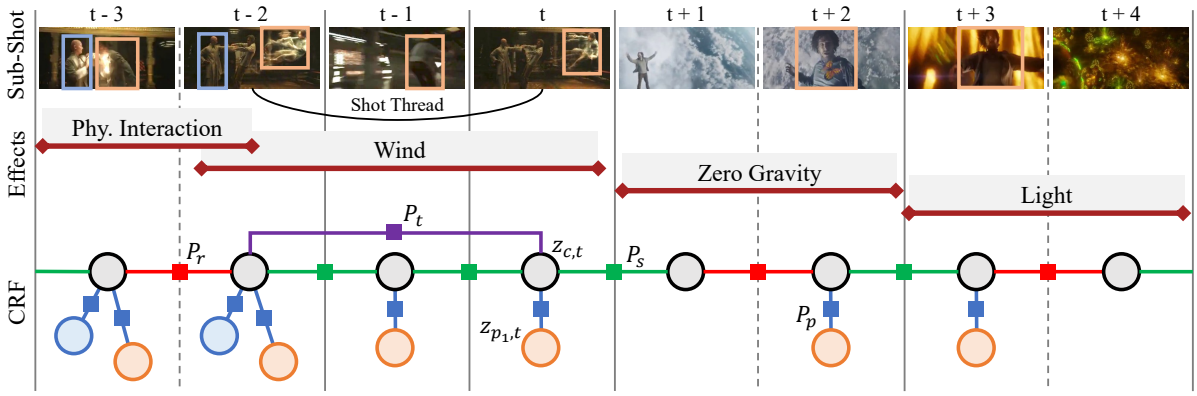

Figure 8: Depiction of our CRF model on a movie clip from *Doctor Strange*. We display the sub-shot (dashed) and shot boundaries (solid) as vertical lines, and show a sample image frame along with person tracks. Ground-truth effect annotations are also displayed. Finally, we present a glimpse of the CRF model that corresponds to a random variable for each sub-shot and person track, and the four types of pairwise potentials connecting them.

Learning and Inference details. While the pre-trained feature extractors are fixed, the feature embedding layers (Eq. 1) and effect C_e, intensity C_i, and detail $C_{d.}$ classifiers are trained. We adopt the cross-entropy loss for each classifier and optimize our model with Adam [20] using a constant learning rate $1e-6$. We found that training all classifiers jointly by accumulating losses worked well. Due to the large class imbalance at both levels: effects and details, we use an inverse frequency weight capped at 50 to train our models. For long clips, we select a maximum of $T = 4$ consecutive sub-shots ($\sim$10s), automatically providing some data augmentation by random selection. Additionally, we use a dropout rate of 0.3 for MLP classifiers and set weight decay to 0.1. We choose a model checkpoint that has the highest intensity-weighted effect-classification accuracy weighted on the validation set.

At test time, we predict the highest scoring effect and obtain detail predictions corresponding to it.

4.4. Untrimmed video: Effect detection

While trimmed video effect recognition focuses on classifying effects globally over a trimmed clip, we now aim to predict the effects experienced by both the characters and camera, and localize them in time. In this scenario, we are given an entire video along with sub-shot boundaries, threading information, and person tracks within each sub-shot. While tracking is performed within a shot, we divide the track across sub-shot boundaries if required. Our goal is to determine the effect type, start and end-time and POV.

We compute and average frame-level predictions within the sub-shot. Predictions are denoted by subscripts: camera as $\hat{\mathbf{y}}_{c,t}$, and for person track p_l (l indexes tracks within a sub-shot) as $\hat{\mathbf{y}}_{p_l,t}$ within each sub-shot t. Two important predictions are considered: (i) number of effects C^n, that predicts $\hat{\mathbf{y}}^n$ in 4 classes (0 – no effect, and 1, 2, or 3 effects), and (ii) C^e that predicts the effect labels $\hat{\mathbf{y}}^e$. Prediction scores are converted to probabilities via softmax.

Conditional Random Field. Effects display a strong correlation across sub-shots, shots, and threads. For example, within a shot (across sub-shots), the atmospheric conditions *weather* and *wind* are unlikely to change.

We construct a CRF that incorporates these correlations, with the aim to obtain joint predictions for the entire video. Fig. 8 illustrates our model. For each sub-shot at time t, we assign a random variable $z_{c,t}$ to denote the effect experienced by POV camera, and $z_{p_l,t}$ for effects of person tracks within that sub-shot. All variables are $z \in \mathbb{R}^{10}$, corresponding to scores for the "no-effect" class and 9 effect labels.

In our CRF we consider unary $U(\cdot)$ and pairwise potentials $P_q(\cdot,\cdot)$ and $P_p(\cdot,\cdot)$:

$$E(z,\mathbf{w}) = \sum_t \Big(w_{u_c} U(z_{c,t}) + \sum_l w_{u_p} U(z_{p_l,t}) \Big) + w_q \sum_{(t,t')\in\mathcal{Q}} P_q(z_{c,t}, z_{c,t'}) + w_p \sum_{(t,l)} P_p(z_{c,t}, z_{p_l,t}) \,. \quad (4)$$

For brevity, we denote three camera-related edge types as $\mathcal{Q}$. These edges link (i) $\mathcal{R}$: neighboring sub-shots; (ii) $\mathcal{S}$: neighboring shots; and (iii) $\mathcal{T}$: the last and first sub-shot of a threaded set of shots.

Our unary potential takes the form

$$U(\cdot) = [p^n, (1-p^n)\cdot \hat{\mathbf{y}}^e] \,, \quad (5)$$

where, p^n refers to the probability of the current sub-shot not seeing an effect, and is the first element of the number of effects $p^n = \hat{\mathbf{y}}^n[0]$.

The pair-wise potentials encode co-occurrence between the 9 effects and the no-effect class, and are denoted by P_r between sub-shots, P_s between shots, and P_t between threads. The relationship between camera and person-tracks P_p is within the same sub-shot. We learn a different weight with each edge type.

CRF learning and inference. Our CRF may contain cycles due to thread edges (see Fig. 8), and thus inference is NP-hard. To perform inference we use distributed convex belief propagation [32], which has convergence guarantees. To learn the weights, we use the primal-dual method [15, 33], and use the typical 0-1 loss.

Sub-shot predictions to time intervals. Inspired by its success in untrimmed action recognition, we group sub-shot level predictions using the watershed transform [46] to obtain contiguous time intervals with effect predictions. Each of the 9 classes are processed separately to obtain effect detections of the form: $(t_{\text{start}}, t_{\text{end}}, e)$.

Details. The neural network is separately trained to predict unaries using cross-entropy loss. Similar to the trimmed model, we train our unaries prediction model with Adam [20]. Recall that we extract image features from the `pool5` layer that provides 512-d vectors in a 7×7 spatial grid. In contrast to the camera unaries, person track unaries average features within the region of interest based on the detection bounding box. We pick a model checkpoint that performs well at predicting effect existence (based on number of effects) and effect recognition.

5. Experiments

We first discuss the metrics for our new dataset and tasks.

Trimmed video: Effect recognition metrics. We propose several metrics to evaluate effect and details prediction when given a trimmed video. Our first metric *E* is effect classification accuracy. We propose intensity-weighted accuracy *IE*, as a viewer experience metric that incorporates user annoyance when mild/strong effects are misclassified: $1\times$ (mild), $2\times$ (medium), and $3\times$ (strong).

To evaluate detail prediction, we use a metric similar to [44]. We introduce *D-GT* that measures the fraction of details that are correct for each sample given GT effect label. Additionally, *DA-GT* measures the fraction of samples that have *all* details correct. Similarly, detail prediction using predicted effect is evaluated by *D-PR* and *DA-PR*.

Finally, we present a slightly modified form of a confusion matrix. As multiple effects can co-occur, one trimmed clip could correspond to more than one effect label. In such a case, if we are able to correctly classify one of the effects, we say that the other effects are "missed", but not "misclassified". For example, during an explosion clip with *light* and *shake*, if our model only predicts *light*, we count light as correct, and shake as missed.

Untrimmed video: Effect detection metrics. We consider two POV paradigms: (i) all effects are mapped to the camera (similar to trimmed); or (ii) both camera and characters experience effects. Nevertheless, given the entire video we are required to predict effect start- and end-times along with effect labels and POV.

Model	E	IE	D-GT	DA-GT	D-PR	DA-PR
Random	11.1	11.1	23.6	3.8	2.6	0.4
Ours	41.5	42.0	35.5	9.9	**15.7**	**5.3**
Ours + Attn.	**43.7**	**45.9**	35.8	9.9	**15.8**	4.6
All - visual	26.9	28.5	36.2	10.4	10.2	3.0
All - optflow	32.1	32.4	33.0	8.4	10.9	3.0
All - audio	40.7	41.9	35.0	9.4	15.2	4.5
All - objdet	36.7	37.4	**37.8**	**11.3**	15.5	4.5

Table 2: Results on the test set for trimmed effect recognition task with ablation study for different features (no attn.). Random indicates the performance when treating all classes as equally likely. E, D-PR and DA-PR are important metrics for future comparison.

Similar to a standard detection setting, our first metric *Exist Average Precision (AP)* compares models on their ability to predict whether a sub-shot (or person track) contains an effect or not. We also include the effect prediction accuracy (*Effect ACC*) at sub-shot (or track) level.

After merging sub-shot predictions, we evaluate time interval predictions based on precision, recall and F1-measure. A prediction is said to be a true positive when it has a temporal IoU >10% (similar to ActivityNet [17]). All other predictions are "false positives", and ground-truth effects that do not see any prediction are "misses".

5.1. Dataset Quality

For 15 clips (5 min each) from 5 movies, we gathered 3 sets of labels from different annotators. We evaluate human effect detection performance by comparing all pairs of annotations (one as GT other as pred.), using the F1 metric. Temporal detection and localization is a hard task even for humans (also seen in action localization), and we obtain an average F1 score of 62.6% (54% - 67% for each movie).

We also analyze effect classification agreement in trimmed clips among humans using Amazon Mechanical Turk (AMT). Each clip was shown to 5 workers. At least one worker agreed with our label for 90% of the clips. When 4 of 5 workers provide the same label, this corresponds to an accuracy of 88.4%. As each clip can exhibit multiple effects (*e.g. shake* and *wind*), it is not necessary to obtain a clear majority.

5.2. Trimmed video: Effect recognition

We present the effect recognition results on the test set in Table 2. A baseline (row 1) that chooses 1 of K classes with equal chance has 11.1% accuracy, however, rarely gets the effect and details all correct (0.4%). Selecting the most likely label (*light*) can achieve 25.8% effect accuracy.

In comparison, our neural model performs much better with an accuracy of 41.5%. With attention, we obtain 43.7%, and intensity-weighted accuracy of 45.9%. We believe that IE does not differ much from accuracy E partially due to inconsistencies in intensity annotations. Finally, our

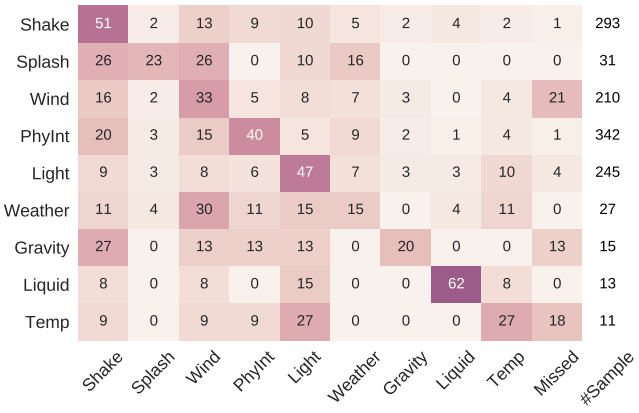

Figure 9: Our modified form of confusion matrix on trimmed effect classification. The second-last column, "Missed" corresponds to effects whose samples have more than one annotation. The last column shows the number of samples in the test set for each class.

Figure 10: Qualitative result of trimmed prediction for a clip in Pirates of the Caribbean. Each frame corresponds to a sub-shot, and we show the probability of predicting *physical interaction* – related to the fight scene between the two characters. Averaging across sub-shots leads to predictions such as *light* and *wind* (possibly because of motion), however, max is able to predict correctly.

model is able to predict the effect label and all details correct for 5.3% of all annotations, hinting towards the difficulty of the task.

We also present an ablation study evaluating the importance of each feature stream by leaving one out at a time. The visual features play an important role, followed by motion (optical flow) features. This makes sense as *light*, *physical interaction* and *shake* are among the most dominant classes. Objects are also quite important (*e.g.* cars *shake*) validated by the approximate 5% drop in effect prediction accuracy when ignoring them. Finally, audio features seem to have smallest contribution, however, do affect DA-PR.

Fig. 9 presents the modified confusion matrix. The confusion between *light* - *temperature*, or *splash* - *wind* - *weather* seem genuine as these are difficult effects to discriminate. Finally, Fig. 10 is an example illustrating max across sub-shots works better than mean.

5.3. Untrimmed video: Effect detection

We present effect detection results in two parts. First, in Table 3 we show the performance of detecting the presence of and classifying effects for each processing unit (sub-shot camera and/or track). The top part (rows 1-5) display results when all effects are mapped to the camera POV. We see the impact of different pairwise potentials: sub-shot, shot, and thread (rows 2-4), while, row 5 corresponds to the best result when using all pairwise terms. A large 8.8% boost in effect accuracy and a substantial 1.9% increase on Exist AP is obtained over the unary outputs from the neural model.

#	POV	Model	Exist AP	Effect ACC
1	Camera	Unaries	55.4	43.6
2		CRF: U + sub-shot	56.3	44.4
3		CRF: U + shot	51.4	50.4
4		CRF: U + thread	53.8	44.6
5		CRF: U + all pairwise	**57.3**	**52.4**
6	Cam + Tracks	Unaries	**29.4**	45.3
7		CRF: U + video pairwise	27.9	48.0
8		CRF: U + all pairwise	28.8	**48.8**

Table 3: Effect detection and classification performance on the test set. Results are evaluated at sub-unit level (before combining into time intervals) and measured for each sub-shot (top) and sub-shot and person track (bottom).

POV	Model	Prec	Recl	F1
Camera	Unaries	15.0	35.1	21.1
	CRF	**25.2**	**35.6**	**29.5**
Cam + Tracks	Unaries	14.2	**37.2**	20.5
	CRF	**16.7**	28.7	**21.1**

Table 4: Effect detection results on the test set, evaluating time interval predictions with IoU >10%. Top: effects are mapped to the camera POV. Bottom: effects for camera and tracks are separate.

The bottom part presents results when considering effects separately for camera and tracks POVs. Row 7 and 8 show the impact of the CRF, and the final pairwise potential connecting camera nodes with tracks P_p (*c.f.* Fig. 8). We observe a small 3.5% improvement in effect accuracy, however detection AP reduces a little.

We combine the unit predictions into time intervals, and display results for comparing ground-truth and predicted time-intervals in Table 4. Note that this task is considerably harder as we need to predict a contiguous set of sub-shots correctly in order to obtain good time boundaries. When assuming all effects apply to camera POV, the CRF provides a 8.4% boost in F1 measure. However, when analyzing camera and person tracks, the improvement is small at 0.6%.

6. Conclusion

We introduced the Movie4D dataset consisting of 63 movies and 9286 effect annotations that enlist physical and special effects in movies along with details such as duration, intensity, effect sub-types and direction. We presented a thorough exploration of the dataset showcasing its features. We proposed a CRF model that combines cues from a multimodal neural network while respecting shot boundaries and threading information in a video. We evaluated our approach through various ablation studies, pointing to exciting avenues going forward.

Acknowledgments. Supported by NSERC. We thank NVIDIA for GPU donation. We thank Relu Patrascu for infrastructure support, and UpWork annotators for data annotation.

References

[1] ALL effects. http://www.alleffects.com/. Retrieved 2017-11-15. 2

[2] R. Arandjelović and A. Zisserman. Look, Listen and Learn. In *ICCV*, 2017. 2, 5

[3] Y. Aytar, C. Vondrick, and A. Torralba. SoundNet: Learning Sound Representations from Unlabeled Video. In *NIPS*, 2016. 2, 5

[4] P. Bojanowski, F. Bach, I. Laptev, J. Ponce, C. Schmid, and J. Sivic. Finding Actors and Actions in Movies. In *ICCV*, 2013. 2

[5] H. Chu, R. Urtasun, and S. Fidler. Song from pi: A musically plausible network for pop music generation. In *arXiv:1611.03477*, 2016. 1

[6] Y. Chunyu, F. Jun, W. Jinjun, and Z. Yongming. Video Fire Smoke Detection Using Motion and Color Features. *Fire Technology*, 46:651–663, 2010. 2

[7] T. Cour, B. Sapp, C. Jordan, and B. Taskar. Learning from ambiguously labeled images. In *CVPR*, 2009. 2

[8] M. Eichner, M. Marin-Jimenez, A. Zisserman, and V. Ferrari. 2D Articulated Human Pose Estimation and Retrieval in (Almost) Unconstrained Still Images. *IJCV*, 99(2):190–214, 2012. 2

[9] M. Everingham, J. Sivic, and A. Zisserman. "Hello! My name is ... Buffy" – Automatic Naming of Characters in TV Video. In *BMVC*, 2006. 2

[10] C. Feichtenhofer, A. Pinz, and A. Zisserman. Convolutional Two-Stream Network Fusion for Video Action Recognition. In *CVPR*, 2016. 1, 5

[11] K. Garg and S. K. Nayar. When does a camera see rain? In *ICCV*, 2005. 2

[12] L. A. Gatys, A. S. Ecker, and M. Bethge. A neural algorithm of artistic style. In *arXiv:1508.06576*, 2015. 1

[13] J. F. Gemmeke, D. P. W. Ellis, D. Freedman, A. Jansen, W. Lawrence, R. C. Moore, M. Plakal, and M. Ritter. Audio set: An ontology and human-labeled dataset for audio events. In *ICASSP*, 2017. 2

[14] R. Girshick. Fast R-CNN. In *ICCV*, 2015. 1

[15] T. Hazan and R. Urtasun. A Primal-Dual Message-Passing Algorithm for Approximated Large Scale Structured Prediction. In *NIPS*, 2010. 7

[16] K. He, X. Zhang, S. Ren, and J. Sun. Deep Residual Learning for Image Recognition. In *CVPR*, 2016. 1

[17] F. C. Heilbron, V. Escorcia, B. Ghanem, and J. C. Niebles. ActivityNet: A Large-Scale Video Benchmark for Human Activity Understanding. In *CVPR*, 2015. 2, 7

[18] A. Karpathy, J. Johnson, and L. Fei-Fei. Visualizing and Understanding Recurrent Networks. In *ICLR 2016 Workshop*, 2016. 1

[19] K.-M. Kim, C.-J. Nan, M.-O. Heo, S.-H. Choi, and B.-T. Zhang. PororoQA: A Cartoon Video Series Dataset for Story Question Understanding. In *NIPS 2016 Workshop on Large Scale Computer Vision Systems*, 2016. 2

[20] D. Kingma and J. Ba. Adam: A Method for Stochastic Optimization. 2015. 6, 7

[21] I. Laptev, M. Marszalek, C. Schmid, and B. Rozenfeld. Learning Realistic Human Actions from Movies. In *CVPR*, 2008. 2

[22] J. Lee and S. Choi. Real-time perception-level translation from audio signals to vibrotactile effects. In *SIGCHI Conference on Human Factors in Computing Systems*, 2013. 2

[23] J. Lee, B. Han, and S. Choi. Motion Effects Synthesis for 4D Films. *IEEE Transactions on Visualization and Computer Graphics*, 22(10):2300–2314, 2016. 1, 2

[24] M. J. Marin-Jimenez, A. Zisserman, M. Eichner, and V. Ferrari. Detecting People Looking at Each Other in Videos. *IJCV*, 106(3):282–296, 2014. 2

[25] P. Mettes, R. T. Tan, and R. C. Veltkamp. Water Detection through Spatio-Temporal Invariant Descriptors. *CVIU*, 154:182–191, 2017. 2

[26] A. Patron-Perez, M. Marszalek, I. D. Reid, and A. Zisserman. Structured Learning of Human Interactions in TV Shows. *PAMI*, 34(12):2441–2453, 2012. 2

[27] V. Ramanathan, A. Joulin, P. Liang, and L. Fei-Fei. Linking people in videos with "their" names using coreference resolution. In *ECCV*, 2014. 2

[28] J. Redmon and A. Farhadi. YOLO9000: Better, Faster, Stronger. In *CVPR*, 2016. 4, 5

[29] A. Rohrbach, A. Torabi, M. Rohrbach, N. Tandon, C. Pal, H. Larochelle, A. Courville, and B. Schiele. Movie Description. *IJCV*, 123(1):94–120, 2017. 2

[30] M. Roser and F. Moosmann. Classification of Weather Situations on Single Color Images. In *IEEE Intelligent Vehicles Symposium*, 2008. 2

[31] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *IJCV*, 115(3):211–252, 2015. 5

[32] A. Schwing, T. Hazan, M. Pollefeys, and R. Urtasun. Distributed Message Passing for Large Scale Graphical Models. In *CVPR*, 2011. 7

[33] A. Schwing, T. Hazan, M. Pollefeys, and R. Urtasun. Efficient Structured Prediction with Latent Variables for General Graphical Models. In *ICML*, 2012. 7

[34] J. Seo, R. H. Osgouei, S.-C. Chung, and S. Choi. Vibrotactile Rendering of Gunshot Events for 4D Films. In *IEEE Haptics Symposium*, 2016. 2

[35] K. Simonyan and A. Zisserman. Very Deep Convolutional Networks for Large-Scale Image Recognition. In *ICLR*, 2015. 5

[36] K. Soomro, A. Roshan Zamir, and M. Shah. UCF101: A Dataset of 101 Human Actions Classes From Videos in The Wild. In *CRCV-TR-12-01*, 2012. 2

[37] M. Tapaswi, M. Bäuml, and R. Stiefelhagen. StoryGraphs: Visualizing Character Interactions as a Timeline. In *CVPR*, 2014. 5

[38] M. Tapaswi, M. Bäuml, and R. Stiefelhagen. Aligning Plot Synopses to Videos for Story-based Retrieval. *International Journal of Multimedia Information Retrieval (IJMIR)*, 4(1):3–16, 2015. 2

[39] M. Tapaswi, M. Bäuml, and R. Stiefelhagen. Book2Movie: Aligning Video scenes with Book chapters. In *CVPR*, 2015. 2

[40] M. Tapaswi, Y. Zhu, R. Stiefelhagen, A. Torralba, R. Urtasun, and S. Fidler. MovieQA: Understanding Stories in Movies through Question-Answering. In *CVPR*, 2016. 2

[41] L. J. P. van der Maaten and G. E. Hinton. Visualizing High-Dimensional Data using t-SNE. *JMLR*, 9(11):2579–2605, 2008. 4

[42] E. Waite, D. Eck, A. Roberts, and D. Abolafia. Project magenta. https://magenta.tensorflow.org/. 1

[43] K. Xu, J. Ba, R. Kiros, K. Cho, A. Courville, R. Salakhutdinov, R. Zemel, and Y. Bengio. Show, Attend and Tell: Neural Image Caption Generation with Visual Attention. In *ICML*, 2015. 1

[44] M. Yatskar, L. Zettlemoyer, and A. Farhadi. Situation Recognition: Visual Semantic Role Labeling for Image Understanding. In *CVPR*, 2016. 2, 7

[45] Y. Yusoff, W. Christmas, and J. Kittler. A Study on Automatic Shot Change Detection. *Multimedia Applications and Services*, 1998. 4

[46] Y. Zhao, Y. Xiong, L. Wang, Z. Wu, D. Lin, and X. Tang. Temporal Action Detection with Structured Segment Networks. In *ICCV*, 2017. 7

[47] Y. Zhu, R. Kiros, R. Zemel, R. Salakhutdinov, R. Urtasun, A. Torralba, and S. Fidler. Aligning Books and Movies: Towards Story-like Visual Explanations by Watching Movies and Reading Books. In *ICCV*, 2015. 2