VIDEO UNDERSTANDING: DATA PRIVACY, PIPELINE SIMPLICITY, AND IMPLEMENTATION EFFICIENCY

A Dissertation

by

ZHENYU WU

Submitted to the Office of Graduate and Professional Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

Chair of Committee,	Zhangyang Wang
Committee Members,	Dezhen Song
	Anxiao Jiang
	Yang Shen
Head of Department,	Scott Schaefer

August 2021

Major Subject: Computer Science

ABSTRACT

Video understanding aims to automatically detect objects of interest from videos and recognize the scene, actions, content, or attributes. Some well-known tasks defined on videos include human action recognition, human action spatio-temporal localization (a.k.a, action detection), and text spotting. We improve the existing performance-driven approaches for video understanding tasks in three aspects: privacy in data sharing, simplicity in model design, and efficiency in hardware deployment. For privacy-aware data sharing, we formulate a novel adversarial training framework to learn an anonymization transform for input videos such that the trade-off between target utility task performance and the associated privacy budgets is explicitly optimized on the anonymized videos. Given few public datasets available with both utility and privacy labels, we constructed a new dataset, termed PA-HMDB51, with both target task labels (action) and selected privacy attributes (skin color, face, gender, nudity, and relationship) annotated on a per-frame basis. For efficiency-driven hardware deployment, we propose a multi-stage image processor that takes videos' redundancy, continuity, and mixed degradation into account. The model is pruned and quantized in a hardware-friendly way to save energy consumption further. Our proposed energy-efficient video text spotting solution outperforms all previous methods by achieving a competitive trade-off between energy efficiency and performance. For simplicity-guided model design, we develop a new paradigm for video action detection using Transformers, which effectively removes the need for such specialized components and achieves superior performance. Without using pre-trained person/object detectors, RPN, or memory bank, our Transformer-based Video Action Detector (TxVAD) utilizes two types of Transformers to capture scene context information and long-range spatio-temporal context information, for person localization and action classification, respectively.

ACKNOWLEDGMENTS

Throughout the writing of this thesis, I have received a great deal of support and assistance. Here I would like to express my sincere thanks and appreciation to all the people who have helped me in my research and the completion of the thesis.

First, I would like to thank my advisor, Professor Zhangyang (Atlas) Wang, for his supervision, inspiration, and advice that made this dissertation possible. His expertise and research philosophy were invaluable in formulating the research problems and solutions.

Second, I would like to thank my dissertation committee members, Professor Anxiao Jiang, Professor Yang Shen, and Professor Dezhen Song, for their time, advice, and firm support. Also, I would like to acknowledge my lab mates and collaborators at the VITA (Visual Informatics Group @ Texas A&M University) Lab in the Department of Computer Science and Engineering, for offering valuable advice on my work.

Most importantly, I would like to express the sincerest gratitude to my parents and my girlfriend. Their love supports me mentally during my Ph.D. program.

Finally, I would like to thank Texas A&M University and all the funding agencies, including Adobe Research and U.S. Army Research Lab.

CONTRIBUTORS AND FUNDING SOURCES

Contributors

This work was supported by a thesis (or) dissertation committee consisting of Professor Zhangyang Wang [advisor], Professor Dezhen Song and Professor Anxiao Jiang of the Department of Computer Science and Engineering, and Professor Yang Shen of the Department of Electrical and Computer Engineering.

All other work conducted for the dissertation was completed by the student independently.

Funding Sources

Graduate study was supported by Adobe Research and U.S. Army Research Lab.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

TABLE Page

1. INTRODUCTION*

1.1 Background and Motivation

Video understanding, a key functionality of intelligent video systems, automatically detects objects of interest from videos and recognizes the scene, actions, content, or attributes. For example, in video action detection, where the objects of interest are persons, the goal is to localize persons and recognize their actions in space and time. Likewise, in video text spotting, where the objects of interest are texts, the goal is to localize texts and recognize their content in space and time.

We improve the existing performance-driven approaches for video action detection or video text spotting in three aspects: data sharing privacy, model design simplicity, and hardware deployment efficiency.

1.1.1 Privacy

Due to the computationally demanding nature of video-based recognition tasks, only some of the tasks can run on the resource-limited local devices, which makes transmitting (part of) data to the cloud necessary. Such a data sharing scenario has raised growing concern about the potential privacy risk caused by malicious third parties' misusing the data. Therefore, the objective of privacy-aware data sharing is to alleviate privacy concerns without compromising data utility.

1.1.2 Simplicity

Video action detection solutions are growing to be more and more sophisticated by adopting many specialized components, *e.g.*, pretrained person/object detectors, region proposal network, memory bank, and so on. The adoption of many specialized components has made video ac-

*Part of this chapter is reprinted with permission from "Privacy-Preserving Deep Action Recognition: An Adversarial Learning Framework and A New Dataset" [1] by Z. Wu, H. Wang, Z. Wang, H. Jin, and Z. Wang, IEEE Transactions on PatternAnalysis and Machine Intelligence, Copyright 2020 by IEEE. DOI: https://doi.org/10.1109/TPAMI.2020.3026709. And part of this chapter is reprinted with permission from "Towards Privacy-Preserving Visual Recognition via Adversarial Training: A Pilot Study" [3] by Z. Wu, Z. Wang, Z. Wang, and H. Jin, Proceedings of the European Conference on Computer Vision, Pages 605-624, Copyright 2018 by Springer. And part of this chapter is reprinted with permission from "E^{2}VTS: Energy-Efficient Video Text Spotting From Unmanned Aerial Vehicles" [2] by Z. Hu, P. Pi, Z. Wu, Y. Xue, J. Shen, J. Tan, X. Lian, Z. Wang, and J. Liu, Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops, Pages 905-913, 2021.

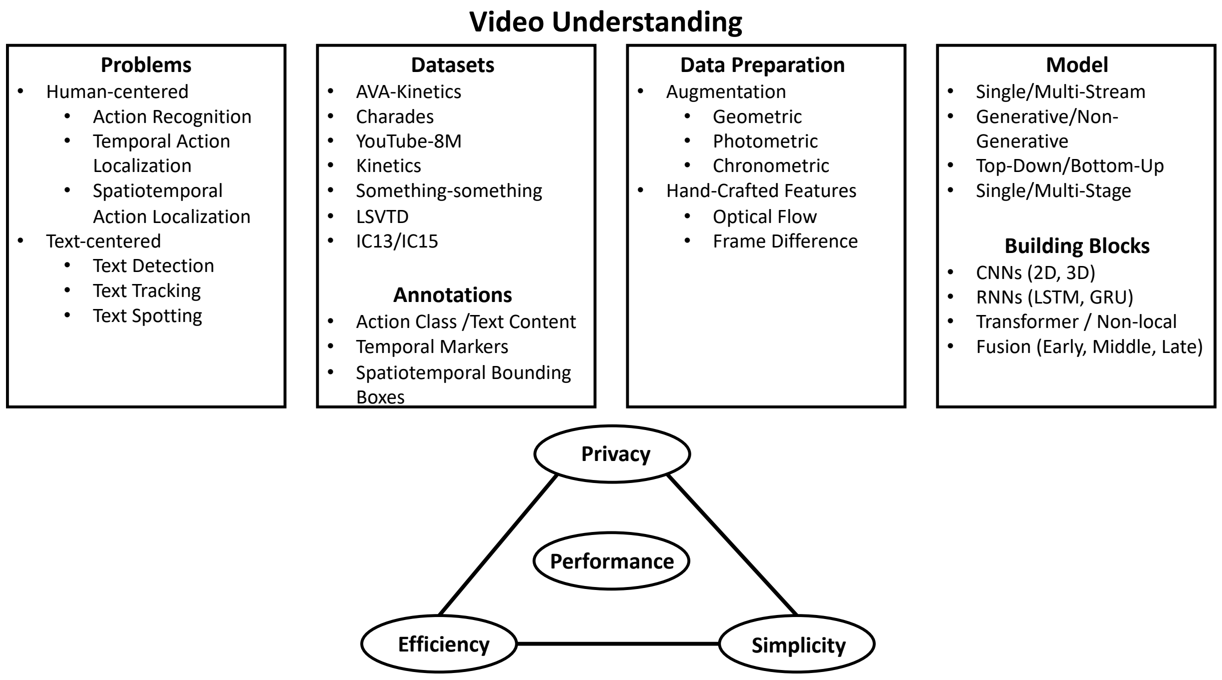


Figure 1.1: Overview of video understanding steps (problem definition, dataset selection, data preparation, and model design) and underlying principles (computational performance, data privacy, model simplicity, and energy efficiency.

tion detection model gigantic and conceptually highly sophisticated. The objective of simplified model design is to pursue a general, conceptually simple, and sufficiently versatile architecture and achieve even better performance.

1.1.3 Efficiency

Video text spotting system is usually deployed on edge devices, *e.g.*, mobile phones or unmanned aerial vehicles. Existing solutions are solely performance-driven and fail to take energy consumption into account. A standard pipeline for video text spotting has four stages: text detector, text recognizer, text tracker, and post-processing. The energy-efficient model design aims to design a data processor that utilizes video redundancy, continuity, and mixed degradation for energy-saving purposes.

1.2 Thesis Contributions

1.2.1 Privacy

We take one of the first steps towards addressing this challenge of privacy-preserving, video-based action recognition, via the following contributions:

- **A General Adversarial Training and Evaluation Framework.** We address the privacy-preserving action recognition problem with a novel adversarial training framework. The framework explicitly optimizes the trade-off between target utility task performance and the associated privacy budgets by learning to anonymize the original videos. To reduce the training instability, we design and compare three different optimization strategies. We empirically find that one strategy generally outperforms the others under our framework and provide intuitive explanations.

- **Practical Approximations of "Universal" Privacy Protection.** The privacy budget in our framework cannot be defined w.r.t. one model that predicts privacy attributes. Instead, the ideal privacy protection must be universal and model-agnostic, *i.e.*, preventing every possible attacker model from predicting private information. To resolve this so-called **"$\forall$ challenge"**, we propose two effective strategies, *i.e.*, *restarting* and *ensembling*, to enhance the generalization capability of the learned anonymization to defend against unseen models. We leave it as our future work to find better methods for this challenge.

- **A New Dataset with Action and Privacy Annotations.** When evaluating privacy protection on complicated privacy attributes, there is no off-the-shelf video dataset with both action (utility) and privacy attributes annotated, either for training or testing. Such a dataset challenge is circumvented in our previous work [3] by using the VISPR [4] dataset as an auxiliary dataset to provide privacy annotations for cross-dataset evaluation (details in Section 3.6). However, this protocol inevitably suffers from the domain gap between the two datasets: while the utility was evaluated on one dataset, the privacy was measured on a different dataset. The incoherence in utility and privacy evaluation datasets makes the obtained utility-privacy trade-off less convincing. To reduce this gap, in this paper, we construct the very first testing benchmark dataset, dubbed **P**rivacy-**A**nnotated **HMDB51** (PA-HMDB51), to evaluate privacy protection and action recognition on the same videos simultaneously. The new dataset consists of 515 videos originally from HMDB51. For each video, privacy labels (five attributes: skin color, face, gender, nudity, and relationship) are annotated on a per-frame basis. We benchmark our proposed framework on the new dataset and justify its effectiveness.

1.2.2 Efficiency

We propose an Energy-Efficient Video Text Spotting solution, dubbed as E^{2}VTS. The contributions of E^{2}VTS are summarized as follows:

- **Novel Training & Inference Strategies.** To obtain better text spotting performance, we revisit RCNN and empirically find that crop and resize outperforms aligned RoI Pooling when connecting the text recognizer with the text detector. To further save energy consumption, we propose a multi-stage image processor to select the highest-quality frame in a sliding window, reject text-free frames as well as crop non-text regions, and reject out-of-distribution frames.

- **Practical Evaluation.** Given a real-world video text spotting dataset for evaluation and a Raspberry Pi for model deployment, we conducted thorough ablation studies on the proposed training and inference strategies. The evaluation metric takes both energy consumption and text spotting performance into consideration. Models are pruned and quantized before being deployed on Pi.

1.2.3 Simplicity

We present a new paradigm for video action detection using a Transformer based architecture. Our proposed Transformer-based Video Action Detector (**TxVAD**) simplifies the video action detection pipeline, which drops multiple specialized components, including pretrained person/object detectors, RPN, and memory bank. It requires no customized layer beyond existing standard libraries. The contributions of TxVAD are outlined below:

- **Framework.** TxVAD carries out an ambitious attempt to solve video action detection using a Transformer based paradigm. Notably, it relies purely on two spatio-temporal Transformers of encoder-decoder architecture for person localization and action classification, respectively. TxVAD fully relies on the self-attentions and cross-attentions to explicitly model all pairwise interactions between spatio-temporal elements in videos.

- **Training.** We find training such an end-to-end Transformer architecture on the giant video datasets highly challenging, similarly as reported. To remove that hurdle, we first design a hardness-aware curriculum training strategy leveraging the naturally different "semantic difficulty levels" of video action categories (person pose, person-person interactions, and person-object interactions). That strategy substantially stabilizes TxVAD's training.

1.3 Related Work

1.3.1 Videos Understanding Related Tasks

1.3.1.1 Action Recognition

Action recognition is a fundamental task in video understanding that recognizes human actions based on the complete action execution in trimmed videos. Many datasets [5–12] have been released to promote research in video action recognition. *Temporal Modeling* is the key factor in designing deep networks. There are roughly three type of architectures in temporal modeling: ConvNet+LSTM [13–15], 3D ConvNets [16–19], and Two-Stream Networks [20, 21]. Action recognition has focused on trimmed video clips, *i.e.*, classifying a short clip into action classes. However, most videos involve multiple people performing multiple actions across different space and time.

1.3.1.2 Action Detection

Spatio-temporal video action detection requires localizing persons and recognizing their actions in space and time from video sequences, including person pose actions, person-object interaction actions, and person-person interaction actions. Below we briefly review a few representative approaches.

STEP [22] is a progressive learning framework that consists of spatial refinement and temporal extension, which iteratively refines the coarse cuboid proposals towards action locations and incorporates longer-range temporal information to improve action classification. STAGE [23] learns relations between actors and objects by self-attention operations over a spatiotemporal graph representation of the video. VideoCapsuleNet is a simple end-to-end 3D capsule network that jointly performs pixel-wise segmentations of actions and action classification. HISAN [24] combines the two-stream CNN with a hierarchical bidirectional self-attention mechanism that captures long-term temporal dependency and spatial context information to improve action localization. In addition, motion saliency captured by optical flow is fused to enhance the motion information. LFB [25] introduces a long-term feature bank that stores long-term supportive memory extracted over the entire video. SlowFast Networks [18] involve a slow pathway operating at low frame rates to capture spatial semantic information, and a lightweight fast pathway operating at high frame rates to capture rapid motion better. Recently, Context-Aware RCNN [26] enlarges the resolution of small actor boxes by cropping and resizing instead of RoI-Pooling, also extracting scene context information aided by LFB. AIA [27] leverages an interaction aggregation structure and an asynchronous memory update algorithm to efficiently model very long-term interaction dynamics. ACAR-Net [28] learns to reason high-order relations actor-context-actor relations using relational feature bank to preserve spatial contexts. All above methods, albeit effective, heavily hinge on various specially crafted components such as region proposals, memory banks, or relation reasoning.

1.3.1.3 Text Reading in Images

Scene Text Detection. *1. Object Detection-based Methods:* Inspired by SSD, TextBoxes [29] and TextBoxes++ [30] detected text by directly predicting word bounding boxes with quadrilaterals via multiple Text-box layers at different scales. TextBoxes, a horizontal text detector, was extended to handle arbitrary-oriented text in TextBoxes++. EAST [31] followed DenseBox [32] to generate multiple channels of pixel-level text score maps and geometry and adopted the U-shaped design [33] to integrate features from different levels. RRPN and RRoI pooling [34] were proposed to generate rotated region proposals and project the proposals to feature maps. LOMO [35] came with an iterative refinement module to revise the predicted position of text instances and a shape expression module to obtain tight representation for irregular text by considering its geometry properties. Similarly, Wang *et al.* [36] proposed an arbitrary shape scene text detection method using RNN-based adaptive text region representation. *2. Sub-Text Components-based Methods:* Text detection is different from generic object detection in the homogeneity and locality of text instances; *i.e.*, any part of a text instance is still text.

- *Pixel-level Methods.* PixelLink [37] predicted whether two adjacent pixels belong to the same text instance by adding extra output channels to indicate links between adjacent pixels. Border learning [38] casted each pixel into three classes: text, border, and non-text, under the assumption that the border can well separate text instances. Shape-Aware Loss [39] was proposed to ease the separation of adjacent instances and detection of long or large instances, based on the scales and adjacency of text instances. PSENet [40] adopted a progressive scale expansion algorithm to gradually expand the detected instances from small to large and complete instances via multiple semantic segmentation maps.

- Component-level Methods. CTPN [41] developed a vertical anchor mechanism to jointly predict the location and text/non-text score of each fixed-width proposal, and a recurrence mechanism to connect sequential text proposals in the convolutional feature maps. SegLink [42] first decomposed text into two locally detectable elements: segment (*i.e.*, an oriented box covering partial word or text line) and link (*i.e.*, an indication of two adjacent segments' belong to the same word or text line), and then detected these two elements densely at multiple scales. CRAFT [43] localized the individual character regions and linked the detected characters to text instances. However, CRAFT required character-level annotations that were mostly unavailable in real-world datasets.

Scene Text Recognition. There are two major strategies in decoding text content from image encoded features from CNN, Connectionist Temporal Classification (CTC) [44] and the encoder-decoder framework [45]. *1. CTC-based Methods:* CRNN [46] stacked multiple bidirectional LSTMs on top of convolutional layers and used CTC for training and inference. Yin *et al.* [47] simultaneously detected and recognized characters by sliding the text line image with character models learned end-to-end on text line images labeled with text transcripts. *2. Encoder-Decoder Methods:* R^{2}AM [48] was a sequential attention-based modeling mechanism that performs "soft" deterministic image feature selection as the character sequence being decoded from RNNs for language modeling. Edit Probability (EP) [49] handled the misalignment between the ground-truth string and the attention's output sequence of the probability distribution, by considering the possible occurrences of missing/superfluous characters.

Scene Text Spotting. *1. Regular-shaped Text:* Li *et al.* [50, 51] proposed the first deep learning-based end-to-end trainable scene text spotting method by incorporating RoI Pooling [52] to join the detection and recognition stage, but it could only spot horizontal text. Deep TextSpotter [53] could handle multi-orientation text instances, but it didn't have a sharable feature. Thus, minimizing the recognition loss would have no influence on the former detection stage. End-to-End TextSpotter [54] and FOTS [55] adopted an anchor-free mechanism to improve both the training and inference speed. They use two similar sampling strategies, *i.e.*, Text-Alignment and RoI-Rotate, to extract features from arbitrary-oriented quadrilateral detection results. *2. Arbitrary-shaped Text:* Mask TextSpotter [56–58] used character-level supervision to simultaneously detect and recognize characters and instance masks. Nonetheless, the character-level ground truths are expensive, thus mostly unavailable for real data. RoI Masking [59] cropped out the features from the predicted axis-aligned rectangular bounding boxes and multiplied the features with the corresponding instance segmentation mask. TextDragon [60] proposed RoISlide to transform the whole text features into axis-aligned features indirectly by transforming each local quadrangle sequentially. As the first one-stage text spotting method, CharNet [61] directly outputted bounding boxes of words, with corresponding character labels. An iterative character detection was proposed to transfer the character detection capability learned from synthetic data to real-world images. ABCNet [62] adaptively fitted arbitrarily-shaped text by a parameterized Bezier curve and used BezierAlign layer to extract accurate convolution features. CRAFTS [63] used the character region feature from the detector as input character attention to the recognizer.

1.3.1.4 Text Reading in Videos

Text Detection & Tracking. Wang *et al.* [64] proposed a multi-scale feature sampling and warping network on adjacent frames, and an attention-based multi-frame feature aggregation mechanism to fuse the complementary text features from related frames. Wu *et al.* [65] explored Delaunay triangulation to detect and track texts. The triangular mesh pattern reflects text properties, such as regular spacing between characters and constant stroke width, thus distinguishable from non-text. Yang *et al.* [66] combined single-frame detection with cross-frame motion-based tracking. The text association was formulated into a cost-flow network. Tian *et al.* [67] located character candidates locally and searched text regions globally. Specifically, a multi-strategy tracking-based text detection approach [68] was used to globally search and select the best text region with dynamic programming. Wang *et al.* [69] proposed a fully convolutional model based on a novel refine block structure, which refines the low-resolution semantic features with the high-resolution low-level features.

End-to-End Text Spotting. Wang *et al.* [70] proposed a multi-frame tracking-based method, where text detection and recognition are done on each frame before recognized texts are tracked over the video sequence. FREE [71, 72] proposed a text recommender to select the highest-quality text from text streams for recognizing and released a large-scale video text spotting dataset.

1.3.2 Data Privacy

1.3.2.1 Privacy-Related Machine Learning Problems

According to a survey by Liu *et al.* [73], privacy-related machine learning problems could be categorized by the roles of machine learning: protection target, attack tool, and protection tool. *i) ML as protection target:* the privacy concerns include the model parameters and structures that serve as commercial and intellectual property. When the machine learning model is protected, the adversary can only query the model on the inputs through an interface. *ii) ML as attack tool:* the attack models include re-identification attacks and inference attacks on target users' private information. *iii) ML as a protection tool:* ML has been used for privacy risk assessment and private data release.

1.3.2.2 Data Privacy Protection in Computer Vision

With pervasive cameras for surveillance or smart home devices, visual data privacy has drawn increasing interest from industry and academia.

Differential Privacy Differential privacy [74–78] gives a rigorous cryptographically-motivated privacy guarantee for statistical databases that allows users to query statistical aggregates without leak of any individual record. Specifically, two datasets x and x' are neighbors if they have difference only in one element. Given a privacy loss budget $\epsilon > 0$ and a probability $\delta \in [0, 1]$, a randomized function $\tilde{\mathbf{A}} : \mathcal{X} \to \mathcal{Z}$ guarantees (ϵ, δ)-approximate differential privacy, if $\forall Z \subset \mathcal{Z}$

and $\forall$ paired x, x' that are neighbors of each other:

$$\mathbb{P}(\tilde{\mathbf{A}}(x) \in Z) \leq e^{\epsilon}\mathbb{P}(\tilde{\mathbf{A}}(x' \in Z)) + \delta. \tag{1.1}$$

If $\delta = 0$, $(\epsilon, 0)$-approximate differential privacy is shortened to ϵ-differential privacy. δ is interpreted as the probability of ϵ-differential privacy guarantee's failing. Intuitively speaking, differential privacy guarantees that even though some adversary knows the whole dataset x except for a single item, no more information could be obtained from the unknown item from the output of $\tilde{\mathbf{A}}$, thus making two datasets different in only one element statistically indistinguishable. A practice for turning a non-private function $\mathbf{A}$ into a private function $\tilde{\mathbf{A}}$ is perturbation by additive noise. Differential privacy can also be defined for function valued data [79] as output and possibly input. However, differential privacy is not resistant to inference attacks, as differential privacy only prevents any adversary from acquiring extra knowledge by adding or removing some individual record, and not from gaining knowledge from the released data itself. Preventing inference attacks is task-dependent and is given in expectation rather than in a deterministic way. Therefore its privacy guarantee is weaker than differential privacy [80].

Forgetting / Unlearning. Cao & Yang [81] introduced the problem of forgetting/unlearning as a remedy that restores the machine learning model's privacy, security, and usability with a minimum amount of effort. Ginart *et al.* [82] proposed two efficient data deletion solutions for k-means clustering with solid theoretical guarantees. DeltaGrad [83] was proposed for rapid and efficient retraining when training data has slight changes (*e.g.*, deletion or addition of samples) by differentiating the optimization path with the Quasi-Newton method, based on cached model parameters and gradients. Inspired from distributed training and ensemble learning, SISA (short for Sharded, Isolated, Sliced, and Aggregated training) [84] replaced the model to be learned several times where each replica (a.k.a. constituent model) was fed with a disjoint shard of the dataset, and there was no information flow between constituent models, *i.e.* no shared gradient. Influence function [85, 86] in robust statistics was an analytical tool that measured the effect (or "influence") of upweighting a training point by an infinitesimal amount on the value of a statistic without recalculation. Under the assumption of twice-differentiability and strictly convexity, Koh & Liang [87] scaled-up influence function to modern machine settings as an asymptotic approximation of leave-one-out retraining. Influence function estimated the effect of upweighting or perturbing a training sample in a closed-form. Linear filtration [88] sanitized classification models by hiding the information related to some classes for unlearning from the output logits, without removing the information in the full model weights.

Transmitting Feature Descriptors. A seemingly reasonable and computationally cheaper option is to extract feature descriptors from raw images and transmit those features only. Unfortunately, previous studies [89–93] revealed that considerable details of original images could still be recovered from standard HOG, SIFT, LBP, 3D point clouds, Bag-of-Visual-Words or neural network activations (even if they look visually distinctive from natural images).

Homomorphic Cryptographic Solutions. Most classical cryptographic solutions secure communication against unauthorized access from attackers. However, they are not immediately applicable to preventing authorized agents (such as the back-end analytics) from the unauthorized abuse of information, causing privacy breach concerns. A few encryption-based solutions, such as Homomorphic Encryption (HE) [94, 95], were developed to locally encrypt visual information. The server can only get access to the encrypted data and conduct a utility task on it. However, many encryption-based solutions will incur high computational costs at local platforms. It is also challenging to generalize the cryptosystems to more complicated classifiers. Chattopadhyay *et al.* [96] combined the detection of regions of interest and the real encryption techniques to improve privacy while allowing general surveillance to continue.

Anonymization by Empirical Obfuscations. An alternative approach towards a privacy-preserving vision system is based on the concept of anonymized videos. Such videos are intentionally captured or processed by empirical obfuscations to be in special low-quality conditions, which only allow for recognizing some target events or activities while avoiding the unwanted leak of the identity information for the human subjects in the video.

Ryoo *et al.* [97] showed that even at the extremely low resolutions, reliable action recognition could be achieved by learning appropriate downsampling transforms, with neither unrealistic activity-location assumptions nor extra specific hardware resources. The authors empirically verified that conventional face recognition easily failed on the generated low-resolution videos. Butler *et al.* [98] used image operations like blurring and superpixel clustering to get anonymized videos, while Dai *et al.* [99] used extremely low resolution (*e.g.*, 16×12) camera hardware to get anonymized videos. Winkler *et al.* [100] used cartoon-like effects with a customized version of mean shift filtering. Wang *et al.* [101] proposed a lens-free coded aperture (CA) camera system, producing visually unrecognizable and unrestorable image encodings. Pittaluga & Koppal [102, 103] proposed to use privacy-preserving optics to filter sensitive information from the incident light-field before sensor measurements are made, by k-anonymity and defocus blur. Earlier work of Jia *et al.* [104] explored privacy-preserving tracking and coarse pose estimation using a network of ceiling-mounted time-of-flight low-resolution sensors. Tao *et al.* [105] adopted a network of ceiling-mounted binary passive infrared sensors. However, both works [104, 105] handled only a limited set of activities performed at specific constrained areas in the room.

The usage of low-quality anonymized videos by obfuscations was computationally cheap and compatible with sensor's bandwidth constraints. However, the proposed obfuscations were not learned towards protecting any visual privacy, thus having limited effects. In other words, privacy protection came as a "side product" of obfuscation, and was not a result of any optimization, making the privacy protection capability very limited. What is more, the privacy-preserving effects were not carefully analyzed and evaluated by human study or deep learning-based privacy recognition approaches. Lastly, none of the aforementioned empirical obfuscations extended their efforts to study deep learning-based action recognition, making their task performance less competitive. Similarly, the recent progress of low-resolution object recognition [106–108] also put their privacy protection effects in jeopardy.

Learning-based Solutions. Very recently, a few learning-based approaches have been proposed to address privacy protection or fairness problems in vision-related tasks [3, 109–117]. Many of them exploited ideas from adversarial learning. They addressed this problem by learning data representations that simultaneously reduce the budget cost of privacy or fairness while maintaining the utility task performance.

Wu *et al.* [109] proposed an adversarial training framework dubbed Nuisance Disentangled Feature Transform (NDFT) to utilize the free meta-data (*i.e.*, altitudes, weather conditions, and viewing angles) in conjunction with associated UAV images to learn domain-robust features for object detection. Pittaluga *et al.* [111] preserved the utility by maintaining the variance of the encoding or favoring a second classifier for a different attribute in training. Bertran *et al.* [112] motivated the adversarial learning framework as a distribution matching problem and defined the objective and the constraints in mutual information. Roy & Boddeti [113] measured the uncertainty in the privacy-related attributes by the entropy of the discriminator's prediction. Oleszkiewicz *et al.* [118] proposed an empirical data-driven privacy metric based on mutual information to quantify the privatization effects on biometric images. Zhang *et al.* [114] presented an adversarial debiasing framework to mitigate the biases concerning demographic groups. Ren *et al.* [115] learned a face anonymizer in video frames while maintaining the action detection performance. Shetty *et al.* [116] presented an automatic object removal model that learns how to find and remove objects from general scene images via a generative adversarial network (GAN) framework.

1.3.2.3 Data Privacy Protection in Social Media/Photo Sharing

User privacy protection is also a topic of extensive interest in social media, especially for photo sharing. The most common means to protect user privacy in an uploaded photo is to add empirical obfuscations, such as blurring, mosaicing, or cropping out certain regions (usually faces) [119]. However, extensive research showed that such an empirical approach could be easily hacked [120, 121].

Adversarial Examples. Human-imperceptible privacy protection against machines [122, 123] described a game-theoretical system where the photo owner and the privacy attributes recognition model as attacker strived for antagonistic goals of disabling the inference attack, and imperceptible obfuscation could be learned from their competition. Shen *et al.* [123] proposed the "human sensitivity map" that visually encoded different levels of human sensitivity to visual changes within an image. The proposed sensitivity-aware perturbation on the image could prevent machines from predicting the suppressed attributes while preserving the targeted attributes (*e.g.*, "contain focused object", "aesthetic", and "pleasant"). However, their system was limited in three aspects. Firstly, the obfuscation was designed to confuse one specific recognition model via finding its adversarial perturbations. Fooling only one recognition model can cause obvious overfitting as merely changing to another recognition model will likely put the learning efforts in vain. Secondly, although the obfuscator can generate adversarial examples to confuse the attacker, the attacker can use the adversarial training to increase the robustness of the learned recognition model. Lastly, such model-specific perturbations cannot even protect privacy from human eyes. Thus, the problem setting in [122] differs from our target problem. Specifically, in social photo sharing, the privacy protection mechanism was expected to generate minimum perceptual quality loss to photos, so that the changes are imperceptible to humans. In comparison, there is no such constraint in our scenario. We can apply a more flexible and aggressive transformation (even degradation) to the image to achieve universal and model-agnostic privacy protection.

Privacy-Annotated Benchmarks. The visual privacy issues faced by blind people were revealed in VizWiz-Priv [124] that included images taken by blind people. PicAlert [125] was a privacy-aware image tool with privacy-oriented image search functionality. The privacy level was judged by the title, tags, and visual features. The associated dataset had $26,458$ images, among which $22,807$ were labeled as public and $3,651$ were private. YourAlert proposed a personalized privacy classification model, together with a dataset containing $1,511$ images. Concrete privacy attributes were defined in [4] with their correlation with image content. The authors categorized possible private information in images, and they ran a user study to understand privacy preferences. They then provided a sizable set of 22k images, dubbed as Visual Privacy (VISPR). VISPR is annotated with 68 privacy attributes, on which they trained privacy attributes predictors. Orekondy *et al.* [126] further curated VISPR to include 8.5k images and annotated them with pixel and instance level labels over 24 privacy attributes. The proposed approach of redaction by segmentation enabled users to selectively sanitize images of private content.

1.3.3 Implementation Efficiency

1.3.3.1 Compact Model Design

Deep Convolutional Neural Networks are often over-parameterized. Many works aim to reduce the model size and computational complexity, such as SqueezeNet [127], MobileNet [128, 129], ShuffleNet [130, 131], EfficientNet [132], and FBNet [133]. SqueezeNet proposed a fire module that first "squeezes" the network with 1×1 convolution filters and then expanded it with multiple 1×1 and 3×3 convolution filters. MobileNet V1 factorized the standard convolution into a depthwise convolution (*i.e.*, applied only on one of the input channels) followed by a pointwise convolution (*i.e.*, 1×1 filters). MobileNet V2 presented a inverted residual module with linear bottleneck block operator that is composed of three operators: a linear transformation for upsampling, a non-linear per-channel transformation, and a linear transformation for downsampling. ShuffleNet V1 utilized pointwise group convolution and channel shuffle to reduce computation cost while maintaining performance. ShuffleNet V2 derived multiple practical guidelines for efficient network design, which includes using balanced convolution, reducing group convolution, mitigating network fragmentation, and reducing element-wise operations. Unlike ResNet's scalablity in depth and MobileNet's scalability in width, EfficientNet [132] uniformly scaled all dimensions of depth/width/resolution using a compound coefficient. FBNet was discovered by a differentiable neural architecture search framework for ConvNet design.

1.3.3.2 Model Compression

Parameter Pruning Deep Compression [134] was a three-stage pipeline that prune the network by removing the redundant connections, quantize the weights by clustering for weight sharing, and apply Huffman coding to encode the code book. For pruning, there are five different levels of granularity: element-wise, channel-wise, shape-wise, filter-wise and layer-wise. Pruning has three steps: selecting parameters, pruning the neurons, and fine-tuning or retraining. Based on the observation that weights with higher magnitude have larger influence on the model's output compared with weights with smaller magnitude, magnitude-based pruning methods seek to identify unnecessary weights or features to remove them from runtime inference. Very early attempt of Optimal Brain Damage [135] and Optimal Brain Surgeon [136] reduced the number of connections based on the Hessian of the loss function. Although methods based on Hessian pruning deliver better performance than those pruned with only magnitude-based methods, computing the Hessian matrix during training for large models is infeasible as it has the complexity of $\mathcal{O}(W^2)$. Besides magnitude of weights, Average Percentage of Zeros (APoZ) [137] could be an indicator to prune the corresponding weights. In penalty-based pruning, a penalty value is used to update the weights to zero or near zero values. LASSO shrinks the least absolute valued feature's corresponding weights, thus increasing sparsity. Group LASSO [138] used a structured pruning method to remove entire groups of neurons while maintaining network structure. Structured Sparsity Learning [139] split weights into multiple groups based on geometry, computational complexity, and group sparsity. Network slimming [140] applied LASSO on the scaling factor of BN, so that setting the scaler by zero enabled channel-wise pruning. Sparse structure selection [141] applied LASSO to sparse scaling factors in neurons, groups, or residual blocks with an improved gradient descent method, dubbed as Accelerated Proximal Gradient (APG).

Quantization Quantization is known as the process of approximating a continuous signal distribution by a set of discrete values. Clustering and parameter sharing in Deep Compression [134] is a typical example of quantization. If categorized by whether considering quantization during training, quantization could be categorized into two areas: quantization-aware training and post-training quantization (PTQ). Moreover, quantization could be categorized by the granularity of data being grouped: layer-wise and channel-wise. Quantization could further be categorized based on bit-width.

1.3.3.3 Dynamic Inference

Dynamic neural networks [142], as opposed to static ones, can adapt the structures or computational graph to the input data during inference. The major advantage of a dynamic network is the capability of allocating computation on demand by selectively activating model components conditioned on the input. As a result, less computation cost is put on easy samples, and more cost is put on hard ones. Besides efficiency, dynamic networks are known for adaptiveness by achieving the desired trade-off between accuracy and efficiency in response to the different hardware platforms and application scenarios. Last but not least, making a neural network dynamic improves its interpretability. Specifically, it is possible to figure out the data-dependent activated components and observe what specific parts of the input are accountable for certain predictions.

1.3.4 Pipeline Simplicity

1.3.4.1 Transformers for Computer Vision

The Transformer architecture has become the *de-facto* standard for natural language processing tasks [143, 144] . Its end-to-end philosophy has led to significant advances in complex structured prediction tasks such as machine translation or speech recognition. Nevertheless, its popularity among computer vision tasks does not rise until recently [145–150]. The core of the Transformer is the self-attention mechanism, which characterizes the dependencies between any two distant tokens. It could be viewed as a special case of non-local operations in the embedded Gaussian [151], that captures long-range dependencies of pixels in image/video.

The recent work DETR [149] treats object detection as a direct set prediction problem. DETR is built upon the Transformer in an encoder-decoder structure, whose self-attention mechanism removes duplicate predictions by explicitly modeling all pairwise interactions between elements in a sequence. As a brand new paradigm for object detection, DETR simplifies the traditional object detector pipeline by removing handcrafted components, including anchor generation and non-maximum suppression.

Visual Transformers [150] represent images as a compact set of visual semantic tokens and apply Transformers to densely model relations between semantic concepts. Transformers' token-based image representation and processing outperform its convolutional counterparts on image classification and semantic segmentation tasks. Vision Transformer [152] directly applies a pure Transformer to a sequence of image patches for classification. Without any image-specific inductive biases (*e.g*. translation equivariance and locality), ViT approaches or beats state-of-the-art CNN models on the dataset at a larger scale than ImageNet [153]. For high-resolution image synthesis, Patrick *et al*. [154] uses CNN to learn a context-rich vocabulary of image constituents and Transformers to model their composition. The long-range dependencies within the compositions are captured by the expressive Transformer architecture.

TimeSformer [155] is a convolution-free video classification model that adapts the self-attention module in Transformer to enable spatiotemporal feature learning from a sequence of frame-level patches. The closest existing work to ours is the Video Action Transformer (VATN) [148]. VATN uses a Transformer-style architecture to aggregate features from the spatio-temporal context around the persons of interest as queries. However, it still relies on RPN to do person localization, and it does not use the Transformer decoder. In contrast, TxVAD exploits the Transformer's full encoder-decoder structure purely to do person localization and action classification, where the bounding boxes and labels are all organically decoded. Besides, VATN's person query is a RoI-Pooled feature over the center frame using a single bounding box, while TxVAD uses a spatio-temporal RoI-pooled feature over the entire video trunk (see Sec. 3).

2. DATA PRIVACY*

2.1 Motivation

Smart surveillance or smart home cameras, such as Amazon Echo and Nest Cam, are now found in millions of locations to link users to their homes or offices remotely. They provide the users with a monitoring service by notifying environment changes, a lifelogging service, and intelligent assistance. However, the benefits come at the heavy price of privacy intrusion from time to time. Due to the computationally demanding nature of visual recognition tasks, only some of the tasks can run on the resource-limited local devices, which makes transmitting (part of) data to the cloud necessary. Growing concerns have been raised [156–160] towards personal data uploaded to the cloud, which could be potentially misused or stolen by malicious third parties. Many laws and regulations in the United States and the European Union [161–164] also bring up guidelines for handling *Personally Identifiable Information*. This new privacy risk is fundamentally different from traditional concerns over unsecured transmission channels (*e.g.*, malicious third-party eavesdropping), and therefore requires new solutions to address it.

The goal is to alleviate privacy concerns without compromising user convenience. The dilemma is that we would like a camera system to assist daily human life by understanding its videos while preventing it from obtaining sensitive visual information (such as faces, gender, skin color, etc.) that can intrude individual privacy. Thus, it is a new problem to find a desired transform to obfuscate the captured raw visual data at the local end so that the transformed data will only enable specific target utility tasks while obstructing undesired privacy-related budget tasks. Recent approaches [97–99] intentionally captures or processes videos in extremely low-resolution to create privacy-preserving "anonymized" videos and showed promising empirical results.

*Part of this chapter is reprinted with permission from "Privacy-Preserving Deep Action Recognition: An Adversarial Learning Framework and A New Dataset" [1] by Z. Wu, H. Wang, Z. Wang, H. Jin, and Z. Wang, IEEE Transactions on PatternAnalysis and Machine Intelligence, Copyright 2020 by IEEE. DOI: https://doi.org/10.1109/TPAMI.2020.3026709. And part of this chapter is reprinted with permission from "Towards Privacy-Preserving Visual Recognition via Adversarial Training: A Pilot Study" [3] by Z. Wu, Z. Wang, Z. Wang, and H. Jin, Proceedings of the European Conference on Computer Vision, Pages 605-624, Copyright 2018 by Springer.

2.2 Method

2.2.1 Problem Definition

Objective. Assume our training data X (raw visual data captured by camera) are associated with a target utility task $\mathcal{T}$ and a privacy budget $\mathcal{B}$. Since $\mathcal{T}$ is usually a supervised task, *e.g.*, action recognition or visual tracking, a label set Y_T is provided on X, and a standard cost function L_T (*e.g.*, cross-entropy) is defined to evaluate the task performance on $\mathcal{T}$. Usually, there is a state-of-the-art deep neural network f_T, which takes X as input and predicts the target labels. On the other hand, we need to define a budget cost function J_B to evaluate its input data's privacy leakage: the smaller $J_B(\cdot)$ is, the less private information its input contains.

We seek an optimal anonymization function f_A^* to transform the original X to anonymized visual data $f_A^*(X)$, and an optimal target model f_T^* such that:

- f_A^* has filtered out the private information in X, *i.e.*,

$$J_B(f_A^*(X)) \ll J_B(X);$$

- the performance of f_T is minimally affected when using the anonymized visual data $f_A^*(X)$ compared to when using the original data X, *i.e.*,

$$L_T(f_T^*(f_A^*(X)), Y_T) \approx \min_{f_T} L_T(f_T(X), Y_T).$$

To achieve these two goals, we mathematically formulate the problem as solving the following optimization problem:

$$f_A^*, f_T^* = \underset{f_A, f_T}{\mathrm{argmin}}[L_T(f_T(f_A(X)), Y_T) + \gamma J_B(f_A(X))]. \tag{2.1}$$

Definition of J_B and L_T. The definition of the privacy budget cost J_B is not straightforward. Practically, it needs to be placed in concrete application contexts, often in a task-driven way. For example, in smart workplaces or smart homes with video surveillance, one might often want to avoid disclosure of the face or identity of persons. Therefore, reducing J_B could be interpreted as suppressing the success rate of identity recognition or verification. Other privacy-related attributes, such as race, gender, or age, can be similarly defined too. We denote the privacy-related annotations (such as identity label) as Y_B, and rewrite $J_B(f_A(X))$ as $J_B(f_B(f_A(X)), Y_B)$, where f_B denotes the privacy budget model which takes (anonymized or original) visual data as input and predicts the corresponding private information. Different from L_T, minimizing J_B will encourage $f_B(f_A(X))$ to diverge from Y_B. Without loss of generality, we assume both f_T and f_B to be classification models and output class labels. Under this assumption, we choose both L_T and L_B as the cross-entropy function, and J_B as the negative cross-entropy function:

$$J_B \triangleq -H(Y_B, f_B(f_A(X))),$$

where $H(\cdot, \cdot)$ is the cross-entropy function.

Two Challenges. Such a supervised, task-driven definition of J_B poses at least two challenges: (1) *Dataset challenge:* The privacy budget-related annotations, denoted as Y_B, often have less availability than target utility task labels. Specifically, it is often challenging to have both Y_T and Y_B available on the same X; (2) $\forall$ *challenge:* Considering the nature of privacy protection, it is not sufficient to merely suppress the success rate of one f_B model. Instead, we define a privacy prediction function family

$$\mathcal{P} : f_A(X) \mapsto Y_B,$$

so that the ideal privacy protection by f_A should be reflected as *suppressing every possible model* f_B from $\mathcal{P}$. That differs from the common supervised training goal, where only one model needs to be found to fulfill the target utility task successfully.

We address the *dataset challenge* by two ways: (1) cross dataset training and evaluation (Section 2.2.4); and more importantly (2) building a new dataset annotated with both utility and privacy labels (Section 2.2.5). We defer their discussion to respective experimental paragraphs.

Handling the $\forall$ *challenge* is more challenging. Firstly, we re-write the general form in Eq. (2.1) with the task-driven definition of J_B as follows:

$$f_A^*, f_T^* = \underset{(f_A, f_T)}{\operatorname{argmin}} [L_T(f_T(f_A(X)), Y_T) + \gamma \sup_{f_B \in \mathcal{P}} J_B(f_B(f_A(X)), Y_B)]. \tag{2.2}$$

The $\forall$ *challenge* is the infeasibility to directly solve Eq. (2.2), due to the infinite search space of f_B in $\mathcal{P}$. Secondly, we propose to solve the following approximate problem by setting f_B as a neural network with a fixed structure:

$$f_A^*, f_T^* = \underset{(f_A, f_T)}{\operatorname{argmin}} [L_T(f_T(f_A(X)), Y_T) + \gamma \max_{f_B} J_B(f_B(f_A(X)), Y_B)]. \tag{2.3}$$

Lastly, we propose "model ensemble" and "model restarting" (Section 2.2.6) to handle the $\forall$ *challenge* better and boost the experimental results further.

Considering the $\forall$ *challenge*, the evaluation protocol for privacy-preserving action recognition is more intricate than traditional action recognition task. We propose a two-step protocol (as described in Section 2.2.7) to evaluate f_A^* and f_T^* on the trade-off they have achieved between target task utility and privacy protection budget.

Solving the Minimax. Solving Eq. (2.4) is still challenging because the minimax problem is hard by its nature. Traditional minimax optimization algorithms based on alternating gradient descent can only find minimax points for convex-concave problems, and they achieve sub-optimal solutions on deep neural networks since they are neither convex nor concave. Some very recent minimax algorithms, such as K-Beam [165], have been shown to be promising in none convex-concave and deep neural network applications. However, these methods rely on heavy parameter tuning and are effective only in limited situations. Besides, our optimization goal in Eq. (2.4) is even harder than common minimax objectives like those in GANs, which are often interpreted as a two-party competition game. In contrast, our Eq. (2.4) is more "hybrid" and can be interpreted as a more complicated three-party competition, where (adopting machine learning security terms) f_A is an obfuscator, f_T is a utilizer collaborating with the obfuscator, and f_B is an attacker trying to breach the obfuscator. Specifically,

$$\min_{(f_A,f_T)} \max_{f_B} [L_T(f_T(f_A(X)), Y_T) + \gamma J_B(f_B(f_A(X)), Y_B)]. \tag{2.4}$$

Therefore, we see no obvious best choice from the off-the-shelf minimax algorithms to achieve our objective.

We are thus motivated to try different state-of-the-art minimax optimization algorithms on our framework. We tested two state-of-the-art minimax optimization algorithms, namely GRL [166] and K-Beam [165], on our framework and proposed an innovative entropy maximization method to solve Eq. (2.4). We empirically show our entropy maximization algorithm outperforms both state-of-the-art minimax optimization algorithms and discuss its advantages. In Section 2.2.3, we present the comparison of three methods and hope it will benefit future research on similar problems.

2.2.2 Basic Framework

Pipeline. Our framework is a privacy-preserving action recognition pipeline that uses video data as input. It is a prototype of the in-demand privacy protection in smart camera applications. Figure 2.1 depicts the basic framework implementing the proposed formulation in Eq. (2.4). The framework consists of three parts: the anonymization model f_A, the target utility model f_T, and the privacy budget model f_B. f_A takes raw video X as input, filters out private information in X, and outputs the anonymized video $f_A(X)$. f_T takes $f_A(X)$ as input and carries out the target utility task. f_B also take $f_A(X)$ as input and try to predict the private information from $f_A(X)$. All three models are implemented with deep neural networks, and their parameters are learnable during the training procedure. The entire pipeline is trained under the guidance of the hybrid loss of L_T and J_B. The training procedure has two goals. The first goal is to find an optimal anonymization model f_A^* that can filter out the private information in the original video while keeping useful information for the target utility task. The second goal is to find a target model that can achieve good performance on the target utility task using anonymized videos $f_A^*(X)$. Similar frameworks have been used

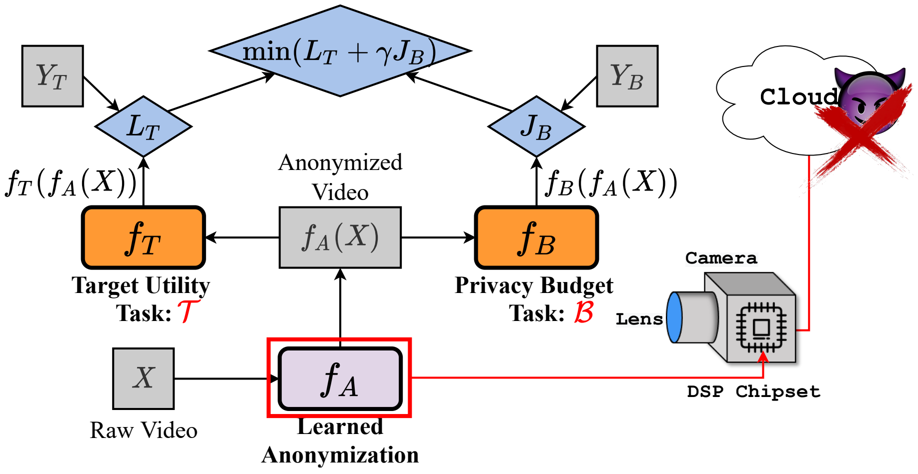


Figure 2.1: Basic adversarial training framework for privacy-preserving action recognition. Modified from [1].

in feature disentanglement [167–170]. After training, the learned anonymization model can be applied on a local device (*e.g.*, smart camera), by designing an embedded chipset responsible for the anonymization at the hardware-level [115]. We can convert raw video to anonymized video locally and only transfer the anonymized video through the Internet to the backend (*e.g.*, cloud) for target utility task analysis. The private information in the raw videos will be unavailable on the backend.

Implementation. Specifically, f_A is implemented using the model in [171], which can be taken as a 2D convolution-based frame-level filter. In other words, f_A converts each frame in X into a feature map of the same shape as the original frame. We use state-of-the-art human action recognition model C3D [16] as f_T and state-of-the-art image classification models, such as ResNet [172] and MobileNet [128], as f_B. Since the action recognition model we use is C3D, we need to split the videos into clips with a fixed frame number. Each clip is a 4D tensor of shape $[T, W, H, C]$, where T is the number of frames in each clip and W, H, C are the width, height, and the number of color channels in each frame respectively. Unlike f_T, which takes a 4D tensor as an input data sample, f_B takes a 3D tensor (*i.e.*, a frame) as input. We average[1] the logits over the temporal

[1]AVERAGING the logits temporally gave a better performance in privacy budget prediction of J_B, compared with MAXING the logits.

dimension of each video clip to calculate J_B and predict the budget task label.

2.2.3 Optimization Strategies

In the following algorithms, we denote θ_B as the parameters of f_B. Similarly, f_A and f_T are parameterized by θ_A and θ_T respectively. $\alpha_A, \alpha_B, \alpha_T$ are learning rates used to update $\theta_A, \theta_B, \theta_T$. th_B and th_T are accuracy thresholds for the target utility task and the privacy budget prediction. max_iter is the maximum number of iterations. For simplicity concern, we abbreviate $L_T(f_T(f_A(X)), Y_T)$, $L_B(f_B(f_A(X)), Y_B)$, and $J_B(f_B(f_A(X)), Y_B)$ as $L_T(\theta_A, \theta_T)$, $L_B(\theta_A, \theta_B)$, and $J_B(\theta_A, \theta_B)$[2] respectively. Acc is a function to compute accuracy on the privacy budget and the target utility tasks, given training data (X^t, Y_B^t and Y_T^t) or validation data (X^v, Y_B^v and Y_T^v).

2.2.3.1 Gradient reverse layer (GRL)

We consider Eq. (2.4) as a minimax problem [173]:

$$\theta_A^*, \theta_T^* = \underset{(\theta_A, \theta_T)}{\operatorname{argmin}} L(\theta_A, \theta_T, \theta_B^*),$$

$$\theta_B^* = \underset{\theta_B}{\operatorname{argmax}} L(\theta_A^*, \theta_T^*, \theta_B),$$

where $L(\theta_A, \theta_T, \theta_B) = L_T(\theta_A, \theta_T) + \gamma J_B(\theta_A, \theta_B) = L_T(\theta_A, \theta_T) - \gamma L_B(\theta_A, \theta_B)$.

GRL [166] is a state-of-the-art algorithm to solve such a minimax problem. The underlying mathematical gist is to solve the problem by alternating minimization:

$$\theta_A \leftarrow \theta_A - \alpha_A \nabla_{\theta_A}(L_T(\theta_A, \theta_T) - \gamma L_B(\theta_A, \theta_B)), \tag{2.5a}$$

$$\theta_T \leftarrow \theta_T - \alpha_T \nabla_{\theta_T} L_T(\theta_A, \theta_T), \tag{2.5b}$$

$$\theta_B \leftarrow \theta_B - \alpha_B \nabla_{\theta_B} L_B(\theta_A, \theta_B). \tag{2.5c}$$

We denote this method as **GRL** in the following parts and give the details in Algorithm 1.

[2]Remember that J_B is the negative cross-entropy by definition.

Algorithm 1: GRL Algorithm

```
1  Initialize $\theta_A$, $\theta_T$ and $\theta_B$;
2  for $t_0 \leftarrow 1$ to max_iter do
3      Update $\theta_A$ using Eq. (2.5a)
4      while $Acc(f_T(f_A(X^v)), Y_T^v) \leq th_T$ do
5          Update $\theta_T$ using Eq. (2.5b)
6      end
7      while $Acc(f_B(f_A(X^t)), Y_B^t) \leq th_B$ do
8          Update $\theta_B$ using Eq. (2.5c)
9      end
10 end
```

2.2.3.2 Alternating optimization of two loss functions

The goal in Eq. (2.4) can also be formulated as alternatingly solving the following two optimization problems:

$$\theta_A^*, \theta_T^* = \underset{(\theta_A, \theta_T)}{\operatorname{argmin}} \; L_T(\theta_A, \theta_T), \tag{2.6a}$$

$$\theta_B^*, \theta_A^* = \underset{\theta_B}{\operatorname{argmin}} \, \underset{\theta_A}{\operatorname{argmax}} \; L_B(\theta_A, \theta_B). \tag{2.6b}$$

Eq. (2.6a) is a standard minimization problem which can be solved by end-to-end training f_A and f_T. Eq. (2.6b) is a minimax problem which we solve by state-of-the-art minimax optimization method "K-Beam" [165]. K-Beam method keeps track of K different sets of budget model parameters (denoted as $\{\theta_B^i\}_{i=1}^K$) during training time, and alternatingly updates θ_A and $\{\theta_B^i\}_{i=1}^K$.

Inspired by K-Beam method, we present the following parameter update rules to alternatingly solve the two loss functions in Eq. (2.6):

$$\theta_A, \theta_T \leftarrow \theta_A, \theta_T - \alpha_T \nabla_{(\theta_T, \theta_A)} L_T(\theta_A, \theta_T), \tag{2.7a}$$

$$j \leftarrow \underset{i \in \{1,\ldots,K\}}{\operatorname{argmin}} L_B(\theta_A, \theta_B^i), \tag{6b-i}$$

$$\theta_A \leftarrow \theta_A + \alpha_A \nabla_{\theta_A} L_B(\theta_A, \theta_B^j), \tag{6b-ii}$$

$$\theta_B^i \leftarrow \theta_B^i - \alpha_B \nabla_{\theta_B^i} L_B(\theta_A, \theta_B^i), \ \forall i \in \{1, \ldots, K\}. \tag{6c}$$

We denote this method as **Ours-K-Beam** in the following parts and give the details in Algorithm 2, where d_iter is the number of iterations used in the step of maximizing L_B.

Algorithm 2: Ours-K-Beam Algorithm

```
Initialize $\theta_A$, $\theta_T$ and $\{\theta_B^i\}_{i=1}^K$;
for $t_0 \leftarrow 1$ to max_iter do
    /*$L_T$ step:*/
    while $Acc(f_T(f_A(X^v)), Y_T^v) \leq th_T$ do
        Update $\theta_T$, $\theta_A$ using Eq. (2.7a)
    end
    /*$L_B$ Max step:*/
    Update $j$ using Eq. (6b-i)
    for $t_1 \leftarrow 1$ to d_iter do
        Update $\theta_A$ using Eq. (6b-ii)
    end
    /*$L_B$ Min step:*/
    for $i \leftarrow 1$ to K do
        while $Acc(f_B^i(f_A(X^t)), Y_B^t) \leq th_B$ do
            Update $\theta_B^i$ using Eq. (6c)
        end
    end
end
```

2.2.3.3 Maximize entropy

We empirically find that minimizing negative cross-entropy J_B, which is a *concave* function, causes numerical instabilities in Eq. (2.5a). So, we replace J_B with $-H_B$, the negative entropy of $f_B(f_A(X))$, which is a *convex* function[3]:

$$H_B(f_B(f_A(X))) \triangleq H(f_B(f_A(X))),$$

where $H(\cdot)$ is the entropy function. Minimizing $-H_B$ is equivalent to maximizing entropy, which will encourage "uncertain" predictions. We replace J_B in Eq. (2.5a) by $-H_B$, abbreviate $H_B(f_B(f_A(X)))$ as $H_B(\theta_A, \theta_B)$, and propose the following new update scheme:

$$\theta_A \leftarrow \theta_A - \alpha_A \nabla_{\theta_A}(L_T(\theta_A, \theta_T) - \gamma H_B(\theta_A, \theta_B)), \tag{2.8a}$$

$$\theta_T, \theta_A \leftarrow \theta_T, \theta_A - \alpha_T \nabla_{\theta_T, \theta_A} L_T(\theta_A, \theta_T), \tag{2.8b}$$

$$\theta_B \leftarrow \theta_B - \alpha_B \nabla_{\theta_B} L_B(\theta_A, \theta_B), \tag{2.8c}$$

where L_T and L_B are still cross-entropy loss functions as in Eq. (2.5). Unlike in Eq. (2.5b), where we only update θ_T when minimizing L_T, we train θ_T and θ_A in an end-to-end manner as shown in Eq. (2.8b), since we find it achieves better performance in practice.

We denote this method as **Ours-Entropy** in the following parts and give the details in Algorithm 3.

2.2.4 Addressing the Dataset Challenge by Cross-Dataset Training and Evaluation: An Initial Attempt

An ideal dataset to train and evaluate our framework would be a set of human action videos with both action labels and privacy attributes provided. On the SBU dataset, we can use the actor pair as a simple privacy attribute. But when we want to evaluate our method on more complex

[3]This point discusses the convexity or concavity of different loss functions when viewing them as the outermost function in the composite function. Both loss functions are neither convex nor concave w.r.t. model weights.

Algorithm 3: Ours-Entropy Algorithm

```
1  Initialize $\theta_A$, $\theta_T$ and $\theta_B$;
2  for $t_0 \leftarrow 1$ to max_iter do
3      Update $\theta_A$ using Eq. (2.8a)
4      while $Acc(f_T(f_A(X^v)), Y_T^v) \leq th_T$ do
5          Update $\theta_T$, $\theta_A$ using Eq. (2.8b)
6      end
7      while $Acc(f_B(f_A(X^t)), Y_B^t) \leq th_B$ do
8          Update $\theta_B$ using Eq. (2.8c)
9      end
10 end
```

privacy attributes, we run into the dataset challenge: in the literature, no existing datasets have both human action labels and privacy attributes provided on the same videos.

Given the observation that a privacy attributes predictor trained on VISPR can correctly identify privacy attributes occurring in UCF101 and HMDB51 videos (examples in the Appendix C), we hypothesize that the privacy attributes have good "transferability" across UCF101/HMDB51 and VISPR. Therefore, we can use a privacy prediction model trained on VISPR to assess the privacy leak risk on UCF101/HMDB51.

In view of that, we propose to use cross-dataset training and evaluation as a workaround method. In brief, we train action recognition (target utility task) on human action datasets, such as UCF101 [11] and HMDB51 [7], and train privacy protection (budget task) on visual privacy dataset VISPR [4], while letting the two interact via their shared component - the learned anonymization model. More specifically, during training, we have two pipelines: one is f_A and f_T trained on UCF101 or HMDB51 for action recognition; the other is f_A and f_B trained on VISPR to suppress multiple privacy attribute prediction. The two pipelines share the same parameters for f_A. During the evaluation, we evaluate model utility (*i.e.*, action recognition) on the testing set of UCF101 or HMDB51 and privacy protection performance on the testing set of VISPR. Such cross-dataset training and evaluation shed new possibilities on training privacy-preserving recognition models, even under the practical shortages of datasets that have been annotated for both tasks. Notably, "cross-dataset training" and "cross-dataset testing (or evaluation)" are two independent strategies used in this paper; they can be used either together or separately. Details of our three experiments (SBU, UCF-101, and HMDB51) are explained as follows:

- SBU (Section 4.1): we train and evaluate our framework on the same video set by considering actor identity as a simple privacy attribute. *Neither cross-training nor cross-evaluation is involved.*

- UCF101 (Section 4.2): we perform *both cross-training and cross-evaluation*, on UCF-101 + VISPR. Such a method provides an alternative to flexibly train and test privacy-preserving video recognition for different utility/privacy combinations, without annotating specific datasets.

- HMDB51 (Section 5.5), we use cross-training on HMDB51 + VISPR datasets similarly to the UCF-101 experiment; but for testing, we evaluate both utility and privacy performance on the same, newly-annotated PA-HMDB51 testing set. Therefore, it involves *cross-training, but no cross-evaluation*.

Beyond the above initial attempt, we further construct a new dataset dedicated to the privacy-preserving action recognition task, which will be presented in Section 2.2.5.

2.2.5 Addressing the Dataset Challenge by PA-HMDB51: A New Benchmark

2.2.5.1 Motivation

There is no public dataset containing both human action and privacy attribute labels on the same videos in the literature. This poses two challenges. Firstly, the lack of available datasets has increased the difficulty in employing a data-driven joint training method. Secondly, this complication has made it impossible to directly evaluate the trade-off between privacy budget and action utility achieved by a learned anonymization model f_A^*. To solve this problem, we annotate and present the very first human action video dataset with privacy attributes labeled, named **PA-HMDB51** (**P**rivacy-**A**nnotated HMDB51). We evaluate our method on this newly built dataset and

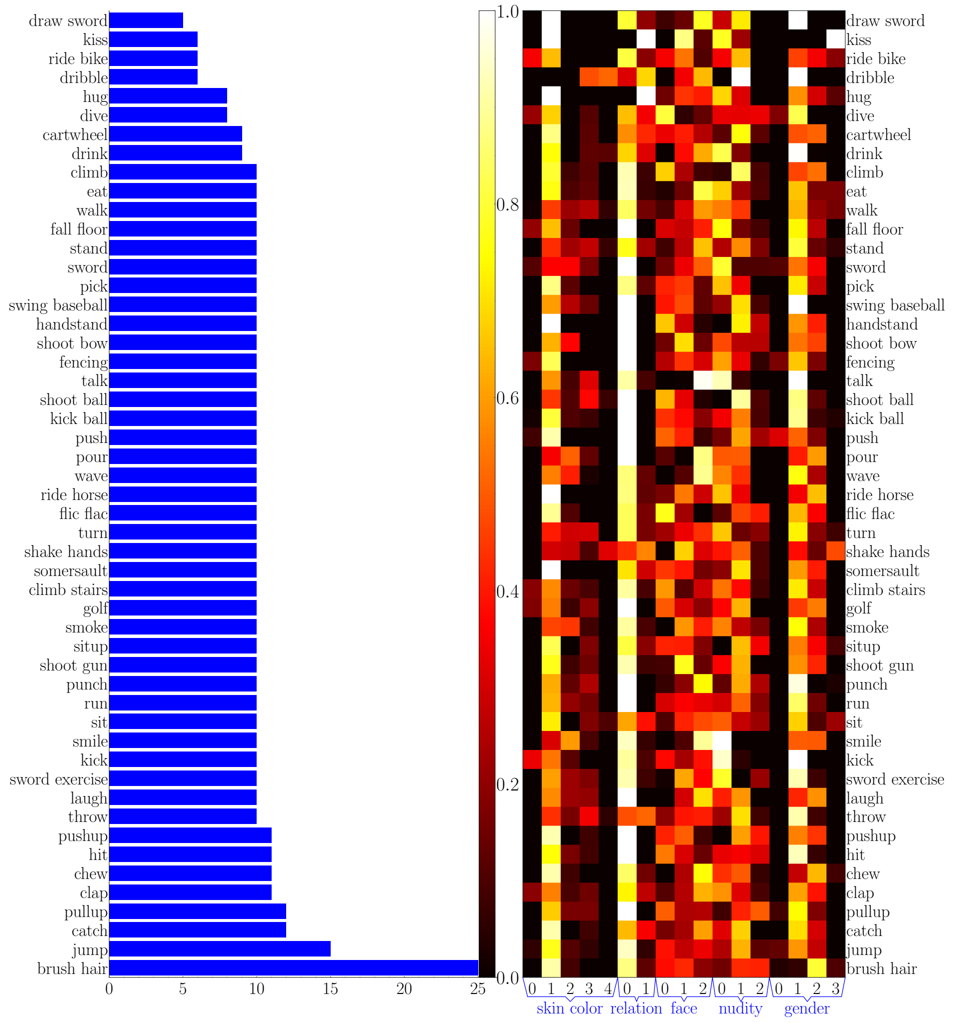


Figure 2.2: **Left**: action distribution of PA-HMDB51. Each bar shows the number of videos with a certain action. *E.g.*, the last bar shows there are 25 "brush hair" videos in the PA-HMDB51 dataset; **Right**: action-attribute correlation in the PA-HMDB51 dataset. The x-axis are all possible values grouped by bracket for each privacy attribute. The y-axis are different action types. The color represents ratio of the number of frames of some action annotated with a specific privacy attribute value w.r.t. the total number of frames of the action. Reprinted from [1].

further demonstrate our method's effectiveness.

2.2.5.2 *Selecting and Labeling Privacy Attributes*

A recent work [4] has defined 68 privacy attributes that could be disclosed by images. However, most of them seldom make any appearance in public human action datasets. We carefully select 5 privacy attributes that are most relevant to our smart home settings out of the 68 attributes from [4]:

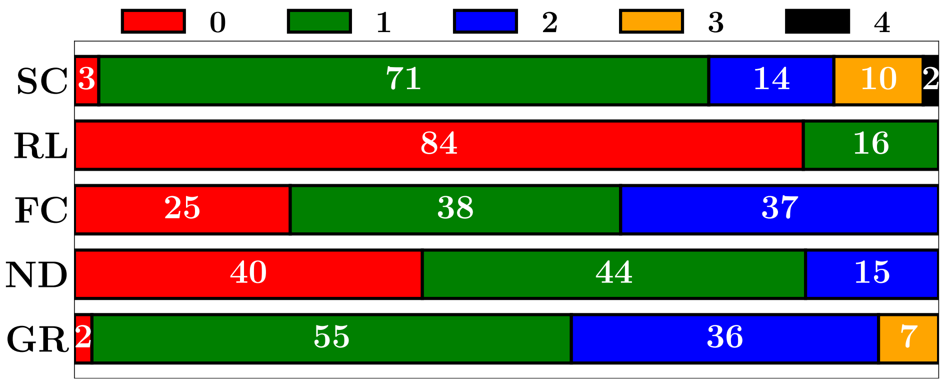


Figure 2.3: Label distribution per privacy attribute in the PA-HMDB51. SC, RL, FC, ND, and GR stand for skin color, relationship, face, nudity, and gender, respectively. The rounded ratio numbers are shown as white text (in $\%$ scale). Definitions of label values $(0, 1, 2, 3, 4)$ for each attribute are described in Table 2.1. Reprinted from [1].

skin color, gender, face, nudity, and personal relationship (only intimate relationships such as friends, couples, or family members are considered in our setting). The detailed description of each attribute, their possible ground truth values, and their corresponding meanings are listed in Table 2.1. Some annotated frames in our PA-HMDB51 dataset are shown in Table 2.2 as examples.

Privacy attributes may vary during the video clip. For example, in some frames, we may see a person's full face, while in the next frames, the person may turn around, and his/her face is no longer visible. Therefore, we decide to label all the privacy attributes on each frame [4].

The annotation of privacy labels was manually performed by a group of students at the CSE department of Texas A&M University. Each video was annotated by at least three individuals and then cross-checked.

2.2.5.3 HMBD51 as the Data Source

Now that we have defined the 5 privacy attributes, we need to identify a source of human action videos for annotation. There are a number of choices available, such as [6, 7, 11, 174, 175]. We choose HMDB51 [7] to label privacy attributes since it consists of more diverse private information, especially nudity/semi-nudity and relationship.

[4]A tiny portion of frames in some HMDB51 videos do not contain any person. No privacy attributes are annotated on those frames.

We provide a per-frame annotation of the selected 5 privacy attributes on 515 videos selected from HMDB51. In this paper, we treat all 515 videos as testing samples[5]. Our ultimate goal would be to create a larger-scale version of PA-HMDB51 that allows for both training and testing coherently on the same benchmark. For now, we use PA-HMDB51 to facilitate better testing, while still considering cross-dataset training as a rough yet useful option to train privacy-preserving video recognition (before the larger dataset becomes available).

2.2.5.4 Dataset Statistics

Action Distribution. When selecting videos from the HMDB51 dataset, we consider two criteria on action labels. First, the action labels should be balanced. Second (and more implicitly), we select more videos with non-trivial privacy labels. For example, "brush hair" action contains many videos with a "semi-nudity" attribute, and "pull-up" action contains many videos with a "partially visible face" attribute. Despite their practical importance, these privacy attributes are relatively less seen in the entire HMDB51 dataset, so we tend to select more videos with these attributes, regardless of their action classes. The resultant distribution of action labels is depicted in Figure 2.2 (left panel), showing a relative class balance.

Privacy Attribute Distribution. We try to make the label distribution for each privacy attribute as balanced as possible by manually selecting those videos containing uncommon privacy attribute values in original HMDB51 to label. For instance, videos with semi-nudity are overall uncommon, so we deliberately select those videos containing semi-nudity into our PA-HMDB51 dataset. Naturally, people are reluctant to release data that contains privacy concerns to the public, so the privacy attributes are highly unbalanced in any public video datasets. Although we have used this method to reduce the data imbalance, the PA-HMDB51 is still unbalanced. Frame-level label distributions of all 5 privacy attributes are shown in Figure 2.3.

Action-Attribute Correlation. If there was a strong correlation between a privacy attribute and an

[5]Labeling per-frame privacy attributes on a video dataset is extremely labor-consuming and subjective (needing individual labeling then cross-checking). As a result, the current size of PA-HMDB51 is limited. So far, we have only used PA-HMDB51 as the testing set, and we seek to annotate more data and hopefully expand PA-HMDB51 for training as future work.

Table 2.1: Attribute definition in the PA-HMDB51 dataset

Attribute	Possible Values & Meaning
Skin Color	0: Skin color of the person(s) is/are *unidentifiable*. 1: Skin color of the person(s) is/are *white*. 2: Skin color of the person(s) is/are *brown/yellow*. 3: Skin color of the person(s) is/are *black*. 4: Persons with different skin colors are *coexisting*.[6]
Face	0: *Invisible* ($< 10\%$ area is visible). 1: *Partially visible* ($\geq 10\%$ and $\leq 70\%$ area is visible). 2: *Completely visible* ($> 70\%$ area is visible).
Gender	0: The gender(s) of the person(s) is/are *unidentifiable*. 1: The person(s) is/are *male*. 2: The person(s) is/are *female*. 3: Persons with different genders are *coexisting*.
Nudity	0: *No-nudity* with long sleeves and pants. 1: *Partial-nudity* with short sleeves, skirts, or shorts. 2: *Semi-nudity* with half-naked body.
Relationship	0: Personal relationship is *unidentifiable*. 1: Personal relationship is *identifiable*.

action, it would be harder to remove the private information from the videos without much harm to the action recognition task. For example, we would expect a high correlation between the attribute "gender" and the action "brush hair" since this action is carried out much more often by females than by males. We show the correlation between privacy attributes and actions in Figure 2.2 (right panel) and more details in Appendix D.

2.2.6 Addressing the $\forall$ Challenge by Privacy Budget Model Restarting and Ensemble

To improve the generalization ability of learned f_A over all possible $f_B \in \mathcal{P}$ (*i.e.*, privacy cannot be reliably predicted by any model), we hereby discuss two simple and easy-to-implement options. Other more sophisticated model re-sampling or model search approaches, such as [176],

[6]For "skin color" and "gender," we allow multiple labels to coexist. For example, if a frame showed a black person's shaking hands with a white person, we would label "black" and "white" for the "skin color" attribute. In the visualization, we use "coexisting" to represent the multi-label coexistence and we don't show in detail whether it is "white and black coexisting" or "black and yellow coexisting." For the remaining three attributes, we label each attribute using the highest privacy-leakage risk among all persons in the frame. *E.g.*, given a frame where a group of people are hugging, if there is at least one complete face visible, we would label the "face" attribute as "completely visible."

Table 2.2: Examples of the annotated frames in the PA-HMDB51 dataset. Reprinted from [1].

Frame	Action	Privacy Attributes
	Brush hair	• skin color: white • face: invisible • gender: female • nudity: semi-nudity • relationship: unrevealed
	Situp	• skin color: black • face: completely visible • gender: male • nudity: semi-nudity • relationship: unrevealed

will be explored in future work.

2.2.6.1 Privacy Budget Model Restarting

Motivation. The max step over $J_B(f_B(f_A(X)), Y_B)$ in Eq. (2.4) leads to the optimizer being stuck in bad local solutions (similar to "mode collapse" in GANs), that will hurdle the entire minimax optimization. Model restarting provides a mechanism to "bypass" the bad solution when it occurs, thus enabling the minimax optimizer to explore a better solution.

Approach. At a certain point of training (*e.g.*, when the privacy budget $L_B(f_B(f_A(X)), Y_B)$ stops decreasing any further), we re-initialize f_B with random weights. Such a random restarting aims to avoid trivial overfitting between f_B and f_A (*i.e.*, f_A is only specialized at confusing the current f_B), without requiring more parameters. We then start to train the new model f_B to be a strong competitor, w.r.t. the current $f_A(X)$: specifically, we freeze the training of f_A and f_T, and change to minimizing $L_B(f_B(f_A(X)), Y_B)$, until the new f_B has been trained from scratch to become a strong privacy prediction model over current $f_A(X)$. We then resume adversarial training by unfreezing f_A and f_T, as well as switching the loss for f_B back to the adversarial loss (negative entropy or negative cross-entropy). Such a random restarting can repeat multiple times.

2.2.6.2 Privacy Budget Model Ensemble

Motivation. Ideally in Eq. (2.4) we should maximize the error over the "current strongest possible" attacker f_A from $\mathcal{P}$ (a large and continuous f_B family), over which searching/sampling is impractical. Therefore we propose a privacy budget model ensemble as an approximation strategy, where we approximate the continuous $\mathcal{P}$ with a discrete set of M sample functions. Such a strategy is empirically verified in Section 4 and 5 to address the critical "$\forall$ Challenge" in privacy protection, *i.e.*, enhancing the defense against unseen attacker models (compared to the clear "attacker overfitting" phenomenon when sticking to one f_A during training).

Approach. Given the budget model ensemble $\bar{\mathcal{P}}_t \triangleq \{f_B^i\}_{i=1}^M$, where M is the number of f_Bs in the ensemble during training, we turn to minimize the following discretized surrogate of Eq. (2.2):

$$\begin{aligned} f_A^*, f_T^* = \underset{f_A, f_T}{\operatorname{argmin}} [L_T(f_T(f_A(X)), Y_T) + \\ \gamma \max_{f_B^i \in \bar{\mathcal{P}}_t} J_B(f_B^i(f_A(X)), Y_B)]. \end{aligned} \tag{2.9}$$

The previous basic framework is a special case of Eq. (2.9) with $M = 1$. The ensemble strategy can be easily incorporated with restarting.

2.2.6.3 Incorporate Budget Model Restarting and Budget Model Ensemble with Ours-Entropy

Budget Model Restarting and Budget Model Ensemble can be easily incorporated with all three optimization schemes described in Section 2.2.3. We take Ours-Entropy as an example here. When model ensemble is used, we abbreviate $L_B(f_B^i(f_A(X)), Y_B)$ and $H_B(f_B^i(f_A(X)))$ as $L_B(\theta_A, \theta_B^i)$ and $H_B(\theta_A, \theta_B^i)$ respectively. The new parameter updating scheme is:

$$\theta_A \leftarrow \theta_A - \alpha_A \nabla_{\theta_A}(L_T + \gamma \max_{\theta_B^i \in \bar{\mathcal{P}}_t} -H_B(\theta_A, \theta_B^i)), \tag{2.10a}$$

$$\theta_A, \theta_T \leftarrow \theta_A, \theta_T - \alpha_T \nabla_{(\theta_A, \theta_T)} L_T(\theta_A, \theta_T), \tag{2.10b}$$

$$\theta_B^i \leftarrow \theta_B^i - \alpha_B \nabla_{\theta_B^i} L_B(\theta_A, \theta_B^i),\ \forall i \in \{1, \ldots, M\}. \tag{2.10c}$$

We only suppress the model f_B^i with the largest privacy leakage $-H_B$, *i.e.*, the "most confident" one about its current privacy prediction, when updating the anonymization model f_A. But we still update all M budget models on the budget task. The formal description of Ours-Entropy with model restarting and ensemble is given in Algorithm 4, where $\{\theta_B^i\}_{i=1}^M$ is reinitialized every $rstrt\_iter$ iterations. Likewise, GRL and Our-K-Beam can also be incorporated with restarting and ensemble.

Algorithm 4: Ours-Entropy Algorithm (with Model Restarting and Model Ensemble)

```
1  Initialize $\theta_A$, $\theta_T$ and $\{\theta_B^i\}_{i=1}^M$;
2  for $t_0 \leftarrow 1$ to max_iter do
3      if $t \equiv 0 \pmod{rstrt\_iter}$ then
4          Reinitialize $\{\theta_B^i\}_{i=1}^M$
5      end
6      Update $\theta_A$ using Eq. (2.10a)
7      while $Acc(f_T(f_A(X^v)), Y_T^v) \leq th_T$ do
8          Update $\theta_T, \theta_A$ using Eq. (2.10b)
9      end
10     for $i \leftarrow 1$ to $M$ do
11         while $Acc(f_B^i(f_A(X^t)), Y_B^t) \leq th_B$ do
12             Update $\theta_B^i$ using Eq. (2.10c)
13         end
14     end
15 end
```

2.2.7 Two-Step Evaluation Protocol

The solution to Eq. (2.2) gives an anonymization model f_A^* and a target utility task model f_T^*. We need to evaluate f_A^* and f_T^* on the trade-off they have achieved between target task utility and privacy protection in two steps: (1) whether the learned target utility task model maintains satisfactory performance on anonymized videos; (2) whether the performance of an *arbitrary* privacy prediction model on anonymized videos will deteriorate.

Suppose we have a training dataset X^t with target and budget task ground truth labels Y_T^t and

Y_B^t, and an evaluation dataset X^e with target and budget task ground truth labels Y_T^e and Y_B^e. In the first step, when evaluating the target task utility, we should follow the traditional routine: compare $f_T^*(f_A^*(X^e))$ with Y_T^e to get the evaluation accuracy on the target utility task, denoted as A_T, which we expect to be as high as possible. In the second step, when evaluating the privacy protection, it is insufficient if we only observe that the learned f_A^* and f_B^* lead to poor classification accuracy on X^e, because of the $\forall$ challenge: the attacker can select *any privacy budget model* to steal private information from anonymized videos $f_A^*(X^e)$. To empirically verify that f_A^* prohibits reliable privacy prediction for other possible budget models, we propose a novel procedure:

- We randomly re-sample N privacy budget prediction models $\bar{\mathcal{P}}_e \triangleq \{f_B^i\}_{i=1}^N$ from $\mathcal{P}$ for evaluation. Note that these N models used in evaluation $\bar{\mathcal{P}}_e$ have no overlap with the M privacy budget model ensemble $\bar{\mathcal{P}}_t$ used in training, *i.e.*, $\bar{\mathcal{P}}_e \cap \bar{\mathcal{P}}_t = \emptyset$.

- We train these N models $\bar{\mathcal{P}}_e$ on anonymized training videos $f_A^*(X^t)$ to make correct predictions on private information, *i.e.*, $\min_{f_B^i} L_B(f_B^i(f_A^*(X^t)), Y_B^t), \forall f_B^i \in \bar{\mathcal{P}}_e$. Note that f_A^* is fixed during this training procedure.

- After that, we apply each f_B^i on anonymized evaluation videos $f_A^*(X^e)$ and compare the outputs $f_B^i(f_A^*(X^e))$ with Y_B^e to get privacy budget accuracy of the i-th budget model, *i.e.*, $Acc(f_B^i(f_A^*(X^e)), Y_B^e)$.

- We select the highest accuracy among all N privacy budget models and use it as the final privacy budget accuracy A_B^N, which we expect to be as low as possible. Specifically, we have

$$A_B^N = \max_{f_B^i \in \bar{\mathcal{P}}_e} Acc(f_B^i(f_A^*(X^e)), Y_B^e). \tag{2.11}$$

2.3 Experiments

We show the effectiveness of our proposed adversarial training framework on *privacy-preserving action recognition* on existing human action datasets.

Overview of Experiment Settings. The target utility task is human action recognition, since it is a highly demanded feature in smart home and smart workplace applications. Experiments are carried out on three widely used human action recognition datasets: SBU Kinect Interaction Dataset [177], UCF101 [11] and HMDB51 [7]. The privacy budget task varies in different settings. In the SBU dataset experiments, the privacy budget is to prevent the videos from leaking human identity information. In the experiments on UCF101 and HMDB51, the privacy budget is to protect visual privacy attributes as defined in [4]. We emphasize that the general framework proposed in Section 2.2.2 can be used for a large variety of target utility tasks and privacy budget task combinations, not only limited to the aforementioned settings.

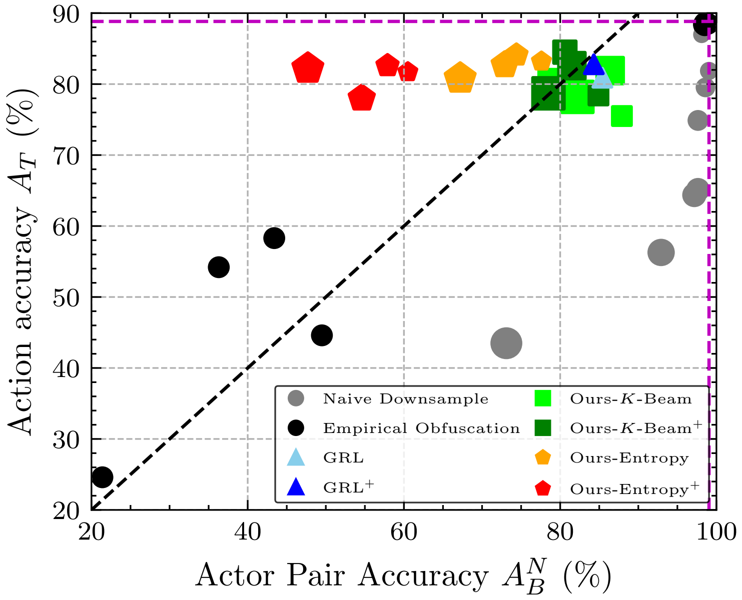


Figure 2.4: The trade-off between privacy budget and action utility on SBU dataset. For *Naive Downsample* method, a larger marker means a larger adopted downsampling rate. For *Ours-K-Beam* method, a larger marker means a larger K (number of beams) in Algorithm 2. For *Ours-Entropy* and *Ours-Entropy (restarting)*, a larger marker means a larger M (number of ensemble models) in Algorithm 4. Methods with "+" superscript are incorporated with model restarting. Vertical and horizontal purple dashed lines indicate A_B^N and A_T on the original non-anonymized videos, respectively. The black dashed line indicates where $A_B^N = A_T$. Reprinted from [1].

Following the notations in Section 2.2.2, on all the video action recognition datasets including SBU, UCF101 and HMDB51, we set $W = 112$, $H = 112$, $C = 3$, and $T = 16$ (C3D's required temporal length and spatial resolution). Note that the original resolution for SBU, UCF101 and

HDMB51 are 640×480, 320×240 and 320×240, respectively. We downsample video frames to resolution 160×120. To reduce the spatial resolution to 112×112, we use random-crop and center-crop in training and evaluation, respectively.

Baseline Approaches. We consistently use two groups of approaches as baselines across the three action recognition datasets. These two groups of baselines are *naive downsamples* and *empirical obfuscations*. The group of *naive downsamples* chooses downsample rates from $\{1, 2, 4, 8, 16\}$, where 1 stands for no down-sampling. The group of *empirical obfuscations* includes approaches selected from different combinations in {box, segmentation} $\times$ {blurring, blackening} $\times$ {face, human body}. Details are listed below:

- *Naive Downsamle*: Spatially downsample each frame.
- *Box-Black-Face*: Boxing and blackening faces.
- *Box-Black-Body*: Boxing and blackening bodies.
- *Seg-Black-Face*: Segmenting and blackening faces.
- *Seg-Black-Body*: Segmenting and blackening bodies.
- *Box-Blur-Face*: Boxing and blurring faces.
- *Box-Blur-Body*: Boxing and blurring bodies.
- *Seg-Blur-Face*: Segmenting and blurring faces.
- *Seg-Blur-Body*: Segmenting and blurring bodies.

Our Proposed Approaches. The previous two groups of baselines are compared with our proposed three approaches:

- *GRL*: as described in Section 2.2.3.1.
- *Ours-K-Beam*: as described in Section 2.2.3.2. We have tried $K = 1, 2, 4, 8$.
- *Ours-Entropy*: as described in Section 2.2.3.3. In the privacy budget model ensemble $\bar{\mathcal{P}}_t$, the M models are chosen from MobileNet-V2 [129] family with different width multipliers. We have tried $M = 1, 2, 4, 8$.

All three approaches are evaluated with and without privacy budget model restarting.

Evaluation. In the two-step evaluation (as described in Section 2.2.7), we have used $N = 10$ different state-of-the-art classification networks, namely ResNet-V1-{50,101} [172], ResNet-V2-{50,101} [178], Inception-V1 [179], Inception-V2 [180], and MobileNet-V1-{0.25,0.5,0.75,1} [128], as $\bar{\mathcal{P}}_e$. Note that $\bar{\mathcal{P}}_e \cap \bar{\mathcal{P}}_t = \emptyset$.

2.3.1 Identity-Preserving Action Recognition on SBU: Single-Dataset Training

We compare our proposed approaches with the groups of baseline approaches to show our methods' significant superiority in balancing privacy protection and model utility. We use three different optimization schemes described in Section 2.2.3 on our framework and empirically show all three largely outperform the baseline methods. We also show that adding the model ensemble and model restarting, as described in Section 2.2.6, to the optimization procedure can further improve the performance of our method.

2.3.1.1 Experiment Setting

SBU Kinect Interaction Dataset [177] is a two-person interaction dataset for video-based action recognition. 7 participants performed actions, and the dataset is composed of 21 sets. Each set uses different pairs of actors to perform all 8 interactions. However, some sets use the same two actors but with different actors acting and reacting. For example, in set 1, actor 1 is acting, and actor 2 is reacting; in set 4, actor 2 is acting, and actor 1 is reacting. These two sets have the same actors, so we combine them as one class to better fit our experimental setting. In this way, we combine all sets with the same actors and finally get 13 different actor pairs. This dataset's target utility task is action recognition, which could be taken as a classification task with 8 different classes. The privacy budget task is to recognize the actor pairs of the videos, which could be taken as a classification task with 13 different classes.

2.3.1.2 Implementation Details

In Algorithms 1-3, we set step sizes $\alpha_T = 10^{-5}$, $\alpha_B = 10^{-2}$, $\alpha_A = 10^{-4}$, accuracy thresholds $th_T = 85\%$, $th_B = 99\%$ and $max\_iter = 800$. In Algorithm 2, we set d_iter to be 30. In Algorithm 4, we set $rstrt\_iter$ to be 100. Other hyper-parameters of Algorithm 4 are identical with those in Algorithm 3. We set γ=2 in Eq. (2.4) and use Adam optimizer [181] to update all parameters.

2.3.1.3 Results and Analyses

We present the experimental results of our proposed methods and other baseline methods in Figure 2.4, which shows the trade-off between the action recognition accuracy A_T, and the actor pair recognition accuracy A_B^N. In order to interpret this figure, we should note that a desirable trade-off should incur maximal target accuracy A_T (y-axis) and minimal privacy budget accuracy A_B^N (x-axis). Therefore, a point closer to the top-left corner represents an anonymization model f_A^* with a more desirable performance. The magenta dotted line suggests the target accuracy A_T on original unprotected videos. This can be roughly considered as the A_T upper bound for all privacy protection methods, under the assumption that f_A^* will unavoidably filter out some useful information for the target utility task.

As we can see, Ours-K-Beams, Ours-Entropy, and GRL all largely outperform the two groups of naive baselines. {box, segmentation} $\times$ {blurring, blackening} $\times$ {face} and naive downsample with a low rate (*e.g.*, 2 and 4) can lead to decent action accuracy, but the privacy budget accuracy A_B^N is still very high, meaning these methods fail to protect privacy. On the other hand, {box, segmentation} $\times$ {blurring, blackening} $\times$ {body} and naive downsample with a high rate (*e.g.*, 8 and 16) can effectively suppress A_B^N to a low level, but A_T also suffers a huge negative impact, which means the anonymized videos are of little practical utility. Our methods, in contrast, achieve a great balance between utility and privacy protection. Ours-Entropy can decrease A_B^N by around 30% with nearly no harm on A_T.

Comparison of three methods. K-Beam is a state-of-the-art minimax optimization problem, and we apply it to solve a sub-problem (*i.e.*, Eq. (2.6b)) of our more complex three-party competition game. Unfortunately, we empirically find the K-Beam algorithm becomes more unstable when we introduce a new competing party to the minimax optimization problem. GRL is originally proposed for domain adaptation. On our new visual privacy protection task, we find it unstable and sensitive to model initialization. By replacing the *concave* negative cross-entropy loss function with the *convex* negative entropy, Ours-Entropy empirically stabilizes the optimization and gives the best performance among all three methods.

The results also show the effectiveness of model restarting and model ensemble: model restarting can further suppress A_B^N with little harm on A_T, and model ensemble with larger M can improve the trade-off even more.

2.3.2 Action Recognition on UCF101 with Multiple Privacy Attributes Protected: Cross-Dataset Training and Evaluation

2.3.2.1 Experiment Setting

UCF101 is an action recognition dataset with 13,320 real-life human action videos collected from YouTube. It contains videos of 101 different actions. We use the official train-test split for this dataset.

The target utility task $\mathcal{T}$ is to do human action recognition on UCF101, which can be taken as a video classification task with 101 classes.

VISPR is a dataset with $22,167$ images annotated with 68 privacy attributes, *e.g.*, semi-nudity, hobbies, face, race, gender, skin color, and so on. Each attribute of an image is labeled as "present" or "non-present" depending on whether the specific

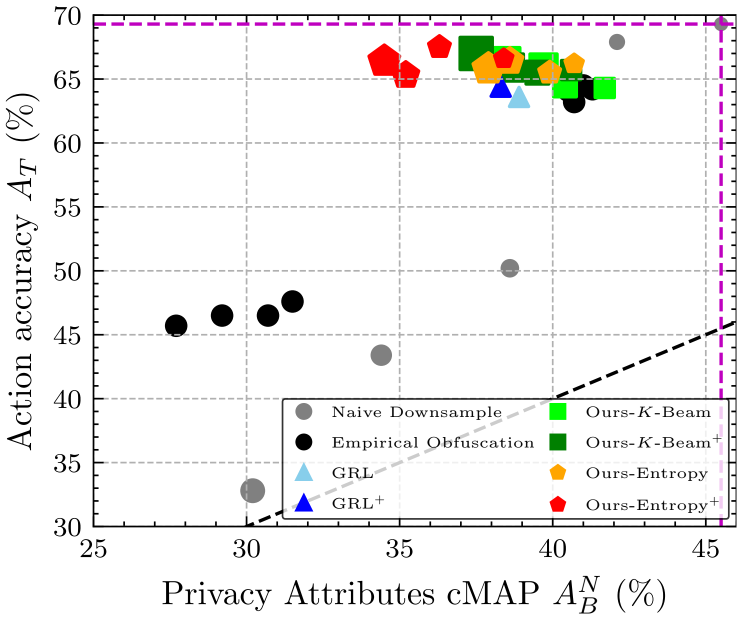


Figure 2.5: The trade-off between privacy budget and action utility on UCF-101/VISPR Dataset. For *Naive Downsample* method, a larger marker means a larger down sampling rate is adopted. For *Ours-K-Beam* method, a larger marker means a larger K (number of beams) in Algorithm 2. For *Ours-Entropy* and *Ours-Entropy (restarting)*, a larger marker means a larger M (number of ensemble models) in Algorithm 4. Methods with "+" superscript are incorporated with model restarting. Vertical and horizontal purple dashed lines indicate A_B^N and A_T on the original non-anonymized videos, respectively. The black dashed line indicates where $A_B^N = A_T$. Reprinted from [1].

privacy attribute information is contained in the image. Among the 68 attributes, there are 7 attributes that frequently appear in both UCF101 datasets and the smart home videos. Therefore we select these 7 attributes for protection in our experiments. These 7 attributes are semi-nudity, occupation, hobbies, sports, personal relationship, social relationship, and safe.

The privacy budget task $\mathcal{B}$ is to predict privacy attributes on the VISPR dataset, which can be taken as a multi-label image classification task (7 labels, each is a binary classification task). We adopt the class-based mean average precision (cMAP) [4] as A_B^N to measure the performance of the privacy budget task. The official train-test split is used on the VISPR dataset.

2.3.2.2 Experiment Setting

We train our models using cross-dataset training on HMDB51 and VISPR datasets as we did in Section 2.3.2, except that we use the 5 attributes defined in Table 2.1 on VISPR instead of the 7 used in Section 2.3.2. The trained models are directly evaluated on the PA-HMDB51 dataset[7] for both target utility task $\mathcal{T}$ and privacy budget task $\mathcal{B}$, without any re-training or adaptation. We exclude the videos in the PA-HMDB51 from the HMDB51 to get the training set. Similar to the UCF101 experiments, the target utility task $\mathcal{T}$ (*i.e.*, action recognition) can be taken as a video classification problem with 51

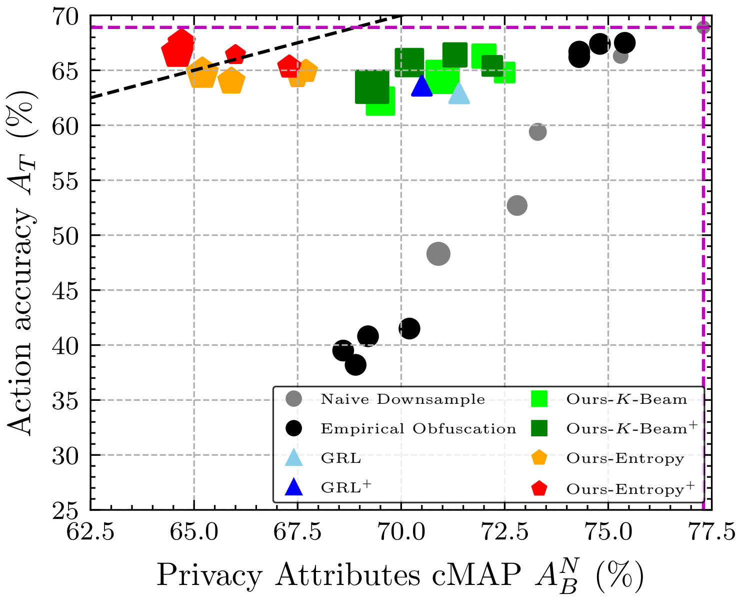


Figure 2.6: The trade-off between privacy budget and action utility on PA-HMDB51 Dataset. For *Naive Downsample* method, a larger marker means a larger downsampling rate is adopted. For *Ours-K-Beam* method, a larger marker means a larger K (number of beams) in Algorithm 2. For *Ours-Entropy* and *Ours-Entropy (restarting)*, a larger marker means a larger M (number of ensemble models) in Algorithm 4. Methods with "+" superscript are incorporated with model restarting. Vertical and horizontal purple dashed lines indicate A_B^N and A_T on the original non-anonymized videos, respectively. The black dashed line indicates where $A_B^N = A_T$. Reprinted from [1].

[7]We only use PA-HMDB51 as the testing set so far, since the current size of PA-HMDB51 is limited for training.

classes, and the privacy budget task $\mathcal{B}$ (*i.e.*, privacy attribute prediction) can be taken as a multi-label image classification task with two classes for each privacy attribute label. Notably, although PA-HMDB51 has provided concrete multi-class labels with specific privacy attribute classes, we convert them into binary labels during testing. For example, for "gender" attribute, we have provided ground truth labels "male," "female," "coexisting," and "cannot tell," but we only use "can tell" and "cannot tell" in our experiments, via combining "male," "female" and "coexisting" into the one class of "can tell." This is because we must keep the testing protocol on PA-HMDB51 consistent with the training protocol on VISPR (a multi-label, "either-or" type *binary* classification task, so that our models cross-trained on UCF101-VISPR can be evaluated directly. We hope to extend training to PA-HMDB51 in the future so that the privacy budget task can be formulated and evaluated as a multi-label classification problem.

All implementation details are identical with the UCF101 case, except that we adjust $th_T = 0.7$ and $th_B = 0.95$.

2.3.2.3 Implementation Details

In Algorithm 4, we set step sizes $\alpha_T = 10^{-5}$, $\alpha_B = 10^{-2}$, $\alpha_A = 10^{-4}$, accuracy thresholds $th_T = 70\%$, $th_B = 99\%$, $max\_iter = 800$ and $rstrt\_iter = 100$. We set $\gamma = 0.5$ in Eq. (2.4) and use Adam optimizer to update all parameters. Values of K and M are identical to those in SBU experiments.

2.3.2.4 Results and Analyses

We present the experimental results in Figure 2.5. All naive downsample and empirical obfuscation methods cause A_T to drop dramatically while A_B^N only drops a little bit, which means the utility of videos is greatly reduced while the private information is hardly filtered out. In contrast, with the help of model restarting and model ensemble, Ours-Entropy can decrease A_B^N by 7% while keeping A_T as high as that on the original raw videos, meaning the privacy is protected at almost no cost on the utility. Hence, Ours-Entropy outperforms all naive downsample and empirical obfuscation baselines in this experiment. It also shows an advantage over GRL and Ours-K-Beam.

2.3.3 Benchmark Results on PA-HMDB51: Cross-Dataset Training

2.3.3.1 Results and Analysis

The results on PA-HMDB51 are shown in Figure 2.6. Our methods achieve much better trade-off between privacy budget and action utility compared with baseline methods. When $M = 4$, our methods can decrease privacy cMAP by around 8% with little harm to utility accuracy. Overall, the privacy gains are more limited compared to the previous two experiments, because no (re-)training is performed; but the overall comparison trends show the same consistency.

Asymmetrical Privacy Attributes Protection Cost. Different privacy attributes have different protection costs. After applying the learned anonymization optimized by *Ours-Entropy (restarting, M=4)* on PA-HMDB51, the drop in AP of "face" is much more significant than "gender," which indicates that the "gender" attribute is much harder to suppress than "face." Such observation agrees that the gender attribute can be revealed by face, body, clothing, and even hairstyle. In future work, we will take such cost asymmetry into account by using a weighted loss combination of different privacy attributes or training a dedicated privacy protector for the most informative private attribute.

Human Study on the Privacy Protection of Our Learned Anonymization. We use a human study to evaluate the trade-off between privacy budget and action utility achieved by our learned anonymization transform. We take both privacy protection and action recognition into account in the study. We emphasize here that both privacy protection and action recognition are evaluated on the video level. There are 515 videos distributed on 51 actions in the PA-HMDB51. For each action in the PA-HMDB51, we randomly pick one video for the human study. Among the 51 selected videos, we only keep 30 videos to reduce the human evaluation cost. There were 40 volunteers involved in the human study. In the study, they were asked to label all the privacy attributes and the action type on the raw videos and the anonymized videos. According to the experimental results the actions in the anonymized videos are still distinguishable to humans, but the privacy attributes are not recognizable at all. This human study further justifies that our learned anonymization

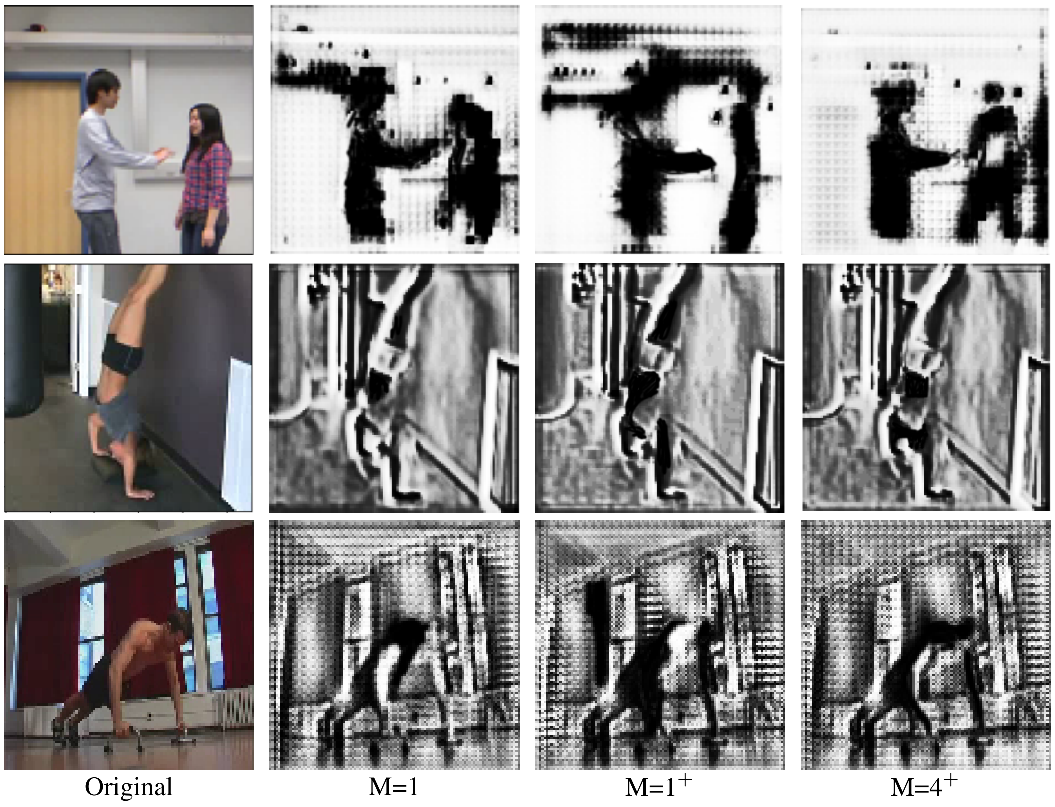


Figure 2.7: The center frame of example videos before (column 1) and after (columns 2-4) applying the anonymization transform learned by Ours-Entropy. The first row shows a frame from a “pushing” video in the SBU dataset; the second row shows a frame from a “handstand” video in the UCF101 dataset; the third row shows a frame from a “push-up” video in the PA-HMDB51 dataset. Privacy attributes in the last two rows include semi-nudity, face, gender, and skin color. Model restarting and ensemble settings are indicated below each anonymized image. M is the number of ensemble models. Methods with a “+” superscript are incorporated with model restarting. Reprinted from [1].

transform can protect privacy and maintain target utility task performance simultaneously.

2.3.4 Anonymized Video Visualization

We provide the visualization of the anonymized videos on SBU, UCF101, and our new dataset PA-HMDB51 (see Section 2.2.5) in Figure 2.7. To save space, we only show the center frame of each anonymized video. The visualization shows that the privacy attributes in the anonymized videos are filtered out, but it is still possible to recognize the actions.

3. IMPLEMENTATION EFFICIENCY[*]

3.1 Motivation

UAV-based video text spotting is broadly applied in assistive navigation, automatic translation, road sign recognition, industrial monitoring, and disaster response, etc. A standard video text spotting model has four components: text detector, text recognizer, text tracker, and post-processing.

Existing video text spotting solutions are purely performance-driven and fail to take energy consumption into account. Multi-frame-related features are first obtained in frame-wise detection or tracking. Then, they are aggregated for enhancement in a cross-frame and a multi-scale way for text recognition. Therefore, existing performance-driven solutions are high in energy consumption and unsuitable for resource-constrained UAV platforms.

[*]Part of this chapter is reprinted with permission from "E^2VTS: Energy-Efficient Video Text Spotting From Unmanned Aerial Vehicles" [2] by Z. Hu, P. Pi, Z. Wu, Y. Xue, J. Shen, J. Tan, X. Lian, Z. Wang, and J. Liu, Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops, Pages 905-913, 2021.

3.2 E^{2}VTS: An Energy-Efficient Video Text Spotting Solution

Overview. The E^{2}VTS two-step text spotting system adopts Efficient and Accurate Scene Text Detector (EAST) as the text detector, and Convolutional Recurrent Neural Network (CRNN) as the text recognizer. The recognizer is connected with the detector via crop & resize. A multi-stage image processor is proposed to further save energy consumption. It has three stages, selecting the highest-quality frame in a sliding window, rejecting text-free images and cropping non-text regions, and rejecting out-of-distribution images. The pipeline is shown in Figure 3.1,

3.2.1 Revisiting RCNN: Crop & Resize vs. Aligned RoI Pooling

We compare two connection mechanisms for the detector and the recognizer: crop+resize versus aligned RoI pooling. Examples of aligned RoI pooling include BezierAlign [62] for arbitrary-shaped text and RoIRotate [55] for rotated text.

Given the predicted bounding box from the detector, in crop+resize, the input to the recognizer is the cropped box area affinely transformed from the original image and resized to a fixed resolution. The detector and the recognizer are trained independently. In aligned RoI pooling, the input to the recognizer is the cropped box area affinely transformed from the feature map. The detector and the recognizer are trained jointly. Note that the text recognition loss uses the ground truth text regions instead of predicted text regions.

Unlike the benchmarks in image-based text spotting, real-world videos for text spotting are full of small size and poor-quality text boxes. Consequently, crop+resize outperforms aligned RoI pooling for two reasons. First, aligned RoI pooling losses the discriminative details for small size text boxes due to the deep convolutions in the detector. In contrast, crop+resize enlarges the input resolution of small size text boxes and preserves their discriminative spatial details [26]. Second, text recognition (*i.e.*, knowing what the text is) is intrinsically more difficult than text detection (*i.e.*, knowing where the text is). Thus, feature sharing and joint training will lead to sub-optimal performance for both tasks [182, 183].

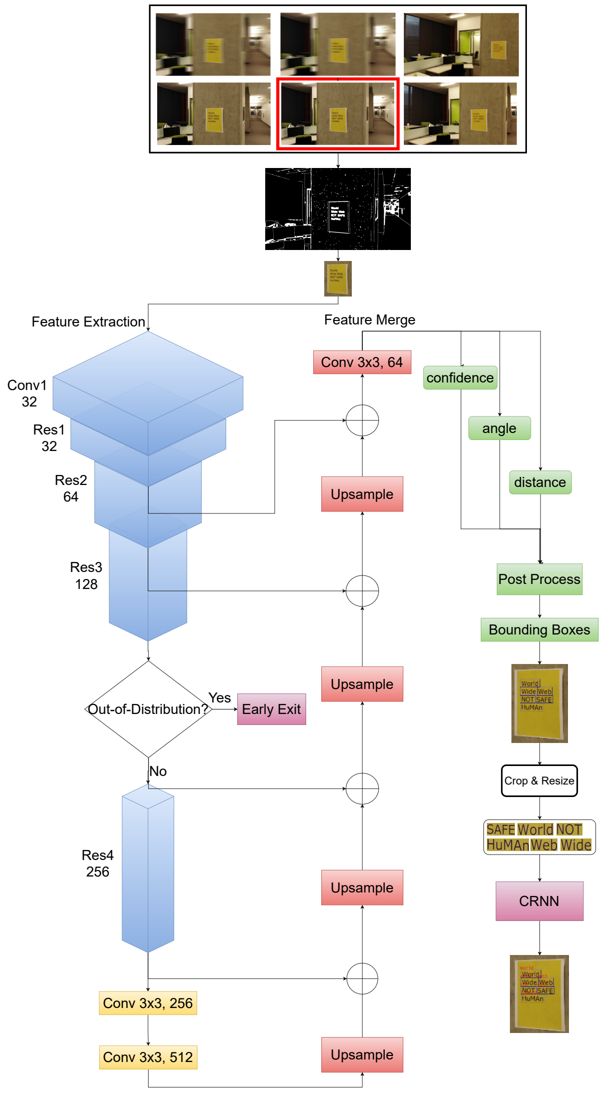


Figure 3.1: Overview: E^{2}VTS consists of two components. Component one is a multi-stage image processor which selects the best frame within a window size and crops out the background. Component two is a two-step crop & resize text spotting system including an EAST detector and a CRNN recognizer. The EAST detector is based on ResNet34 backbone and outputs confidence, angle, and distance. Out-of-distribution frames are rejected at ResNet Layer3. Reprinted from [2].

3.2.2 Multi-Stage Image Processor

Different from a single image, video frames are redundant and continuous in the temporal domain. Comparing one frame with its precedents and successors over certain metrics is a natural filtering process to select the most suitable frame for the later detection task. We also leverage sharp transitions of text regions to remove non-text background preliminarily to further boost efficiency. All these implementations are based on simple signal processing algorithms, which are significantly faster than neural network models.

3.2.2.1 Stage I: Selecting the Highest-Quality Frame in a Sliding Window

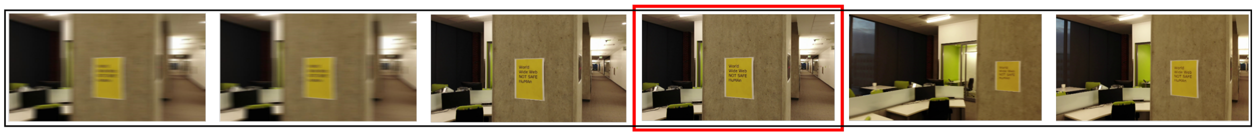


Figure 3.2: Sliding window for highest-quality frame selection. A window iterator is sliding over the temporally sub-sampled frames and quality scoring is conducted on the frames via the proposed measure. The highest ranked frame is selected. Reprinted from [2].

Problem Definition. Blur is the major artifact in UAV captured videos due to camera shake, depth variation, object motion or a combination of them [184, 185]. Among all the frames describing the same visual scene, the clearest image gives the least amount of detector or recognition error. Since blurred frames contain less energy in the high-frequency components, in their associated power spectrum [186], the power tends to fall much faster with increasing frequency, compared with clear frames. Therefore, the average of the power spectra of clear frames is higher than these degraded ones, as degraded ones have a steeper slope on their power spectrum.

Implementation. In Fig. 3.2, we propose a sliding window mechanism and select the highest-quality frame in each window. Given a video containing L frames, the i-th window $\mathcal{W}_i$ is obtained via:

$$\mathcal{W}_i = S_{(i,N)}(I_1, ..., I_L), \tag{3.1}$$

where S represents the sliding rule and N is the window size. The selected highest-quality frame in $\mathcal{W}_i$ is:

$$I_{HQ} = \underset{I \in W_i}{\operatorname{argmax}}\, \mathcal{G}(I), \tag{3.2}$$

where $\mathcal{G}$ is the quality measure. We propose two measures in this work: variance of Laplacian [187] and average fast Fourier transform (FFT) magnitude defined as:

$$
\begin{aligned}
\mathcal{G}_{FFT} &= \frac{1}{hw}\|\text{FFT}(I_0)\|, \\
\mathcal{G}_{LV} &= \text{Var}(k_L * I_0),
\end{aligned}
\tag{3.3}
$$

where I_0 is a given frame with height h and width w. FFT magnitude measure is an approximation of power spectrum density in the frequency domain. The variance of Laplacian stresses spatial information by counting sharp transitions in the frame. These two measure works in a complementary way. Therefore, we integrate the two methods by taking a weighted average of two measures' scores ranking over a certain window. Let $\text{rank}(I, W, \mathcal{G})$ denotes a function that returns the rank of frame I among all the frames in the window W scored by the quality measure $\mathcal{G}$ in ascending order. The selected highest-quality frame is:

$$
I_{HQ} = \underset{I \in W_i}{\text{argmax}}[\lambda \cdot \text{rank}(I, W_i, \mathcal{G}_{FFT}) + (1-\lambda) \cdot \text{rank}(I, W_i, \mathcal{G}_{LV})], \tag{3.4}
$$

where λ is the relative weight parameter.

In practice, the video sequence is sub-sampled at rate r to further boost efficiency before applying the sliding window filter. As a hyper-parameter, the sub-sample rate r has a great impact on the tradeoff between energy efficiency and performance. Although setting a higher sub-sample rate could save more energy, it has a higher chance of missing scenes for text spotting.

3.2.2.2 Stage II: Rejecting Text-free Images and Cropping Non-Text Regions

Problem Definition. Known for high time complexity and energy consumption, connected component-based text detection depends on maximally stable extremal region (MSER) as character candidates, and stroke width transform (SWT) for filtering and pairing of connected components. Given the observation that cohesive characters compose a word or sentence sharing similar properties such as spatial location, size, and stroke width, we turn to Canny edge detector [**?**] to locate the edge pixels that build the text's structure (a.k.a. contour).

Implementation. In Fig. 3.3, we further reject text-free images and crop non-text regions. First,

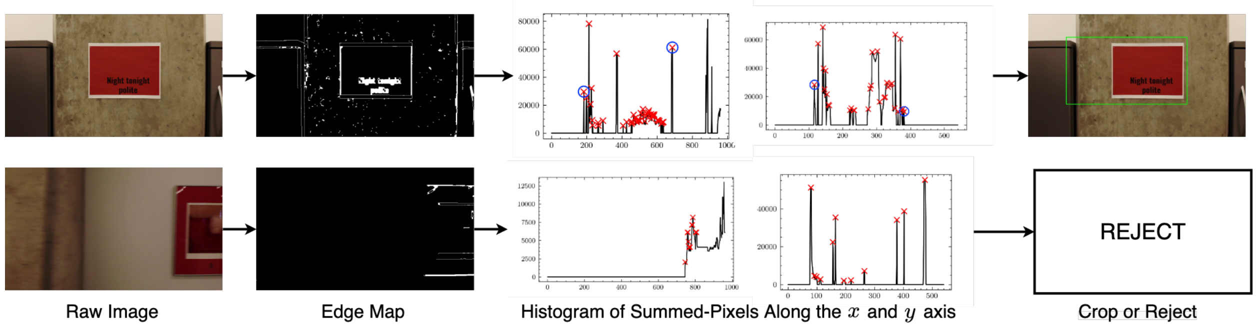


Figure 3.3: Cropping text foreground: we use the histogram to analyze the edge information of the selected frame. If the number of peaks and the mean of intensity satisfies predefined thresholds, text bounding coordinates will be selected from peaks info. Otherwise, the frame will be discarded. Reprinted from [2].

Canny edge detector is applied on the three channels of the input image I_{yuv} represented in YUV color space, and the three channels (Y_c, U_c, V_c) are merged by bitwise OR (denoted as "|") operation to obtain the edge map I_e. Then, morphological closing is applied on the I_e to remove small holes and merge connected components. If any text region is present in the image, a binary image with continued text characters will be returned. Next, the histogram map is obtained by summing up pixels along the x and y axis [1]:

$$H_x[i] = \sum_{k=1}^{h} I_c[i,k], H_y[j] = \sum_{k=1}^{w} I_c[k,j], \tag{3.5}$$

where w and h are the width and height of the image. After that, all the peaks [2] for these two histogram maps, *i.e.*, P_x and P_y, are found. Text regions are assumed to fall within the peaks. Finally, Text-free images are rejected based on two preset thresholds (θ, α) on the peak intensities and numbers, respectively. Note that the second-stage selector cannot deal with images with complicated backgrounds, since the peaks value varies along both axes without any identifiable pattern. Therefore, images with complicated backgrounds whose peaks are consistently high along the entire x and y axis are accepted.

[1]With the origin in the lower-left corner, the x-axis is running from left to right, and the y-axis is running from bottom to up.

[2]Peaks are all local maxima by comparing neighboring values in the histogram.

Cropping text regions improves the SNR [3] in the image. On images with simple background, the text regions are assumed to lie between (x_l, y_b) and (x_r, y_t). The coordinates of the text region are obtained from the peaks via

$$x_l, x_r, y_b, y_t = P_x[1], P_x[-1], P_y[1], P_y[-1]. \tag{3.6}$$

The details of the second-stage selector is shown in Algorithm 5.

Algorithm 5: Rejecting Text-free Images or Cropping Text Regions

```
1  Initialization: $\theta$, $\alpha$: predefined thresholds
2  $I_{yuv} \leftarrow$ RGB2YUV($I$)
3  $Y_c, U_c, V_c \leftarrow$ CannyEdge($I_{yuv}$)
4  $I_e \leftarrow Y_c \mid U_c \mid V_c$
5  $I_c \leftarrow$ MorphClose($I_e$)
6  $H_x, H_y \leftarrow$ Histogram($I_c$) // sum up pixels among axis
7  $P_x, P_y \leftarrow$ FindPeaks($H_x, H_y$)
8  $\mu_x, \mu_y \leftarrow$ Mean($P_x$), Mean($P_y$)
   // Whether the number of peaks or the mean of intensity is
       less than preset thresholds
9  if Count($P_x$) $\leq \theta$ or Count($P_y$) $\leq \theta$ or $\mu_x \leq \alpha$ or $\mu_y \leq \alpha$ then
10     REJECT
11 end
12 else
13     ACCEPT
14     $x_l, x_r, y_b, y_t \leftarrow P_x[1], P_x[-1], P_y[1], P_y[-1]$
15     return $I[x_l : x_r, y_b : y_t]$
16 end
```

3.2.2.3 Stage III: Rejecting Out-of-Distribution Images

Problem Definition. A "trained" E^{2}VTS model f with fixed parameters is able to fit a distribution $\mathcal{X}_f$ defined on the image space. During inference, rejecting the out-of-distribution images in an early-exit way could greatly reduce energy consumption. The out-of-distribution rejection prob-

[3] We treat text related pixels as signal and all other pixels as noise.

lem [188] can be formulated as a binary classification. Examples of the positive cases and negative cases used to train the rejector are shown in Fig. 3.4.

Implementation. Grad-CAM [189], a visual explanations technique via gradient-based localization, is deployed to locate the first text semantic-aware layer l for our model. The outputs of the text semantic-aware layer H_l serve as the high-level features to distinguish the out-of-distribution images from the in-distribution ones. Support Vector Machines (SVM) is used for binary classification on H_l. SVM is preferred over the deep model due to its small size in the number of parameters and low latency.

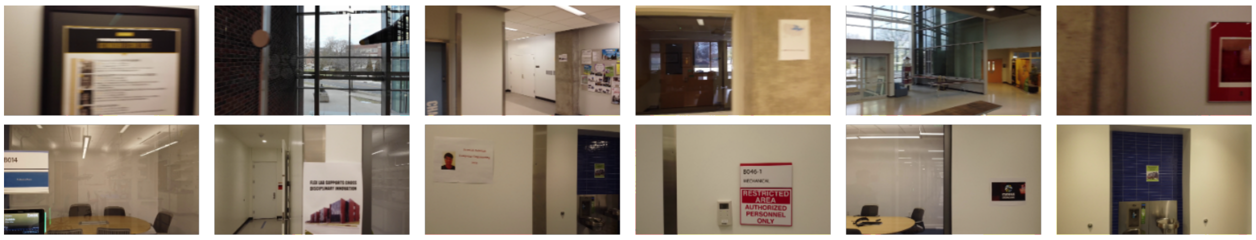


Figure 3.4: The negative samples in the first row and positive samples in the second row are used to train the out-of-distribution rejector. Heavily-blurred, text-free, and truncated-text are all considered as negative cases to be rejected. Reprinted from [2].

3.3 Experiments

3.3.1 Experiment Settings

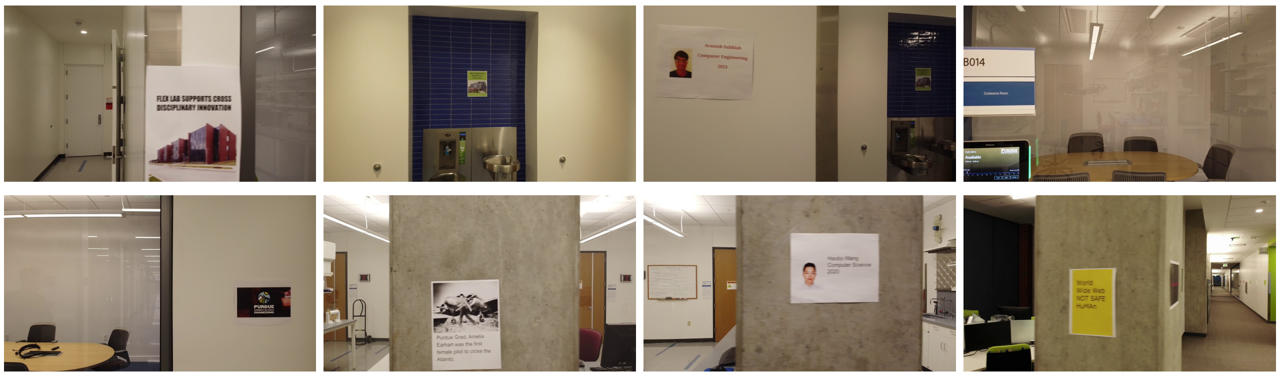


Figure 3.5: Sample Images from the LPCVC-20 Video Text Spotting Dataset. Reprinted from [2].

3.3.1.1 Datasets and Evaluation Protocols

We evaluate the proposed E^{2}VTS approach on the LPCVC-20 video text spotting dataset, abbreviated as **LPCVC-20**. The videos are captured by UAVs flying indoors on the corridors, where tons of posters and board signs with rotated text are presented. Five videos were used for training, and one video was reserved for testing. After converting videos into frames, we handpick text-related images to form a collection that includes $7,886$ for training and $2,033$ for testing. Furthermore, the text was annotated using the Auto Labeling algorithm described in the coming subsection. LPCVC-20 consists of images of resolution 3840×2160, 1920×1080, and 1280×720.

IoU, IoP, IoG [4] are used for detection and edit-distance is used for recognition. The predicted bounding box with the maximum IoU is selected for each ground truth, and the edit distance is calculated between the ground truth text label and the predicted text.

Image Registration-Aided Annotation for Video Text Spotting by Auto Labeling. Given the observation that temporally consecutive frames are describing the same scene, we propose Auto-

[4]Given a ground truth bounding box area G and predicted bounding box area P the IoU is $(P \cap G)/(P \cup G)$, IoP (a.k.a precision) is $(P \cap G)/P$, and IoG (a.k.a recall) is $(P \cap G)/G$.

Labelling in Algorithm 6 to aid the video annotation, which utilizes the videos' temporal redundancy and continuity. It takes advantage of feature matching and perspective transformation to transfer the annotated bounding box from the source frame to the target frame. Figure 3.6 shows the annotation results produced by Algorithm 6.

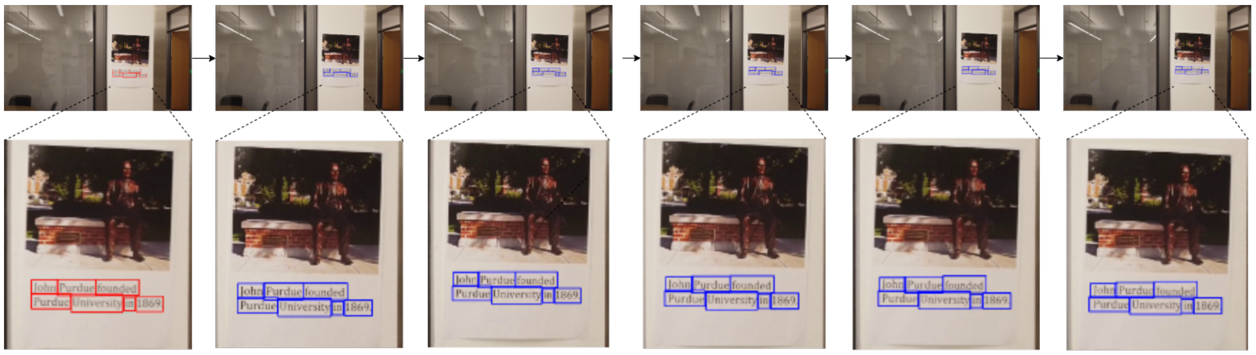


Figure 3.6: Qualitative Results of Auto Labeling: extract features from the source frame and the target frame. Conduct feature matching and perspective transform to update bounding boxes annotation. Repeat the process until the end of the scene. Reprinted from [2].

Energy Consumption Measurement. A USB power meter [5] is used to measure energy consumption. We connect the USB power meter in series to the power supply of the Raspberry Pi. With this setup, the power meter can real-time measure the current through the Raspberry Pi. Since the voltage for Raspberry Pi is constantly 5V, we can calculate the energy consumption by recording the current values. The power meter is connected with a computer through Bluetooth, and the energy measurements of the Raspberry Pi are recorded using [190]. The timestamps for model inference are written down to measure the latency for the model.

3.3.1.2 Model Compression

Pruning. The pruning algorithm compresses the neural network by removing redundant weights or channels of layers. For a Raspberry Pi, structured pruning is preferred over unstructured pruning, since structured pruning does not require specific hardware support for deployment. For our

[5]MakerHawk UM34C USB 3.0 Multimeter Bluetooth USB Voltmeter Ammeter

Algorithm 6: Auto-Labeling

```
   // Annotate the 1st frame
1  $b_s \leftarrow$ Annotate($V[1]$) // Number of frames describing the same scene
2  $N \leftarrow$ Size($V$)
3  for $i \leftarrow 2$ to $N$ do
       // Next Adjacent frame
4      $I_t \leftarrow V[i]$ // Feature Matching
5      $k_1, d_1 \leftarrow$ SIFT($I_s$); $k_2, d_2 \leftarrow$ SIFT($I_t$)
6      $m \leftarrow$ LoweRatioTest(BFMatcher($d_1, d_2$))
7      $p_s, p_t \leftarrow$ FilterKeyPts($m, k_1$), FilterKeyPts($m, k_2$)
       // Perspective Transformation
8      $b_t \leftarrow$ Perspective($b_s$, HomographyMatrix($p_s, p_t$))
9      $I_s \leftarrow I_t$; $b_s \leftarrow b_t$
10 end
```

experiment, we applied ℓ_1 filter Pruner with a one-shot pruning strategy and a sparsity rate of 0.7, which allowed our model to achieve the best trade-off between accuracy and energy efficiency.

Quantization. Quantization refers to techniques for using a reduced precision integer representation for weights and activations. For Raspberry Pi, Pytorch provides QNNPACK backends which supports running quantized operators efficiently on ARMS CPU. For our experiment, we applied static post quantization on all convolutional and fully connected layers; and applied dynamic post quantization on the LSTM modules in the CRNN model.

3.3.2 Ablation Studies

3.3.2.1 Crop & Resize vs. Aligned RoI Pooling

In this section, we conduct ablation studies for the two-step Crop and Resize E^{2}VTS text spotting model and the two-stage Aligned RoIPool text spotting model. From Table 3.1 it can be seen that the E^{2}VTS model performs better than the Aligned RoIPool model at all resolutions. The different factors that influence the performance of the model are also measured. From Tables 3.1 it can be concluded that the greater bounding box to character count ratio and the lesser character count improves the recognition performance. Table 3.2 shows the deployment results of E^{2}VTS on Raspberry Pi.

Table 3.1: E^2VTS and FOTS results on LPCVC-20. Reprinted from [2].

		E^2VTS			FOTS		
EditDistance \ Resolution		1200	600	300	1200	600	300
BBox Area/Char Count	<=20	-	6.86	3.69	-	6.84	4.19
	<=60	5.25	2.12	3.60	4.56	2.84	5.19
	>60	1.93	2.00	2.92	2.62	3.48	5.97
Char Count	<=4	1.08	1.31	2.21	1.83	2.38	3.28
	<=8	1.86	2.08	3.56	2.98	4.21	6.12
	>8	4.20	3.79	5.32	3.95	4.92	8.61
Total		1.93	2.04	3.26	2.65	3.48	5.25

Table 3.2: Performance, Latency, and Energy Measurement of E^2VTS on Raspberry Pi. Reprinted from [2].

Model	IoU	IoP	IoG	EditDistance	Latency	Avg Energy
E^2VTS	72.21	76.24	93.94	1.39	12.90	31.77

3.3.2.2 Multi-Stage Image Processor

We compare the overall performance of our method after incorporating different data level efficiency techniques. As shown in Table. 3.3, incorporating data level efficiency results in a better performance in both accuracy and efficiency.

Table 3.3: Ablation studies on the multi-stage image processor. Performance, latency, and energy consumption are evaluated. Reprinted from [2].

Stage I	Stage II	Stage III	Latency	Energy	EditDistance
✓			627.43	1841.49	**0.78**
	✓		545.20	1349.23	1.14
		✓	571.48	1482.31	1.05
✓	✓	✓	**528.12**	**1267.2**	0.96

Based on the results in Table 3.3, Stage I data pre-processing improves the accuracy of the model by selecting the best quality frame within a window size as the model's input. The latency and energy decrease slightly due to the extra cost introduced by quality scoring and the decrease of the video's sub-sample rate. Stage II data pre-processing decreases the latency and average energy consumption of the model by improving the SNR in the image and rejecting low-quality and non-text frames. Stage III data pre-processing also decreases latency and energy consumption by rejecting out-of-distribution frames at an early stage of the detection model. The integration of Stage I, II, and III as pre-processing benefits the model from the perspective of speed and energy consumption.

3.3.2.3 Deployment on Raspberry Pi

In this section, we evaluate the overall performance of our method after incorporating different model-level efficiency techniques, which include pruning and quantization.

Table 3.4: Ablation studies on pruning ($\mathbf{P}$) and quantization ($\mathbf{Q}$). Reprinted from [2].

P	Q	Latency	Energy	EditDistance
	✓	76.48	195.25	**1.09**
✓		56.67	164.73	1.12
✓	✓	**12.90**	**39.23**	1.14

Based on the results in Table 3.4, model pruning and quantization significantly decrease latency and average energy consumption, respectively. Although implementing model compression results in a sightly drop in accuracy, the tradeoff between energy efficiency and accuracy shows that incorporating model level efficiency notably boosts overall performance.

4. PIPELINE SIMPLICITY

4.1 Motivation

A common paradigm of video action detection is inspired by the two-stage paradigm of Faster R-CNN in object detection, $i.e.$, firstly generating person proposals and then classifying their actions. Specifically, these prior works usually combine a "backbone" 3D CNN (*e.g.*, C3D, I3D, SlowFast) with a region-based person detector. However, the growing adoption of many specialized components has made video action detection models gigantic and conceptually highly sophisticated.

Initially, a video is split into short clips of 2-5 seconds, independently forwarded through the 3D CNN to compute a feature map. On the center frames of the raw short clips, action proposals are then generated by a region proposal algorithm or densely sampled anchors. The region proposals together with the feature map next go through the region of interest (RoI) pooling to compute RoI features for each candidate anchor, and subsequently, for action classification and localization refinement.

Later on, non-local (NL) blocks are incorporated to better capture the long-range dependencies in both space and time to classify more complicated actions, *e.g.*, person-person interactions and person-object interactions.

Recently, memory bank-based methods, *e.g.*, long-term Feature Bank (LFB) was proposed to encode and store the rich, time-indexed information over the entire span of a video, providing a supportive context that allows a video model to understand the present better.

4.1.1 Why Transformers for Video Action Detection?

There is an emerging trend of using Transformer architectures for computer vision tasks [147–149, 152]. Transformer is designed to learn long-range relations on sequential data. In comparison, Transformer has no inductive biases that are inherent to CNN, such as locality and spatial invariance. Therefore, when understanding the visual data, CNN is complementary to Transformer so that CNN can model the low-level structure of images/videos, and Transformer can capture the high-level semantics.

Without denying all those specialized models' success, we pose a reflection: can we alternatively pursue a general, conceptually simple, and sufficiently versatile architecture and achieve even better performance for video action detection using the Transfomer architecture?

VATN [148] made the first attempt at introducing Transformers to video action detection. It used a modified Transformer-style architecture to classify the action of a person of interest, yielding a decent performance on the Atomic Visual Actions (AVA) dataset [175]. However, the Transformer part in VATN is only "half-baked", since it leverages merely the Transformer encoder for action classification, but not using its decoder, while instead still using region proposal network (RPN) as the sampling mechanism for person localization. Starting from the milestone of VATN, the next question is: *can we take further steps, utilizing less or even nearly no specialized components, while obtaining more competitive performance?*

4.2 Proposed Approach: TxVAD

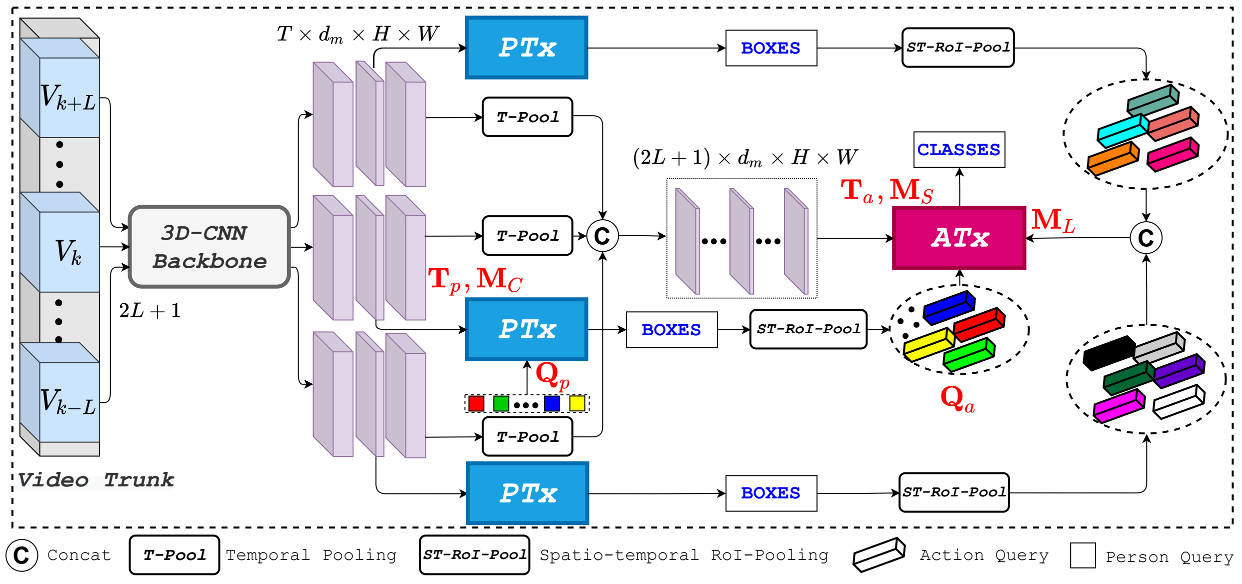


Figure 4.1: TxVAD uses a 3D-CNN backbone to learn a 5D representation of an input video trunk, which is composed of multiple consecutive clips. We use two Transformer architectures: person Transformers (PTx) for localization and an action Transformer (ATx) for classification. PTx works on separate video clips and produces bounding boxes on persons at the central frame for each clip. ATx works on the temporally concatenated feature maps and predicts the final action classes for each box. Note that all person Transformers (PTx) share parameters. According to the bounding box obtained from $\mathbf{Q}_p$, Spatio-temporal RoI-Pooling (ST-RoI-Pool) is used to extract the feature map of the persons of interest to produce the action queries $\mathbf{Q}_a$. Temporal-Pooling (T-Pool) is used to average the temporal channels in the 4D feature map for each clip. Please refer to section 3.2.1 and 3.2.2 for details of $\mathbf{M}_C, \mathbf{M}_S, \mathbf{M}_L, \mathbf{T}_p, \mathbf{T}_a, \mathbf{Q}_a, \mathbf{Q}_p$. $\mathbf{M}_C$ is the transformed tokens by the encoder on $\mathbf{T}_p$. $\mathbf{M}_S$ is the transformed tokens by the encoder on $\mathbf{T}_a$.

Using 3D CNNs to model low-level structures and Transformers to capture high-level semantics, TxVAD is designed to localize all persons and classify all their actions. It ingests a video trunk composed of consecutive clips centered on their corresponding annotated “keyframes”, and generates a set of human bounding boxes for all the people on the temporal center frame, where each box is labeled with the predicted action category.

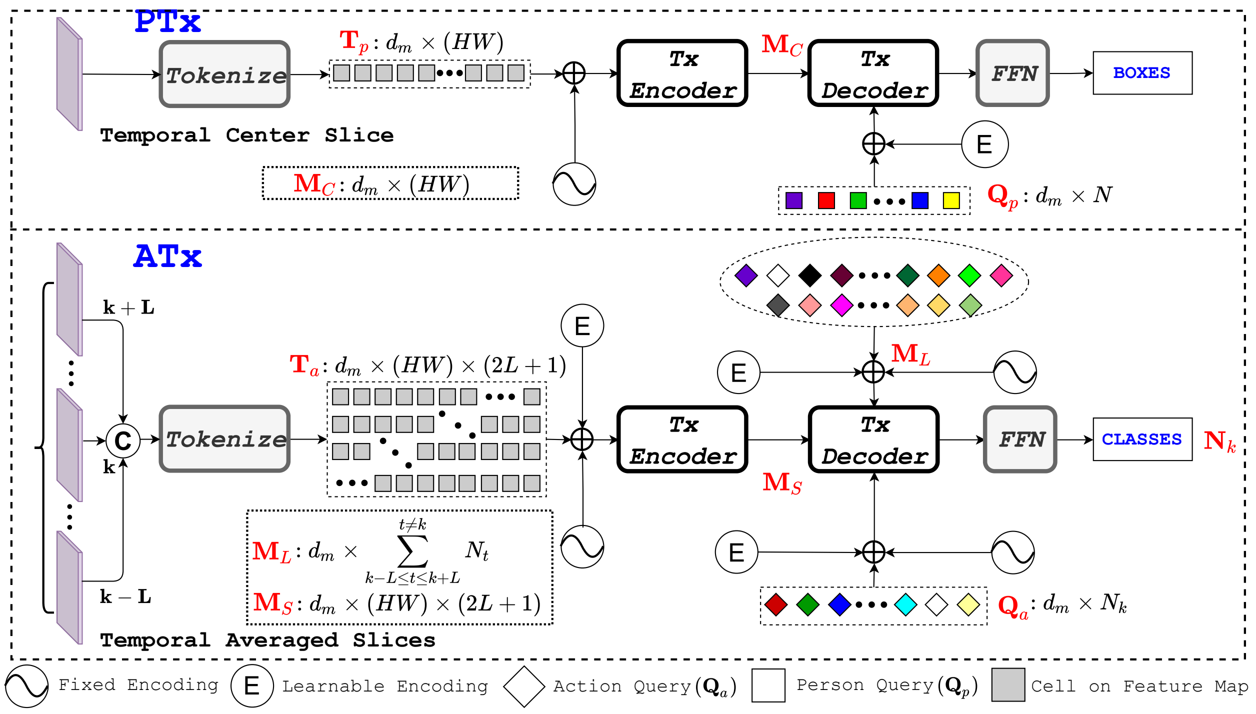

PTx
Tokenize
$\mathbf{T}_p: d_m \times (HW)$
Tx Encoder
$\mathbf{M}_C$
Tx Decoder
FFN
BOXES
Temporal Center Slice
$\mathbf{M}_C: d_m \times (HW)$
E
$\mathbf{Q}_p: d_m \times N$
ATx
$\mathbf{k+L}$
$\mathbf{k}$
$\mathbf{k-L}$
C
Tokenize
$\mathbf{T}_a: d_m \times (HW) \times (2L+1)$
E
Tx Encoder
Tx Decoder
$\mathbf{M}_L$
$\mathbf{M}_S$
FFN
CLASSES
$\mathbf{N}_k$
$\mathbf{M}_L: d_m \times \sum_{k-L \le t \le k+L}^{t \neq k} N_t$
$\mathbf{M}_S: d_m \times (HW) \times (2L+1)$
$\mathbf{Q}_a: d_m \times N_k$
Temporal Averaged Slices
Fixed Encoding
Learnable Encoding
Action Query($\mathbf{Q}_a$)
Person Query($\mathbf{Q}_p$)
Cell on Feature Map

Figure 4.2: A detailed look into the Person Transformer (PTx) for localization and the Action Transformer(ATx) for classification in TxVAD. In PTx, Tx-Encoder encodes the spatial information at a temporal center slice from the video clip and Tx-Decoder decodes person queries into box coordinates with $\mathbf{M}_C$. In ATx, Tx-Encoder encodes the spatio-temporal information from the whole video trunk and Tx-Decoder decodes action queries into action classes with $\mathbf{M}_s$ and $\mathbf{M}_L$.

4.2.1 TxVAD Architecture

As shown in Figure 4.1, our TxVAD has three components: a 3D-CNN backbone to extract a video trunk-level feature representation, a Transformer for person localization by modeling spatial relations, and another Transformer for action classification by modeling spatio-temporal relations.

4.2.2 Detecting Actions by Two Transformers

3D-CNN Backbone. Given a video trunk $v \in \mathbb{R}^{(2L+1)\times T_0 \times C_0 \times H_0 \times W_0}$ containing $(2L+1)$ consecutive video clips, a 3D-CNN backbone [17, 18] produces a feature map $f(v) \in \mathbb{R}^{(2L+1)\times T \times C \times H \times W}$ where T, C, H, W stands for temporal, channel, height, and width dimensions, respectively.

A 3D convolution with $1 \times 1 \times 1$ kernel is used to reduce the number of channels of $f(v)$ from C to a smaller number d_m, *i.e.*, Transformer's model dimension. The trunk-level feature map is represented as $w \in \mathbb{R}^{(2L+1)\times T \times d_m \times H \times W}$.

Revisiting Transformer. The core of a Transformer is the attention module, which aggregate the values in a weighted sum according to the attention weights measured by the compatibility of all the query-key pairs for each query. Given the set of N_q queries packed into a matrix X_q and the set of N_{kv} keys/values packed into a matrix X_{kv}, each single-head attention feature is obtained by

$$\text{Attn}(X_q, X_{kv}) = \text{softmax}(\frac{X_q' X_k'^{\intercal})}{\sqrt{d_m'}})(X_v'), \tag{4.1}$$

where $X_q' = X_q W_q$, $X_k' = X_{kv} W_k$, and $X_v' = X_{kv} W_v$. W_q, W_k and W_v are embedding matrices of size $d_m \times d_m'$ with learnable weights.

4.2.2.1 Person Transformer (PTx) for Localization.

Overall, PTx produces person bounding boxes on the temporal center frame for each clip in the video trunk. Since we segment the untrimmed video into clips by centering on the annotated "keyframes", the temporal center slice in the 4D tensor naturally corresponds to the feature map of the annotated "keyframes". As shown in Figures 4.1 and 4.2, for each consecutive clip, PTx takes the temporally sliced central frame of their corresponding 4D feature map as input and produces bounding boxes on detected persons.

Tokenization. For the i-th clip, the spatial H, W dimensions of frame-level feature $w(i, \lfloor T/2 \rfloor, ::)$ are collapsed into one dimension, resulting in a sequence of frame-level visual tokens $\mathbf{T}_p \in \mathbb{R}^{d_m \times HW}$. Since Transformer has neither recurrence nor convolution, fixed Sinusoidal "positional encoding" [191] is added to the visual tokens $\mathbf{T}_p$. In the cross-attention of decoder, $\mathbf{T}_p$ updated by the encoder serves as context memory, denoted as $\mathbf{M}_C$ in Figure 4.2. A fixed number of person queries $\mathbf{Q}_p$ are initialized as all $\mathbf{0}$ vectors and supplemented with learnable "positional encoding" as anchors.

Transformer. Following [149], we adopt the standard Transformer [191] encoder-decoder architecture to transform the frame-level visual tokens $\mathbf{T}_p$ to a set of person queries $\mathbf{Q}_p$. Using self-attention on the frame-level visual tokens $\mathbf{T}_p$, the encoder reasons in the spatial scene to roughly disentangles different persons. The decoder uses self-attention on the person queries $\mathbf{Q}_p$ to capture their relations and cross-attention on $\mathbf{M}_C$ to attend to person extremities such as heads or legs. Finally, an FFN is used to decode the updated person queries $\mathbf{Q}_p$ into bounding box coordinates.

4.2.2.2 Action Transformer (ATx) for Classification.

As shown in Figures 4.1 and 4.2, ATx also adopts the Transformer encoder-decoder architecture, with "positional encoding" and parallel decoding. Given temporally discrete bounding boxes predicted by PTx on multiple "keyframes" from each consecutive clip, we extend them by repeating the boxes in time to form action tubes that surround the persons of interest in each clip. Unlike the person queries $\mathbf{Q}_p$ initialized as all zeros to decode person bounding box coordinates in PTx, here action queries $\mathbf{Q}_a$ are initialized as feature maps of the action tubes on the center clip of the video trunk. ST-RoI-Pooling is adopted to reduce the size of $\mathbf{Q}_a$ to d_m, which is 1024 if SlowFast backbone is used.

Tokenization. The H, W dimensions of the $5D$ feature map from the trunk are collapsed into one dimension and the T dimension is averaged, giving a sequence of trunk-level visual tokens $\mathbf{T}_a \in \mathbb{R}^{d_m \times (HW \times (2L+1))}$. To make up for the missing spatio-temporal information, fixed spatial "positional encoding" for the HW dimension and learnable temporal "positional encoding" for the $(2L + 1)$ dimension are added to the visual tokens $\mathbf{T}_a$. In the first cross-attention of decoder, $\mathbf{T}_a$ updated by the encoder serves as *scene context memory*, denoted as $\mathbf{M}_S$ in Figure 4.1 and 4.2. In the second cross-attention of decoder, the feature maps of the action tubes not on the center clip of the video trunk serve as *long-term context memory*, denoted as $\mathbf{M}_L$ in Figure 4.1 and 4.2.

Transformer. Likewise, we use the standard Transformer to transform the video-level visual tokens $\mathbf{T}_a$ to a set of action queries $\mathbf{Q}_a$ that are supplemented with spatio-temporal "positional encoding" (the coordinates of the bounding boxes) and "group encoding" (person's identity tags). Using self-attention on the visual Tokens $\mathbf{T}_a$, the encoder reasons in the spatio-temporal scene to roughly associate interaction information from different persons and objects. The decoder uses self-attention on the action queries $\mathbf{Q}_a$ to capture their relations (*e.g.*, person-person interaction) and two cross-attentions on the *scene context memory* $\mathbf{M}_S$ and *long-term context memory* $\mathbf{M}_L$ respectively, to aggregate context information from other persons and objects (*e.g.*, person pose action, person-object and person-person interactions). Finally, an FFN is used to decode the updated action queries $\mathbf{Q}_a$ into action classes.

4.2.3 Training Details

Training such Transformer-based architectures from end to end is time-costly and often unstable [149], especially now on video data. After presenting the loss functions, a hardness-aware curriculum training strategy is proposed to *stabilize* the TxVAD training.

Loss Functions. We use a multi-task loss to jointly train TxVAD, with two loss terms for PTx and ATx respectively (λ is the weight to control their relative importance):

$$\mathcal{L}_{\text{TxVAD}} = \mathcal{L}_{\text{PTx}} + \lambda \mathcal{L}_{\text{ATx}}. \tag{4.2}$$

We next detail the design of each loss term.

Following [149], we use a set prediction loss to train the PTx. Binary classification loss is adopted since there is only two classes of foreground (person) and background (non-person). Given the set of predicted person boxes $\hat{y}_p = \{(\hat{\boldsymbol{b}}_i, \hat{p}_i)\}_{i=1}^{N}$ and the ground truth set of action boxes $y_{gt} = \{(\boldsymbol{b}_i, \boldsymbol{c}_i)\}_{i=1}^{N'}$, the bipartite matching between y_{gt} and $\hat{y}_p$ is obtained by finding a permutation $\hat{\sigma}$ as assignment via Hungarian algorithm that minimizes the matching cost [149]. Using the optimal assignment $\hat{\sigma}$, the person localization loss is defined as:

$$\mathcal{L}_{\text{PTx}}(y_{gt}, \hat{y}_p) = \sum_{i=1}^{N'} \Big[-\log \hat{p}_{\hat{\sigma}(i)} + \mathcal{L}_{\text{box}}(\boldsymbol{b}_i, \hat{\boldsymbol{b}}_{\hat{\sigma}(i)}) \Big], \tag{4.3}$$

where $\hat{p}_{\hat{\sigma}(i)}$ is the predicted probability of person, $\hat{b}_{\hat{\sigma}(i)}$ is the predicted bounding box, and $\mathcal{L}_{\text{box}}$ is a linear combination of the ℓ_1 loss and the generalized IoU loss in [149].

Given the set of predicted action boxes $\hat{y}_a = \{(\hat{\boldsymbol{b}}_i, \hat{\boldsymbol{p}}_i^c)\}_{i=1}^{N}$ and the ground truth set of action boxes $y_{gt} = \{(\boldsymbol{b}_i, \boldsymbol{c}_i)\}_{i=1}^{N'}$, we use a multi-label classification loss in the form of binary cross-entropy for training ATx:

$$\mathcal{L}_{\text{ATx}}(y_{gt}, \hat{y}_a) = \sum_{i=1}^{N'} -\log \boldsymbol{c}_i \hat{\boldsymbol{p}}_i^c. \tag{4.4}$$

Hardness-Aware Curriculum Learning. Similar to as found in DETR [149], our proposed TxVAD also suffers from a long and unstable training schedule. We locate one crucial cause: at initialization, the attention modules cast nearly uniform attention weights to all the cells in the feature maps.

Our solution refers to curriculum training [192], which first focuses on learning from a subset of simple training examples that requires localized attention for classification, and gradually includes the remaining more complex samples that require longer-range spatio-temporal modeling by attention. Such a curriculum learning strategy gives a localize attention as initialization for the Transformer and gradually increases the "receptive field" of Transformer as harder samples are included in the training.

The core question here is how to design the "curriculum". Human actions are naturally grouped in three categories: person pose, person-person interactions, and person-object interactions[1]. Those three categories intuitively have different "semantic difficulty levels", *e.g.*,

- Single person poses are most "standalone", have the least interventions with surroundings, and do not rely much on temporal information to recognize correctly;
- Classifying person-person interactions needs to capture multiple person subjects' appearance features, and may suffer from body occlusions;
- Person-object interactions can be more challenging, since the objects are smaller and actions' spatial ranges often become smaller too, compared to person-person.

Our hardness-aware curriculum training strategy hence has three stages. In the first stage, we only train with the subset of videos with class labels belong to the person pose category, and detect/classify those person poses only. The second stage extends to training with classes from both person pose and person-person interactions categories. Finally, the third stage trains with all three categories. We find the strategy to improve TxVAD's training substantially.

[1]As actions of different types can coexist in videos, *e.g.*, talk to (person-person) and stand (person pose), the three categories are NOT disjoint.

4.3 Experiments

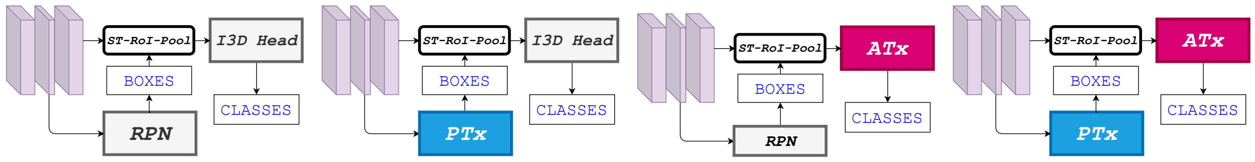
4.3.1 Datasets and Implementation Details

Datasets. We train and test our model on the Atomic Visual Actions (AVA) version 2.1 benchmark [175] dataset, which contains 211K training clips and 57K validation clips segmented from 235 training movie videos and 64 validation movie videos, respectively. AVA is annotated on "keyframes" sparsely sampled at 1 frame per second (FPS). Each person at the keyframe is labeled with one bounding box and labels from 80 atomic visual actions, broadly covering person-person interactions, person-object interactions, and person pose actions. As the most challenging and comprehensive benchmark, AVA is used as the main benchmark to conduct ablation studies and demonstrate the effectiveness of our approach.

Besides AVA, we also report the performance of TxVAD on JHMDB-21 [8] and UCF101-24 [11]. JHMDB-21 contains 928 temporally trimmed short video clips with 21 action classes. Every frame in JHMDB-21 is annotated with one actor bounding box and a single action label. As a subset of UCF101 with 24 action classes, UCF101-24 has $3,207$ videos provided with framewise spatio-temporal annotations for action detection.

The proposed curriculum learning approach can be applied to other video action detection datasets such as JHMDB-21, since the proposed different difficulty levels of action categories (single person pose, person-person interactions, and person-object interactions) are general across datasets. Note that JHMDB-21 has excluded action classes belonging to person-person interaction. Therefore, to apply curriculum training, we have two stages, first on single-person pose actions then on person-object interactions.

Evaluation Metrics. Frame-level mean average precision (frame-mAP) at an IoU threshold of 0.5 is reported. To compare with previous works [22, 26–28, 148] on AVA v2.1, our proposed TxVAD is evaluated on the subset of 60 classes that have at least 25 validation examples. On AVA, both the action agnostic person localization and action classification are evaluated on the annotated "keyframes" using mAP at IoU threshold of 0.5. On JHMDB-21, frame-mAP averaged over three



(a) RPN + I3D-head ($n = 1$) (b) PTx + I3D-Head ($n = 1$) (c) RPN + ATx ($n = 1$) (d) PTx + ATx ($n = 1$)

Figure 4.3: Ablation studies on action detection with different architectures. $f(V_k)$ is generated by I3D-base which includes all the layers up to "Mixed_4f" in the I3D. I3D-head is the "Mixed_5b" and "Mixed_5c" layers in the I3D network. ST-RoI-Pool is an abbreviation of Spatio-Temporal RoI-Pool, which extracts feature maps guided by action tubes, which is formed by repeating the box in time. Tx stands for Transformer. For simplicity, only one clip is contained in the video trunk, *i.e.*, $n = 1$.

training/validation splits is reported. On UCF101-24, frame-mAP is reported on the first split.

Implementation Details. We experiment with two families of backbones: I3D [17] and SlowFast [18]. I3D is inflated from 2D Inception architecture. SlowFast is modified from 3D ResNet architecture and involves two pathways fused by lateral connection. The slow pathway captures spatial semantics, and the fast pathway captures motion. The 3D ResNet can be augmented with non-local (NL) blocks[2]. Both I3D and SlowFast are pretrained on the video classification task on Kinetics [6, 193, 194].

If I3D is used as the backbone, all frames are resized to $H_0 \times W_0 = 400 \times 400$ and the video clip length is set to $T_0 = 64$. The 3D backbone, and the output of "Mixed_4f" is used as $f(v)$. $f(v)$ is of size $n \times T \times C \times H \times W$, where n is the number of clips in a video trunk, $T = T_0/4$ is the temporal length of each video clip, $C = 832$ is number of channels, $H, W = H_0/16, W_0/16$ are the spatial dimensions. If SlowFast is used, the short side is resized to be 256 and $C = 2048 + 256 = 2304$. $\tau = 16$, $\alpha = 8$, and $\beta = 1/8$.

Training Strategies. The Adam optimizer with decoupled weight decay (AdamW) [195] is adopted with a mini-batch size of 32 on 8 GPUs. Batch normalization layers in 3D-CNNs are frozen. There are two pathways in TxVAD: Person Transformer for localization (PTx) and Action Transformer for classification (ATx). Firstly, PTx is self-supervised pretrained using the pretext task of "ran-

[2]In experiments, R50-NL and R101-NL are the backbones using SlowFast based on 3D ResNet-50 and ResNet-101 with NL blocks, respectively.

dom query patch detection" [196] and finetuned on AVA using action-agnostic bounding boxes adopting $\mathcal{L}_{\text{PTx}}$ in Eq. 4.3. Secondly, with PTx's parameters being fixed, ATx is trained using action queries Q_a obtained from RoI-Pooled features extract from ground-truth action boxes using $\mathcal{L}_{\text{ATx}}$ in Eq. 4.4. Lastly, PTx and ATx are connected and trained in an end-to-end way using the multi-task loss in Eq. 4.2.

4.3.2 Ablation Study

We conduct comprehensive ablation studies on the AVA dataset to investigate the effects of different components in our proposed TxVAD, to show the superiority.

Table 4.1: Ablation studies on action detection with different architectures. The structures of type (a), (b) (c), and (d) are shown in Figure 4.3. We set $n = 1$ in this study.

Type	Loc Arc	Cls Arc	Loc (mAP)	Cls (mAP)
a	RPN	I3D-head	92.8	20.8
b	RPN	Tx	92.4	24.2
c	Tx	I3D-Head	95.2	22.4
d	Tx	Tx	**95.4**	**26.1**

Different Architectures. As illustrated in Figure 4.3, to show the advantages of TxVAD with two Transformers (Tx), we come up with 4 different architectures with different localization architectures (Loc Arc) and classification architectures (Cls Arc). Type (a) is based on RPN [52] and I3D-head, where RPN is used for action agnostic person localization and I3D-head is used for action classification. Type (d) is our proposed TxVAD with $n = 1$. As shown in Table 4.1, Transformer outperforms I3D-head in action classification since Transformer's attention mechanism can capture the scene and long-range temporal context information simultaneously, especially in AVA, where actors interact with other actors or objects.

Temporal Length of Video Chunk. The input video trunk has n video clips. The larger n means that the longer-range temporal information is included. We have tried $n = 1, 3, 5$, and the performance is shown in Table 4.2, where $n = 3$ gives us the best result. We investigate those action types whose classification require long temporal information as context, including “put down”, “lift (a person)”, “enter”, “close (*e.g.*, a door)”, “give/serve (an object) to (a person)”, “open (*e.g.*, a window)”, “fight/hit (a person)”, and so on. There is a clear trend that the numbers reported on these actions get higher when longer-range temporal information is included.

Table 4.2: Ablation studies on the temporal length of the video trunk.

	$n = 1$	$n = 3$	$n = 5$
Loc (mAP)	95.4	95.6	95.6
Cls (mAP)	26.1	**26.8**	26.2

Number of Encoders and Decoders. We conduct ablation studies on the number of encoders and decoders and show the results in Table 4.3. Using $M = 5$ and $M = 3$ encoders and decoders gives us the best performance for person localization and action classification, respectively.

Table 4.3: Ablation studies on the- number of encoders and decoders used in TxVAD. To save the hyperparameter search space, we set the number of encoders, M, to be the same as the number of decoders. We set $n = 1$ in this study.

	$M = 1$	$M = 2$	$M = 3$	$M = 4$	$M = 5$
Loc (mAP)	94.6	94.9	95.2	95.4	**95.5**
Cls (mAP)	24.9	25.5	**26.1**	26.1	25.8

Hardness-aware Curriculum Learning. TxVAD’s long and unstable training schedule motivates us to try the proposed hardness-aware curriculum learning. Our proposed hardness-aware curriculum learning has 3 stages. The first stage only focuses on detecting the person pose actions. The second stage focuses on person pose and person-object interaction actions. The last stage sees all three types of actions together. We proceed to the next stage when the performance on the validation set got saturated.

The performance of different action categories at each stage is shown in Table 4.4. We can observe that incrementally adding harder action classes requiring longer-range temporal and more complicated scene context information gives us better performance on each action category. Using the proposed curriculum training improves the mAP from 26.1 to 26.7 on the validation set of AVA v2.1.

Table 4.4: Hardness-aware curriculum learning. The accuracy (mAP IoU@0.5) of different action categories at different stages in the curriculum learning is reported.

	Person-Pose	Person-Object	Person-Person	Overall
Stage 1	44.5	-	-	-
Stage 2	46.8	29.5	-	-
Stage 3	49.4	31.4	22.1	26.7

4.3.3 Competing Results on AVA, JHMDB-21, and UCF101-24 by Adding Things Together

We compare our proposed TxVAD with the state-of-the-art methods on AVA v2.1 by adding hardness-aware curriculum training, adopting a better backbone, and using data augmentation. We only consider methods that use a single model and single crop for evaluation for a fair comparison with our proposed TxVAD. As shown in Table 4.5, compared with the approaches based on the I3D backbone [22, 23, 148, 197] and SlowFast [18, 26, 27], our proposed TxVAD is able to achieve the state-of-the-art performance on AVA in Table 4.5.

TxVAD is further evaluated on JHMDB-21 and UCF101-24. As shown in Table 4.6 and Table 4.7, TxVAD also achieves the state-of-the-art performance on both datasets.

Even without using any hand-crafted module (*e.g.*, anchors, or RPN module), long-term memory banks, or pre-trained off-the-shelf person/object detectors as used in existing methods [23, 25–27, 148], our method has shown the state-of-the-art performance on AVA, JHMDB-21, and

UCF101-24, which suggests that the proposed non-specific, general versatile architecture based on Transformers can be applied to build strong video action detector.

Table 4.5: Comparison with the state-of-the-art methods on the AVA v2.1 dataset. Our proposed TxVAD achieves the best performance on both I3D and SlowFast backbones. Approaches marked with (*) and (+) are incorporated with the hardness-aware curriculum training and the long-term feature bank (LFB), respectively.

Method	Optical Flow	Pretrain	Backbone	mAP
Relation Graph [198]		Kinetics-400	R50-NL	22.2
LFB [25]		Kinetics-400	R101-NL	27.1
Context-Aware RCNN [26]		Kinetics-400	R50-NL	28.0
SlowFast [18]		Kinetics-600	R101-NL	28.2
AIA [27]		Kinetics-700	R101-NL	31.2
AVA baseline [175]	✓	Kinetics-400	I3D	15.6
ACRN [197]	✓	Kinetics-400	I3D	17.4
STEP [22]		Kinetics-400	I3D	20.2
STAGE [23]		Kinetics-400	I3D	23.0
VAT [148]		Kinetics-400	I3D	25.0
TxVAD		Kinetics-400	I3D	**26.8**
TxVAD		Kinetics-700	R101-NL	**31.5**
TxVAD*		Kinetics-700	R101-NL	**31.9**
TxVAD$^+$		Kinetics-700	R101-NL	**31.8**

Table 4.6: Comparison with the state-of-the-art methods on the JHMDB-21 dataset. Approaches marked with (*) are incorporated with hardness-aware curriculum training.

Method	Optical Flow	Pretrain	Backbone	mAP
Two-stream RCNN [199]	✓	ImageNet	VGG	58.5
T-CNN [200]	✓	Sports-1M	C3D	61.3
ACT [201]	✓	ImageNet	VGG	65.7
Context-Aware RCNN [26]		Kinetics-400	R50-NL	79.2
AVA baseline [175]	✓	Kinetics-400	I3D	73.3
ACRN [197]	✓	Kinetics-400	I3D	77.9
TxVAD		Kinetics-400	I3D	**78.4**
TxVAD		Kinetics-400	R50-NL	**79.4**
TxVAD*		Kinetics-400	R50-NL	**79.8**

Table 4.7: Comparison with the state-of-the-art methods on the UCF101-24 dataset. Approaches marked with (*) are incorporated with hardness-aware curriculum training.

Method	Optical Flow	Pretrain	Backbone	mAP
Two-stream RCNN [199]	✓	ImageNet	VGG	65.7
T-CNN [200]		Sports-1M	C3D	67.3
ACT [201]	✓	ImageNet	VGG	69.5
STEP [22]		Kinetics-400	I3D	75.0
AVA baseline [175]	✓	Kinetics-400	I3D	76.3
Relation Graph [198]		Kinetics-400	I3D	77.9
AIA [27]		Kinetics-400	R50-NL	78.8
TxVAD		Kinetics-400	I3D	**78.2**
TxVAD		Kinetics-400	R50-NL	**79.1**
TxVAD*		Kinetics-400	R50-NL	**79.5**

5. CONCLUSION

In this dissertation, we improve the existing performance-driven approaches for video action detection or video text spotting in three aspects: data sharing privacy, model design simplicity, and hardware deployment efficiency.

First, to approach privacy-preserving video data sharing, we propose an innovative framework to address the newly-established problem of privacy-preserving action recognition. To tackle the challenging adversarial learning process, we investigate three different optimization schemes. To further tackle the $\forall$ challenge of universal privacy protection, we propose the privacy budget model restarting and ensemble strategies. Both are shown to improve the privacy-utility trade-off further. Various simulations verified the effectiveness of the proposed framework. More importantly, we collect the first dataset for privacy-preserving video action recognition, an effort that we hope could engage a broader community into this research field. We note that there is much room to improve the proposed framework before it can be used in practice. For example, the definition of privacy leakage risk is core to the framework. Considering the $\forall$ challenge, current L_B defined with any specific f_B is insufficient; the privacy budget model ensemble could only be viewed as a rough discretized approximation of $\mathcal{P}$. More elaborated ways to approach this $\forall$ challenge may lead to a further breakthrough in achieving the optimization goal.

Second, to save energy in video text spotting, we present an energy-efficient video text spotting solution, dubbed as E^{2}VTS, for Unmanned Aerial Vehicles. E^{2}VTS is an energy-efficiency driven model without compromising text spotting performance. The proposed system utilizes data level efficiency enhancement techniques and uses model level efficiency-boosting methods such as pruning and quantization. Specifically, a sliding window is used to select scene-wise highest quality frame; a Canny edge-based algorithm is proposed to reject text-free images and non-text frames; a dynamic routing mechanism emphasizes the in-distribution inputs. Far from the application on UAV devices, our video text spotting system is competent for any energy-constrained scenario.

Finally, to design a simple model for video action detection, we come up with TxVAD, a new paradigm based on Transformer architecture for video action detection. TxVAD has achieved state-of-the-art results on AVA, JHMDB-21, and UCF101-24 datasets. We show that the traditional reliance on many specialized modules and tricks may not be necessary for video action detection, and Transformer-based architecture can build a strong approach for video understanding tasks. Such a Transformer-based architecture brings versatility to TxVAD. For turning from specialized to general-purpose architectures, one strong motivation is to simplify and unify the different task pipelines, so one general suite of models, and all their associated techniques, could be extensively reused by many applications without re-inventing wheels everywhere. Transformers have already demonstrated superb powers in NLP, speech processing, computer vision, biological sequence analysis [202], and even chess gaming [203], making them one ideal model candidate for that convergence.

REFERENCES

[1] Z. Wu, H. Wang, Z. Wang, H. Jin, and Z. Wang, "Privacy-preserving deep action recognition: An adversarial learning framework and a new dataset," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020.

[2] Z. Hu, P. Pi, Z. Wu, Y. Xue, J. Shen, J. Tan, X. Lian, Z. Wang, and J. Liu, "E2vts: Energy-efficient video text spotting from unmanned aerial vehicles," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 905–913, 2021.

[3] Z. Wu, Z. Wang, Z. Wang, and H. Jin, "Towards privacy-preserving visual recognition via adversarial training: A pilot study," in *ECCV*, 2018.

[4] T. Orekondy, B. Schiele, and M. Fritz, "Towards a visual privacy advisor: Understanding and predicting privacy risks in images," in *ICCV*, 2017.

[5] S. Abu-El-Haija, N. Kothari, J. Lee, P. Natsev, G. Toderici, B. Varadarajan, and S. Vijayanarasimhan, "Youtube-8m: A large-scale video classification benchmark," *arXiv*, 2016.

[6] W. Kay, J. Carreira, K. Simonyan, B. Zhang, C. Hillier, S. Vijayanarasimhan, F. Viola, T. Green, T. Back, P. Natsev, *et al.*, "The kinetics human action video dataset," *arXiv*, 2017.

[7] H. Kuehne, H. Jhuang, E. Garrote, T. Poggio, and T. Serre, "Hmdb: a large video database for human motion recognition," in *ICCV*, 2011.

[8] H. Jhuang, J. Gall, S. Zuffi, C. Schmid, and M. J. Black, "Towards understanding action recognition," in *ICCV*, 2013.

[9] M. Monfort, A. Andonian, B. Zhou, K. Ramakrishnan, S. A. Bargal, T. Yan, L. Brown, Q. Fan, D. Gutfreund, C. Vondrick, *et al.*, "Moments in time dataset: one million videos for event understanding," *TPAMI*, 2019.

[10] G. A. Sigurdsson, G. Varol, X. Wang, A. Farhadi, I. Laptev, and A. Gupta, "Hollywood in homes: Crowdsourcing data collection for activity understanding," in *ECCV*, Springer, 2016.

[11] K. Soomro, A. R. Zamir, and M. Shah, "Ucf101: A dataset of 101 human actions classes from videos in the wild," *arXiv*, 2012.

[12] H. Zhao, A. Torralba, L. Torresani, and Z. Yan, "Hacs: Human action clips and segments dataset for recognition and temporal localization," in *ICCV*, 2019.

[13] J. Donahue, L. Anne Hendricks, S. Guadarrama, M. Rohrbach, S. Venugopalan, K. Saenko, and T. Darrell, "Long-term recurrent convolutional networks for visual recognition and description," in *CVPR*, 2015.

[14] J. Yue-Hei Ng, M. Hausknecht, S. Vijayanarasimhan, O. Vinyals, R. Monga, and G. Toderici, "Beyond short snippets: Deep networks for video classification," in *CVPR*, 2015.

[15] L. Sun, K. Jia, K. Chen, D.-Y. Yeung, B. E. Shi, and S. Savarese, "Lattice long short-term memory for human action recognition," in *ICCV*, 2017.

[16] D. Tran, L. Bourdev, R. Fergus, L. Torresani, and M. Paluri, "Learning spatiotemporal features with 3d convolutional networks," in *ICCV*, 2015.

[17] J. Carreira and A. Zisserman, "Quo vadis, action recognition? a new model and the kinetics dataset," in *CVPR*, 2017.

[18] C. Feichtenhofer, H. Fan, J. Malik, and K. He, "Slowfast networks for video recognition," in *ICCV*, 2019.

[19] C. Feichtenhofer, "X3d: Expanding architectures for efficient video recognition," in *CVPR*, 2020.

[20] K. Simonyan and A. Zisserman, "Two-stream convolutional networks for action recognition in videos," in *NeurIPS*, 2014.

[21] C. Feichtenhofer, A. Pinz, and A. Zisserman, "Convolutional two-stream network fusion for video action recognition," in *CVPR*, 2016.

[22] X. Yang, X. Yang, M.-Y. Liu, F. Xiao, L. S. Davis, and J. Kautz, "Step: Spatio-temporal progressive learning for video action detection," in *CVPR*, 2019.

[23] M. Tomei, L. Baraldi, S. Calderara, S. Bronzin, and R. Cucchiara, "Stage: Spatio-temporal attention on graph entities for video action detection," *arXiv*, 2019.

[24] R. R. A. Pramono, Y.-T. Chen, and W.-H. Fang, "Hierarchical self-attention network for action localization in videos," in *ICCV*, 2019.

[25] C.-Y. Wu, C. Feichtenhofer, H. Fan, K. He, P. Krahenbuhl, and R. Girshick, "Long-term feature banks for detailed video understanding," in *CVPR*, 2019.

[26] J. Wu, Z. Kuang, L. Wang, W. Zhang, and G. Wu, "Context-aware rcnn: A baseline for action detection in videos," in *ECCV*, Springer, 2020.

[27] J. Tang, J. Xia, X. Mu, B. Pang, and C. Lu, "Asynchronous interaction aggregation for action detection," *arXiv*, 2020.

[28] J. Pan, S. Chen, Z. Shou, J. Shao, and H. Li, "Actor-context-actor relation network for spatio-temporal action localization," *arXiv*, 2020.

[29] M. Liao, B. Shi, X. Bai, X. Wang, and W. Liu, "Textboxes: A fast text detector with a single deep neural network," *arXiv*, 2016.

[30] M. Liao, B. Shi, and X. Bai, "Textboxes++: A single-shot oriented scene text detector," *TIP*, 2018.

[31] X. Zhou, C. Yao, H. Wen, Y. Wang, S. Zhou, W. He, and J. Liang, "East: an efficient and accurate scene text detector," in *CVPR*, 2017.

[32] L. Huang, Y. Yang, Y. Deng, and Y. Yu, "Densebox: Unifying landmark localization with end to end object detection," *arXiv*, 2015.

[33] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *International Conference on Medical image computing and computer-assisted intervention*, Springer, 2015.

[34] J. Ma, W. Shao, H. Ye, L. Wang, H. Wang, Y. Zheng, and X. Xue, "Arbitrary-oriented scene text detection via rotation proposals," *TMM*.

[35] C. Zhang, B. Liang, Z. Huang, M. En, J. Han, E. Ding, and X. Ding, "Look more than once: An accurate detector for text of arbitrary shapes," in *CVPR*, 2019.

[36] X. Wang, Y. Jiang, Z. Luo, C.-L. Liu, H. Choi, and S. Kim, "Arbitrary shape scene text detection with adaptive text region representation," in *CVPR*, 2019.

[37] D. Deng, H. Liu, X. Li, and D. Cai, "Pixellink: Detecting scene text via instance segmentation," *arXiv*, 2018.

[38] Y. Wu and P. Natarajan, "Self-organized text detection with minimal post-processing via border learning," in *ICCV*, 2017.

[39] Z. Tian, M. Shu, P. Lyu, R. Li, C. Zhou, X. Shen, and J. Jia, "Learning shape-aware embedding for scene text detection," in *CVPR*, 2019.

[40] W. Wang, E. Xie, X. Li, W. Hou, T. Lu, G. Yu, and S. Shao, "Shape robust text detection with progressive scale expansion network," in *CVPR*, 2019.

[41] Z. Tian, W. Huang, T. He, P. He, and Y. Qiao, "Detecting text in natural image with connectionist text proposal network," in *ECCV*, 2016.

[42] B. Shi, X. Bai, and S. Belongie, "Detecting oriented text in natural images by linking segments," in *CVPR*, 2017.

[43] Y. Baek, B. Lee, D. Han, S. Yun, and H. Lee, "Character region awareness for text detection," in *CVPR*, 2019.

[44] A. Graves, S. Fernández, F. Gomez, and J. Schmidhuber, "Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks," in *ICML*, 2006.

[45] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," *NeurIPS*, 2014.

[46] B. Shi, X. Bai, and C. Yao, "An end-to-end trainable neural network for image-based sequence recognition and its application to scene text recognition," *TPAMI*, 2016.

[47] F. Yin, Y.-C. Wu, X.-Y. Zhang, and C.-L. Liu, "Scene text recognition with sliding convolutional character models," *arXiv*, 2017.

[48] C.-Y. Lee and S. Osindero, "Recursive recurrent nets with attention modeling for ocr in the wild," in *CVPR*, 2016.

[49] F. Bai, Z. Cheng, Y. Niu, S. Pu, and S. Zhou, "Edit probability for scene text recognition," in *CVPR*, 2018.

[50] H. Li, P. Wang, and C. Shen, "Towards end-to-end text spotting with convolutional recurrent neural networks," in *ICCV*, 2017.

[51] P. Wang, H. Li, and C. Shen, "Towards end-to-end text spotting in natural scenes," *arXiv*, 2019.

[52] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," *TPAMI*, 2016.

[53] M. Busta, L. Neumann, and J. Matas, "Deep textspotter: An end-to-end trainable scene text localization and recognition framework," in *ICCV*, 2017.

[54] T. He, Z. Tian, W. Huang, C. Shen, Y. Qiao, and C. Sun, "An end-to-end textspotter with explicit alignment and attention," in *CVPR*, 2018.

[55] X. Liu, D. Liang, S. Yan, D. Chen, Y. Qiao, and J. Yan, "Fots: Fast oriented text spotting with a unified network," in *CVPR*, 2018.

[56] P. Lyu, M. Liao, C. Yao, W. Wu, and X. Bai, “Mask textspotter: An end-to-end trainable neural network for spotting text with arbitrary shapes,” in *ECCV*, 2018.

[57] M. Liao, P. Lyu, M. He, C. Yao, W. Wu, and X. Bai, “Mask textspotter: An end-to-end trainable neural network for spotting text with arbitrary shapes,” *PAMI*, 2019.

[58] M. Liao, G. Pang, J. Huang, T. Hassner, and X. Bai, “Mask textspotter v3: Segmentation proposal network for robust scene text spotting,” in *ECCV*, 2020.

[59] S. Qin, A. Bissacco, M. Raptis, Y. Fujii, and Y. Xiao, “Towards unconstrained end-to-end text spotting,” in *ICCV*, 2019.

[60] W. Feng, W. He, F. Yin, X.-Y. Zhang, and C.-L. Liu, “Textdragon: An end-to-end framework for arbitrary shaped text spotting,” in *ICCV*, 2019.

[61] L. Xing, Z. Tian, W. Huang, and M. R. Scott, “Convolutional character networks,” in *ICCV*, 2019.

[62] Y. Liu, H. Chen, C. Shen, T. He, L. Jin, and L. Wang, “Abcnet: Real-time scene text spotting with adaptive bezier-curve network,” in *CVPR*, 2020.

[63] Y. Baek, S. Shin, J. Baek, S. Park, J. Lee, D. Nam, and H. Lee, “Character region attention for text spotting,” in *ECCV*, 2020.

[64] L. Wang, J. Shi, Y. Wang, and F. Su, “Video text detection by attentive spatiotemporal fusion of deep convolutional features,” in *Proceedings of the 27th ACM International Conference on Multimedia*, pp. 66–74, 2019.

[65] L. Wu, P. Shivakumara, T. Lu, and C. L. Tan, “A new technique for multi-oriented scene text line detection and tracking in video,” *TMM*, 2015.

[66] X.-H. Yang, W. He, F. Yin, and C.-L. Liu, “A unified video text detection method with network flow,” in *ICDAR*, 2017.

[67] S. Tian, W.-Y. Pei, Z.-Y. Zuo, and X.-C. Yin, “Scene text detection in video by learning locally and globally.,” in *IJCAI*, 2016.

[68] Z.-Y. Zuo, S. Tian, W.-y. Pei, and X.-C. Yin, “Multi-strategy tracking based text detection in scene videos,” in *ICDAR*, IEEE, 2015.

[69] Y. Wang, L. Wang, F. Su, and J. Shi, “Video text detection with fully convolutional network and tracking,” in *2019 IEEE International Conference on Multimedia and Expo (ICME)*, pp. 1738–1743, IEEE, 2019.

[70] X. Wang, Y. Jiang, S. Yang, X. Zhu, W. Li, P. Fu, H. Wang, and Z. Luo, “End-to-end scene text recognition in videos based on multi frame tracking,” in *ICDAR*, 2017.

[71] Z. Cheng, J. Lu, B. Zou, L. Qiao, Y. Xu, S. Pu, Y. Niu, F. Wu, and S. Zhou, “Free: A fast and robust end-to-end video text spotter,” *TIP*, 2020.

[72] Z. Cheng, J. Lu, Y. Niu, S. Pu, F. Wu, and S. Zhou, “You only recognize once: Towards fast video text spotting,” in *ACMMM*, 2019.

[73] B. Liu, M. Ding, S. Shaham, W. Rahayu, F. Farokhi, and Z. Lin, “When machine learning meets privacy: A survey and outlook,” *ACM Computing Surveys (CSUR)*, 2021.

[74] C. Dwork, K. Kenthapadi, F. McSherry, I. Mironov, and M. Naor, “Our data, ourselves: Privacy via distributed noise generation,” in *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, Springer, 2006.

[75] C. Dwork, F. McSherry, K. Nissim, and A. Smith, “Calibrating noise to sensitivity in private data analysis,” in *Theory of cryptography conference*, Springer, 2006.

[76] I. Dinur and K. Nissim, “Revealing information while preserving privacy,” in *Proceedings of the twenty-second ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, 2003.

[77] I. Mironov, “On significance of the least significant bits for differential privacy,” in *Proceedings of the 2012 ACM conference on Computer and communications security*, 2012.

[78] I. Mironov, “Rényi differential privacy,” in *2017 IEEE 30th Computer Security Foundations Symposium (CSF)*, IEEE, 2017.

[79] R. Hall, A. Rinaldo, and L. Wasserman, “Differential privacy for functions and functional data,” *The Journal of Machine Learning Research*, 2013.

[80] J. Hamm, “Minimax filter: Learning to preserve privacy from inference attacks,” *The Journal of Machine Learning Research*, 2017.

[81] Y. Cao and J. Yang, “Towards making systems forget with machine unlearning,” in *2015 IEEE Symposium on Security and Privacy*, IEEE, 2015.

[82] T. GINART, M. GUAN, G. VALIANT, and J. ZOU, “Making ai forget you: Data deletion in machine learning,” *Advances in Neural Information Processing Systems*, 2019.

[83] Y. Wu, E. Dobriban, and S. Davidson, “Deltagrad: Rapid retraining of machine learning models,” in *International Conference on Machine Learning*, pp. 10355–10366, PMLR, 2020.

[84] L. Bourtoule, V. Chandrasekaran, C. A. Choquette-Choo, H. Jia, A. Travers, B. Zhang, D. Lie, and N. Papernot, “Machine unlearning,” *arXiv preprint arXiv:1912.03817*, 2019.

[85] R. D. Cook and S. Weisberg, “Characterizations of an empirical influence function for detecting influential cases in regression,” *Technometrics*, 1980.

[86] F. R. Hampel, E. M. Ronchetti, P. J. Rousseeuw, and W. A. Stahel, *Robust statistics: the approach based on influence functions*. John Wiley & Sons, 2011.

[87] P. W. Koh and P. Liang, “Understanding black-box predictions via influence functions,” in *ICML*, PMLR, 2017.

[88] T. Baumhauer, P. Schöttle, and M. Zeppelzauer, “Machine unlearning: Linear filtration for logit-based classifiers,” *arXiv preprint arXiv:2002.02730*, 2020.

[89] F. Pittaluga, S. J. Koppal, S. B. Kang, and S. N. Sinha, “Revealing scenes by inverting structure from motion reconstructions,” in *CVPR*, 2019.

[90] A. Dosovitskiy and T. Brox, “Inverting visual representations with convolutional networks,” in *CVPR*, 2016.

[91] H. Kato and T. Harada, “Image reconstruction from bag-of-visual-words,” in *CVPR*, 2014.

[92] P. Weinzaepfel, H. Jégou, and P. Pérez, “Reconstructing an image from its local descriptors,” in *CVPR*, 2011.

[93] A. Mahendran and A. Vedaldi, “Visualizing deep convolutional neural networks using natural pre-images,” *IJCV*, 2016.

[94] C. Gentry *et al.*, “Fully homomorphic encryption using ideal lattices.,” in *STOC*, 2009.

[95] P. Xie, M. Bilenko, T. Finley, R. Gilad-Bachrach, K. Lauter, and M. Naehrig, “Crypto-nets: Neural networks over encrypted data,” *arXiv*, 2014.

[96] A. Chattopadhyay and T. E. Boult, “Privacycam: a privacy preserving camera using uclinux on the blackfin dsp,” in *CVPR*, 2007.

[97] M. S. Ryoo, B. Rothrock, C. Fleming, and H. J. Yang, “Privacy-preserving human activity recognition from extreme low resolution,” in *AAAI*, 2017.

[98] D. J. Butler, J. Huang, F. Roesner, and M. Cakmak, “The privacy-utility tradeoff for remotely teleoperated robots,” in *HRI*, 2015.

[99] J. Dai, B. Saghafi, J. Wu, J. Konrad, and P. Ishwar, “Towards privacy-preserving recognition of human activities,” in *ICIP*, 2015.

[100] T. Winkler, A. Erdélyi, and B. Rinner, “Trusteye. m4: protecting the sensor—not the camera,” in *AVSS*, 2014.

[101] Z. W. Wang, V. Vineet, F. Pittaluga, S. N. Sinha, O. Cossairt, and S. Bing Kang, “Privacy-preserving action recognition using coded aperture videos,” in *CVPRW*, 2019.

[102] F. Pittaluga and S. J. Koppal, “Privacy preserving optics for miniature vision sensors,” in *CVPR*, 2015.

[103] F. Pittaluga and S. J. Koppal, “Pre-capture privacy for small vision sensors,” *TPAMI*, 2017.

[104] L. Jia and R. J. Radke, “Using time-of-flight measurements for privacy-preserving tracking in a smart room,” *IINF*, 2014.

[105] S. Tao, M. Kudo, and H. Nonaka, “Privacy-preserved behavior analysis and fall detection by an infrared ceiling sensor network,” *Sensors*, 2012.

[106] Z. Wang, S. Chang, Y. Yang, D. Liu, and T. S. Huang, “Studying very low resolution recognition using deep networks,” *CVPR*, 2016.

[107] B. Cheng, Z. Wang, Z. Zhang, Z. Li, D. Liu, J. Yang, S. Huang, and T. S. Huang, “Robust emotion recognition from low quality and low bit rate video: A deep learning approach,” in *ACII*, 2017.

[108] M. Xu, A. Sharghi, X. Chen, and D. J. Crandall, “Fully-coupled two-stream spatiotemporal networks for extremely low resolution action recognition,” in *WACV*, 2018.

[109] Z. Wu, K. Suresh, P. Narayanan, H. Xu, H. Kwon, and Z. Wang, “Delving into robust object detection from unmanned aerial vehicles: A deep nuisance disentanglement approach,” in *ICCV*, 2019.

[110] P. M Uplavikar, Z. Wu, and Z. Wang, “All-in-one underwater image enhancement using domain-adversarial learning,” in *CVPRW*, 2019.

[111] F. Pittaluga, S. Koppal, and A. Chakrabarti, “Learning privacy preserving encodings through adversarial training,” in *WACV*, 2019.

[112] M. Bertran, N. Martinez, A. Papadaki, Q. Qiu, M. Rodrigues, G. Reeves, and G. Sapiro, “Adversarially learned representations for information obfuscation and inference,” in *ICML*, 2019.

[113] P. C. Roy and V. N. Boddeti, “Mitigating information leakage in image representations: A maximum entropy approach,” in *CVPR*, 2019.

[114] B. H. Zhang, B. Lemoine, and M. Mitchell, “Mitigating unwanted biases with adversarial learning,” in *AIES*, 2018.

[115] Z. Ren, Y. Jae Lee, and M. S. Ryoo, “Learning to anonymize faces for privacy preserving action detection,” in *ECCV*, 2018.

[116] R. R. Shetty, M. Fritz, and B. Schiele, “Adversarial scene editing: Automatic object removal from weak supervision,” in *NeurIPS*, 2018.

[117] T. Wang, J. Zhao, M. Yatskar, K.-W. Chang, and V. Ordonez, “Balanced datasets are not enough: Estimating and mitigating gender bias in deep image representations,” in *ICCV*, 2019.

[118] W. Oleszkiewicz, P. Kairouz, K. Piczak, R. Rajagopal, and T. Trzciński, “Siamese generative adversarial privatizer for biometric data,” in *ACCV*, 2018.

[119] Y. Li, N. Vishwamitra, B. P. Knijnenburg, H. Hu, and K. Caine, “Blur vs. block: Investigating the effectiveness of privacy-enhancing obfuscation for images,” in *CVPRW*, 2017.

[120] S. J. Oh, R. Benenson, M. Fritz, and B. Schiele, “Faceless person recognition: Privacy implications in social media,” in *ECCV*, 2016.

[121] R. McPherson, R. Shokri, and V. Shmatikov, “Defeating image obfuscation with deep learning,” *arXiv*, 2016.

[122] S. J. Oh, M. Fritz, and B. Schiele, “Adversarial image perturbation for privacy protection a game theory perspective,” in *ICCV*, 2017.

[123] Z. Shen, S. Fan, Y. Wong, T.-T. Ng, and M. Kankanhalli, “Human-imperceptible privacy protection against machines,” in *ACMMM*, 2019.

[124] D. Gurari, Q. Li, C. Lin, Y. Zhao, A. Guo, A. Stangl, and J. P. Bigham, “Vizwiz-priv: A dataset for recognizing the presence and purpose of private visual information in images taken by blind people,” in *CVPR*, 2019.

[125] S. Zerr, S. Siersdorfer, J. Hare, and E. Demidova, “Picalert!privacy-aware image tools,” 2012.

[126] T. Orekondy, M. Fritz, and B. Schiele, “Connecting pixels to privacy and utility: Automatic redaction of private information in images,” in *CVPR*, 2018.

[127] F. N. Iandola, S. Han, M. W. Moskewicz, K. Ashraf, W. J. Dally, and K. Keutzer, “Squeezenet: Alexnet-level accuracy with 50x fewer parameters and< 0.5 mb model size,” *arXiv preprint arXiv:1602.07360*, 2016.

[128] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, “Mobilenets: Efficient convolutional neural networks for mobile vision applications,” *arXiv*, 2017.

[129] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, “Mobilenetv2: Inverted residuals and linear bottlenecks,” in *CVPR*, 2018.

[130] X. Zhang, X. Zhou, M. Lin, and J. Sun, “Shufflenet: An extremely efficient convolutional neural network for mobile devices,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 6848–6856, 2018.

[131] N. Ma, X. Zhang, H.-T. Zheng, and J. Sun, “Shufflenet v2: Practical guidelines for efficient cnn architecture design,” in *Proceedings of the European conference on computer vision (ECCV)*, pp. 116–131, 2018.

[132] M. Tan and Q. Le, “Efficientnet: Rethinking model scaling for convolutional neural networks,” in *International Conference on Machine Learning*, pp. 6105–6114, PMLR, 2019.

[133] B. Wu, X. Dai, P. Zhang, Y. Wang, F. Sun, Y. Wu, Y. Tian, P. Vajda, Y. Jia, and K. Keutzer, “Fbnet: Hardware-aware efficient convnet design via differentiable neural architecture search,” in *CVPR*, 2019.

[134] S. Han, H. Mao, and W. J. Dally, “Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding,” *arXiv*, 2015.

[135] Y. LeCun, J. S. Denker, and S. A. Solla, “Optimal brain damage,” in *NeurIPS*, 1990.

[136] B. Hassibi and D. Stork, “Second order derivatives for network pruning: Optimal brain surgeon,” in *NeurIPS* (S. Hanson, J. Cowan, and C. Giles, eds.), Morgan-Kaufmann, 1993.

[137] H. Hu, R. Peng, Y.-W. Tai, and C.-K. Tang, "Network trimming: A data-driven neuron pruning approach towards efficient deep architectures," *arXiv*, 2016.

[138] M. Yuan and Y. Lin, "Model selection and estimation in regression with grouped variables," *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 2006.

[139] W. Wen, C. Wu, Y. Wang, Y. Chen, and H. Li, "Learning structured sparsity in deep neural networks," *arXiv*, 2016.

[140] Z. Liu, J. Li, Z. Shen, G. Huang, S. Yan, and C. Zhang, "Learning efficient convolutional networks through network slimming," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 2736–2744, 2017.

[141] Z. Huang and N. Wang, "Data-driven sparse structure selection for deep neural networks," in *ECCV*, 2018.

[142] Y. Han, G. Huang, S. Song, L. Yang, H. Wang, and Y. Wang, "Dynamic neural networks: A survey," *arXiv*, 2021.

[143] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv*, 2018.

[144] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever, "Language models are unsupervised multitask learners,"

[145] N. Parmar, A. Vaswani, J. Uszkoreit, Ł. Kaiser, N. Shazeer, A. Ku, and D. Tran, "Image transformer," *arXiv*, 2018.

[146] F. Yang, H. Yang, J. Fu, H. Lu, and B. Guo, "Learning texture transformer network for image super-resolution," in *CVPR*, 2020.

[147] Y. Zeng, J. Fu, and H. Chao, "Learning joint spatial-temporal transformations for video inpainting," in *ECCV*, Springer, 2020.

[148] R. Girdhar, J. Carreira, C. Doersch, and A. Zisserman, "Video action transformer network," in *CVPR*, 2019.

[149] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, and S. Zagoruyko, "End-to-end object detection with transformers," *arXiv*, 2020.

[150] B. Wu, C. Xu, X. Dai, A. Wan, P. Zhang, M. Tomizuka, K. Keutzer, and P. Vajda, "Visual transformers: Token-based image representation and processing for computer vision," *arXiv preprint arXiv:2006.03677*, 2020.

[151] X. Wang, R. Girshick, A. Gupta, and K. He, "Non-local neural networks," in *CVPR*, 2018.

[152] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, *et al.*, "An image is worth 16x16 words: Transformers for image recognition at scale," *arXiv*, 2020.

[153] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, *et al.*, "Imagenet large scale visual recognition challenge," *IJCV*, 2015.

[154] P. Esser, R. Rombach, and B. Ommer, "Taming transformers for high-resolution image synthesis," *arXiv*, 2020.

[155] G. Bertasius, H. Wang, and L. Torresani, "Is space-time attention all you need for video understanding?," *arXiv*, 2021.

[156] C. H. Donna Lu, "How abusers are exploiting smart home devices?," October 2019.

[157] M. W. Tobias, "Is your smart security camera protecting your home or spying on you?," August 2016.

[158] D. Harwell, "Doorbell-camera firm ring has partnered with 400 police forces, extending surveillance concerns," August 2019.

[159] B. Heater, "Amazon's camera-equipped echo look raises new questions about smart home privacy," April 2017.

[160] T. Brewster, "Thousands of banned chinese surveillance cameras are watching over america," August 2019.

[161] T. U. S. D. of Justice, "Privacy act of 1974, as amended, 5 u.s.c. § 552a," 1974.

[162] M. A. Weiss and K. Archick, "Us-eu data privacy: from safe harbor to privacy shield," 2016.

[163] T. U. S. D. of Homeland Security, "Passenger name records agreements," 2004.

[164] R. Leenes, R. Van Brakel, S. Gutwirth, and P. De Hert, *Data protection and privacy:(In) visibilities and infrastructures*. 2017.

[165] J. Hamm and Y.-K. Noh, "K-beam minimax: Efficient optimization for deep adversarial learning," in *ICML*, 2018.

[166] Y. Ganin and V. Lempitsky, "Unsupervised domain adaptation by backpropagation," in *ICML*, 2015.

[167] X. Xiang and T. D. Tran, "Linear disentangled representation learning for facial actions," *TCSVT*, 2018.

[168] G. Desjardins, A. Courville, and Y. Bengio, "Disentangling factors of variation via generative entangling," *arXiv*, 2012.

[169] A. Gonzalez-Garcia, J. van de Weijer, and Y. Bengio, "Image-to-image translation for cross-domain disentanglement," *arXiv*, 2018.

[170] S. Reddy, I. Labutov, S. Banerjee, and T. Joachims, "Unbounded human learning: Optimal scheduling for spaced repetition," in *SIGKDD*, 2016.

[171] J. Johnson, A. Alahi, and L. Fei-Fei, "Perceptual losses for real-time style transfer and super-resolution," in *ECCV*, 2016.

[172] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," *CVPR*, 2016.

[173] D.-Z. Du and P. M. Pardalos, *Minimax and applications*. 2013.

[174] B. G. Fabian Caba Heilbron, Victor Escorcia and J. C. Niebles, "Activitynet: A large-scale video benchmark for human activity understanding," in *CVPR*, 2015.

[175] C. Gu, C. Sun, D. A. Ross, C. Vondrick, C. Pantofaru, Y. Li, S. Vijayanarasimhan, G. Toderici, S. Ricco, R. Sukthankar, *et al.*, "Ava: A video dataset of spatio-temporally localized atomic visual actions," in *CVPR*, 2018.

[176] B. Zoph, V. Vasudevan, J. Shlens, and Q. V. Le, "Learning transferable architectures for scalable image recognition," in *CVPR*, 2018.

[177] K. Yun, J. Honorio, D. Chattopadhyay, T. L. Berg, and D. Samaras, "Two-person interaction detection using body-pose features and multiple instance learning," in *CVPRW*, 2012.

[178] K. He, X. Zhang, S. Ren, and J. Sun, "Identity mappings in deep residual networks," in *ECCV*, 2016.

[179] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *CVPR*, 2015.

[180] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *CVPR*, 2016.

[181] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *ICLR*, 2015.

[182] B. Cheng, Y. Wei, H. Shi, R. Feris, J. Xiong, and T. Huang, "Revisiting rcnn: On awakening the classification power of faster rcnn," in *ECCV*, 2018.

[183] B. Cheng, Y. Wei, H. Shi, R. Feris, J. Xiong, and T. Huang, "Decoupled classification refinement: Hard false positive suppression for object detection," *arXiv*, 2018.

[184] J. Wu, X. Yu, D. Liu, M. Chandraker, and Z. Wang, "David: Dual-attentional video deblurring," in *WACV*, 2020.

[185] O. Kupyn, T. Martyniuk, J. Wu, and Z. Wang, "Deblurgan-v2: Deblurring (orders-of-magnitude) faster and better," in *ICCV*, 2019.

[186] X. Yi and M. Eramian, "Lbp-based segmentation of defocus blur," *TIP*, 2016.

[187] J. L. Pech-Pacheco, G. Cristóbal, J. Chamorro-Martinez, and J. Fernández-Valdivia, "Diatom autofocusing in brightfield microscopy: a comparative study," in *ICPR*, IEEE, 2000.

[188] S. Liang, Y. Li, and R. Srikant, “Enhancing the reliability of out-of-distribution image detection in neural networks,” in *ICLR*, 2018.

[189] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, “Grad-cam: Visual explanations from deep networks via gradient-based localization,” in *ICCV*, 2017.

[190] “Usb power meter measurement github repository.” `https://github.com/kolinger/rd-usb.git/`.

[191] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, “Attention is all you need,” in *NeurIPS*, 2017.

[192] Y. Bengio, J. Louradour, R. Collobert, and J. Weston, “Curriculum learning,” in *ICML*, 2009.

[193] J. Carreira, E. Noland, A. Banki-Horvath, C. Hillier, and A. Zisserman, “A short note about kinetics-600,” *arXiv preprint arXiv:1808.01340*, 2018.

[194] J. Carreira, E. Noland, C. Hillier, and A. Zisserman, “A short note on the kinetics-700 human action dataset,” *arXiv preprint arXiv:1907.06987*, 2019.

[195] I. Loshchilov and F. Hutter, “Decoupled weight decay regularization,” *arXiv*, 2017.

[196] Z. Dai, B. Cai, Y. Lin, and J. Chen, “Up-detr: Unsupervised pre-training for object detection with transformers,” *arXiv*, 2020.

[197] C. Sun, A. Shrivastava, C. Vondrick, K. Murphy, R. Sukthankar, and C. Schmid, “Actor-centric relation network,” in *ECCV*, 2018.

[198] Y. Zhang, P. Tokmakov, M. Hebert, and C. Schmid, “A structured model for action detection,” in *CVPR*, 2019.

[199] X. Peng and C. Schmid, “Multi-region two-stream r-cnn for action detection,” in *ECCV*, Springer, 2016.

[200] R. Hou, C. Chen, and M. Shah, “Tube convolutional neural network (t-cnn) for action detection in videos,” in *ICCV*, 2017.

[201] V. Kalogeiton, P. Weinzaepfel, V. Ferrari, and C. Schmid, “Action tubelet detector for spatio-temporal action localization,” in *ICCV*, 2017.

[202] A. Rives, S. Goyal, J. Meier, D. Guo, M. Ott, C. L. Zitnick, J. Ma, and R. Fergus, “Biological structure and function emerge from scaling unsupervised learning to 250 million protein sequences,” *bioRxiv*, p. 622803, 2019.

[203] D. Noever, M. Ciolino, and J. Kalin, “The chess transformer: Mastering play using generative language models,” *arXiv preprint arXiv:2008.04057*, 2020.