Automated Visual Weak Supervision for Object Recognition in Videos

Swetha Revanur
Stanford University
srevanur@cs.stanford.edu

Vishnu Sarukkai
Stanford University
sarukkai@stanford.edu

Abstract

The quantity of video data is increasing en masse, and it's rapidly becoming intractable to employ human labeling. In this paper, we propose a pipeline for automated visual weak supervision for video classification. In our architecture, convolutional autoencoders encode regions of interest from video frames that are linked to produce action tubes. These action tubes are used to programmatically generate heuristics for weak labeling of videos with the Reef framework. By augmenting a CNN-based classifier that discriminates between two vehicle types with the weak labels, we saw dramatic gains in classification accuracy. These results highlight the potential of this weak supervision pipeline on visual data.

1. Introduction

As cameras become an increasingly ubiquitous part of modern life, videos offer a valuable source of data for a variety of applications. A significant number of video cameras are stationary, from traffic cameras, to home security and geostationary satellites. However, it is intractable for humans to manually observe most of this incoming content. As a result, in order to fully utilize the data that is available to us, it is crucial to be able to quickly semantically parse the video, starting with entity classification.

1.1. Overview of Weak Supervision

One method for maximizing the utility of available data involves weak supervision, where a variety of heuristics and models can be created with a small number of labeled data points and used to predict labels for a large number of unlabeled points, which are then used as part of a training data set. For instance, if there exist 100 labeled data points and 900 unlabeled data points, a traditional approach would train a model on the 100 labeled points, but a weakly supervised approach would generate heuristics to "guess" the labels of the remaining 900 points, then use all 1000 points in training the model, trading certainty in the labels of the points in exchange for having a larger training set.

1.2. Approach

This paper addresses two key issues involving weak supervision and object detection in videos. The first focus is creating an automated pipeline for automating the generation of heuristics for visual weak supervision. The second is exploring the trade-off between the number of weakly-labeled data points used in training a subsequent classification model for videos. The input to our algorithm is a video of an intersection, where vehicles are driving through at varying speeds and from varying directions. We then use a convolutional autoencoder to express individual regions of interest within video frames as one-dimensional vectors. Next, we pass a small number of labeled frames and a large number of unlabeled frames to the Reef framework [10] in order to automate the weak labeling of the unlabeled frames as containing either a car or a truck. These frame-by-frame labels are consolidated to create noisy labels for entire videos of vehicles. Finally, we use both the hand-labeled and algorithmically-labeled data in order to train a convolutional neural network (CNN) that labels unseen videos as containing either a car or a truck. In summary, our pipeline will take in clips of vehicles and generate vehicle labels for each of the clips, utilizing an automated weak supervision pipeline in the process and aiming to maximize the relative performance of a classifier using weakly-labeled data relative to a classifier only using hand-labeled data.

This work is important because videos are particularly time-consuming to label by hand, and the growing Internet of Things (IoT) ecosystem is making far more video data available than in the past. In addition, not much work has been done on applying weak supervision to image labeling, let alone video labeling, so this research addresses a relatively unexplored area in the field.

2. Related Work

The task of object recognition in videos is one that has attracted considerable interest over the past few years. In 2007, "Real-time Object Classification in Video Surveillance Based on Appearance Learning" [1] approached this task in real-time through using a regression tree algorithm

with AdaBoost. As computing technology has steadily improved, the more computationally intensive CNNs became a new alternative for approaching the video classification problem. A 2014 paper which approached the problem using CNNs was "Large-scale Video Classification with Convolutional Neural Networks" [3], which trained on the UCF-101 action recognition dataset and suggests a variety of convolutional architectures for the classification task at hand, improving significantly on the UCF-101 baseline model (63.3% success rate vs 43.9%). However, these papers use large, well-labeled datasets, and they do not tackle the weak supervision problem.

A common approach to using weak supervision with images involves applying weak supervision not to the image itself, but rather to the captions associated with the labeled images. "Constrained Convolutional Neural Networks for Weakly Supervised Segmentation" [4] approaches an image segmentation task by applying weak supervision to generate latent probability distributions for objects in the image based on the caption, using a variety of linear constraints in the weak supervision. However, this type of approach is dependent on a preexisting caption for the image, and with the type of data this paper is concerned with, captions are often not available.

An interesting approach which focuses on the image itself, but abandons labels entirely is "Discriminative Unsupervised Feature Learning with Convolutional Neural Networks" [2], which creates surrogate classes through applying a variety of transformations to a given image and assigning all the transformed images to the same class, then applies discriminative functions to distinguish across classes. Interestingly, this approach works better with smaller datasets because the surrogate classes generated through data augmentation are more easily distinguishable. In the future, data augmentation with surrogate classes would be an interesting avenue to pursue alongside weak supervision with videos as well. In addition Ahsan et al. [9] applies deep convolutional GANs (DCGANs) with a weak supervision approach to initialize the weights for a supervised model later used for classification. Ahsan's approach is an interesting alternative in that it uses unlabeled data to improve the model's weight initialization rather than augment the data available to the model for training. Ahsan and Dosovitskiy suggest that using an unsupervised approach to at least extract features from the input videos is a worthwhile path to explore in weakly supervised video classification.

In tackling large volume of video, the contributions of Ye [5] and Wang et al. [7] are very useful. Specifically, Ye discusses automatic event discovery in very large video datasets, while Wang et al. extends this to untrimmed videos. As these papers deal with fairly long, untrimmed videos, the ideas proposed with Ye and Wang are very relevant to the problem being discussed. Ng et. al. approached full-length video classification on the UCF-101 dataset with CNN-based LSTMs [6], performing much better than the traditional benchmarks at the time by bringing CNNs from being used mainly just with images to being used with videos as well. Wang et al. also explores extremely weak supervision, so that a supervised task can become increasingly unsupervised[8].

Finally, Stanford InfoLab's Reef project [10] produced an automated pipeline for weak supervision of text classification. Unlike Ahsan's DCGANs or Dosovitskiy's surrogate classes, Reef places an emphasis on generating heuristics that are relatively simple and interpretable. In addition, by automating selection of the best heuristics available from a search space of k-nearest-neighbor, logistic regression, and decision-tree heuristics, Reef makes it possible to test far more heuristics than manual approaches to weak supervision and makes the task much more tractable. Unfortunately, Reef is only meant to function for one-dimensional textual data. This paper aims to extend Reef as a pipeline for weakly supervising video classification by refactoring the Reef framework to work on condensed feature representations of images and videos.

3. Methods

3.1. Overview

This paper proposes an end-to-end pipeline enabling the automatic visual weak labeling of videos followed by the use of the aforementioned data in a CNN classifier. In preparation for analysis, all frames in the video are cropped to their regions-of-interest using bounding box data and padded to become a consistent 224×224 size. These cropped frames are simply referred to as frames in the remainder of the paper. After preprocessing, labeled frames are passed from the dataset in order to train a convolutional autoencoder (CAE). The CAE is used to extract a 1D vector of features from 3D frames (the third dimension being channel). These 1D vectors are grouped and passed into the Reef framework in order to weakly label the remaining frames in the training dataset. Finally, the labeled videos are used to train a CNN classifier that predicts whether the object in the video is a car or a truck.

4. Dataset and Features

The dataset we are using for this paper is a 19-hour video from a stationary camera near a four-way street intersection in Jackson Hole, Wyoming. This video contains 312321 frames. In addition, we have also obtained bounding boxes for the vehicles present in each frame of the video. These bounding boxes are associated with 4205 different vehicles. Though the task of identifying bounding boxes for objects in video over time is an important one, we believe that there

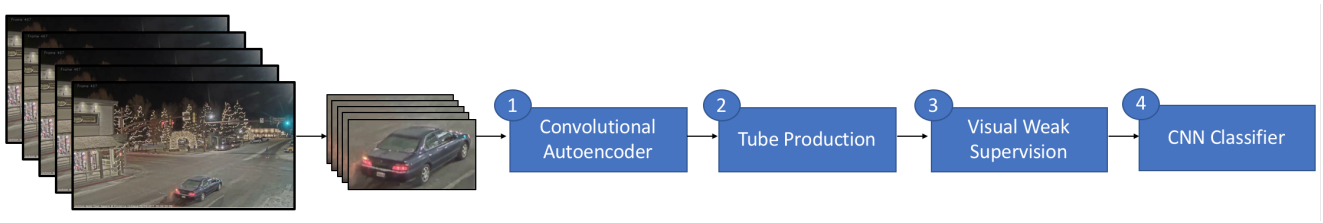

Figure 1. Visual weak supervision pipeline for video action tubes.

is already significant literature that has tackled this problem. Rather, in this paper we focus on the problem of labeling the data effectively with weak supervision for vehicle classification. All models described were implemented with the PyTorch framework.

4.1. Encoding

The unsupervised CAE aims to extract a 1D high-level feature representation from a 3D frame. CAEs aim to maximize the information contained in a 1-D "encoded" form of the image by training an encoder and decoder together where an image is passed into the encoder, producing an "encoded" vector, the vector is decoded, and the original image is compared to the final image. The two images are compared because, the better the decoder is at being able to replicate the original image, it implies that the encoder has successfully stored more of the information from the original image. Binary cross-entropy (BCE) loss between these two images is used as a metric for the difference between the images. This metric is optimal for measuring autoencoder success because it aims to measure the difference between two distributions with values ranging between 0 and 1, as expressed by the following formula:

$$\sum_{i} H(y_i', y_i),$$

$$H(p, q) = -\sum_{\forall x} p(x) log(q(x))$$

. The structure of the network involves three convolutional layers and a fully connected layer in the encoder, separated by ReLU activation and max pooling layers, and a fully connected layer followed by three transpose convolution layers in the decoder, separated by ReLU activation and batch normalization layers. The convolutional layers in the encoder are chosen to capture complex spatial features from the original image, and the transpose convolutional layers in the decoder are intended to replicate these features in the decoded image. Max pooling is used in the encoder to maximize the receptive field of each of the features in the encoded form of the image. A sigmoid function is used at the end of the decoder in order to make sure its values are in the range from 0 to 1, which is the only range appropriate for an image tensor in PyTorch. A 3x224x224 image is reduced to an encoded 512-feature vector.

4.2. Action Tube Production

When a vehicle drives through the intersection in the video, it appears in multiple frames. We can link these frames to produce action tubes that describe the motion of the vehicle over time. Our task of vehicle classification in the large-scale video can now be feasibly reduced to the task of classifying action tubes.

Here, we produce action tubes that span 30 frames to match the video stream's frame rate. Vehicles that appear for fewer frames are ignored in the remainder of this work. For vehicles that appear in more than the allowed tube length, 30 frames that feature the vehicle are randomly sampled from a uniform distribution. For memory efficiency, the encoded version of each frame is stored in the tube. The ground truth label of every tube is the majority vote of the ground truth labels of its constituent frames from the original dataset. The labels are selectively ignored in subsequent steps. We used this ground truth to balance vehicle classes.

4.3. Visual Weak Supervision with Reef

4.3.1 Dataset Partition

As mentioned previously, the Reef framework was built for one-dimensional textual input. To take advantage of Reef's automatic heuristic generation, we encoded frames using a convolutional autoencoder and then organized these frames into tubes. Reef requires two datasets with a non-traditional naming convention: an unlabeled train set and a labeled validation set. We randomly selected 30 tubes to be in the labeled validation set, and allocated 300 tubes for the unlabeled train set.

4.3.2 Automated Heuristic Generation

Using the labeled dataset, Reef generates a variety of heuristics including, k-nearest neighbors, decision trees, and logistic regression models. F1 score, a combination of precision and recall, is used to determine a β-parameter for each heuristic, which is the minimum confidence label needed for a heuristic to assign a label.

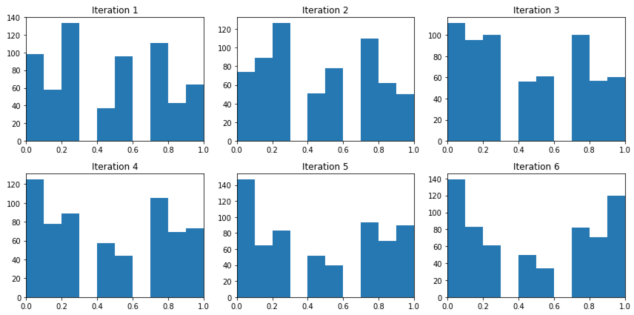

Figure 2. Distribution of probabilistic labels Reef assigns to the training set in the first few iterations. Low confidence points are near 0.5.

$$\text{F1} = 2 * \frac{\text{precision} * \text{recall}}{\text{precision} + \text{recall}}$$

These β-values and F1 scores are used in turn to prune the set of heuristics, eliminating all but the top heuristics that perform well on the metrics. These top heuristics are able to classify outstanding points that previous heuristics were unable to classify. Finally, the data points with low confidence labels are passed back through the process in order to generate more heuristics in an iterative process.

4.4. Classifier

4.4.1 Dataset Partition

Following weak supervision with Reef, all tubes in the previously unlabeled training set have been tagged with noisy labels. From this point, we can proceed in a traditional supervised approach, beginning with a re-partition step. The sets passed into Reef were merged and shuffled, and a new train and validation set were sampled with an 80:20 ratio. These sets are passed into the binary classifier described below.

4.4.2 Model Overview

The binary classifier is a convolutional neural network (CNN) that accepts the two-dimensional input of the encoded frames in an action tube, or a 30×512 matrix representing 30 frames and 512 features. The CNN then applies a series of 2D convolutional layers and fully connected layers to output a binary classification score, normalized with a sigmoid transformation. The weights are initialized using a normal distribution according to the He initialization, which adjusts the variance of the distribution based on the number of features in the layer. Convolutional layers with kernel size 3 are used in order to maximize the receptive field of the stacked filters relative to the number of parameters needed. We use cross-entropy loss (which is, in effect, the same as logistic regression loss for binary classification) for a similar reason to why it was used with the CAE: it is a good measure of similarity between two labels in the range (0, 1) and approximates the KL-divergence.

Note that it would have been a possibility to write a 3D CNN on the raw list of frames in the video rather than a 2-Dimensional CNN on the encoded list of frames. However, having already created a high-level feature representation in the encodings, it seems unnecessary to discard this entirely and learn new high-level feature representations in a 3-Dimensional CNN. Therefore, we chose to use a 2-Dimensional CNN instead, which convolves across both space and time, capturing more complex relationships between features across time than a fully-connected network would.

We use Adam optimization because of the advantages associated with bias correction combined with the strengths of Adagrad (normalizing gradients along dimensions and making sure that none of the dimensions are neglected to help avoid saddle points).

5. Experiments and Results

There are three main models used in the pipeline of this paper: the CAE, the refactored Reef weak supervision model, and the final CNN classifier. The CAE optimizes a binary cross entropy loss in order to measure the quality of the encoding and decoding network structure. The Reef weak supervision model uses the accuracy of its weakly generated labels (percentage correct) in order to evaluate its performance. The CNN classifier uses cross-entropy loss to evaluate the outputs from the final softmax layer in the network. We will address the experiments conducted for each model and the results obtained.

5.1. Experimenting with Encoding Models and Reef

We experimented with various different encoding models. First, we tried two different architectures for the CAE. The first potential architecture involved two convolutional layers and two fully connected layers in the encoder, and two fully connected layers in the decoder. The second potential structure attempted involved three convolutional layers and one fully connected layer in the encoder, and three deconvolutional layers and one fully connected layer in the decoder. Both were trained for 30 epochs, at which point loss began to plateau. A logarithmic hyperparameter sweep on learning rates was performed. While the first approach proved faster to train, the second ultimately proved more successful, with a final loss of 0.259.

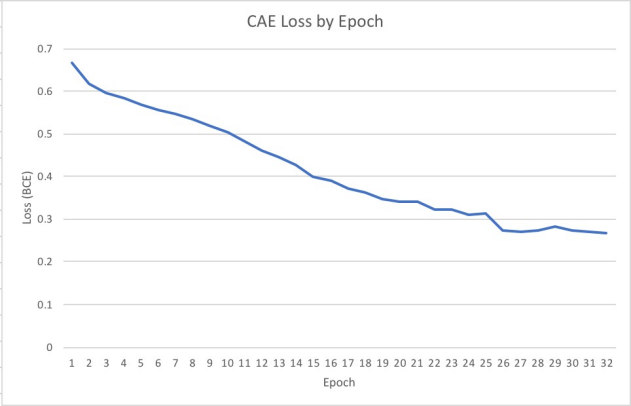

Figure 3. Loss curve of tuned, second CAE architecture.

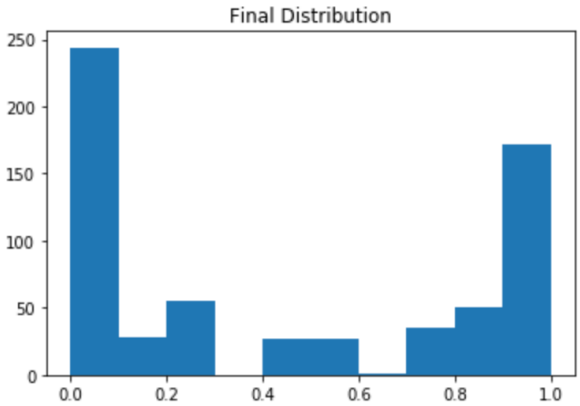

Figure 4. Final distribution of Reef's probabilistic labels with CAE. Most points are in high confidence regions.

Since the objective of this model was to produced a reduced representation of the high-level features of the image, we also tried passing the images into two pretrained models: ResNet-18 and ResNet-50. Extracting the outputs of a forward pass just before the final fully connected layer, we were also able to obtain a one-dimensional code in this manner. We tested the efficacy of the CAE and ResNet approaches by seeing how the encodings fared once passed into Reef.

In running the Reef module, we first tested the outputs of the four candidate encoder models. Taking the inputs of each model into Reef, we fine-tuned the three models independently to find the optimal hyperparameters. Specifically, we compared k nearest neighbors with $k = 5$, decision trees, and logistic regression methods for synthesizing heuristically.We also modulated the number of iterations until performance plateaued. See the collection of Jupyter notebooks for a detailed analysis of Reef tuning here: https://github.com/swetharevanur/cs231n-final-project.

The CAE model performed significantly better than the other two models. This suggests that, for this particular dataset, the benefits of using a large, pretrained network were not worth the cost of sacrificing a network specifically trained for the domain. Further tuning determined that the Reef module performed best when taking inputs from the CAE model if it used k-nearest-neighbors as the heuristic generator. The CAE model achieved 0.58 accuracy and complete coverage this way. Given that Reef labeled 300 points from 30, these results are promising.

While the approach yielded improvements in accuracy over the other methods, k-nearest-neighbors in particular yielded large improvements in coverage, which is not surprising given that logistic regression can often output values close to 0.5, whereas a k-nearest-neighbor model will nearly always classify a point as 0 or 1. One parameter which could have quite possibly improved our results was the cardinality of the heuristics generated: the number of input features involved in creating the heuristic. Since Reef conducts an exhaustive search of the entire space of possible heuristics, a cardinality of 1 implies that one heuristic is generated for every feature in the encoded image–in our case 512 features. Unfortunately, this also implies that, with a cardinality of 2, $\binom{512}{2}$ heuristics needed to be generated in every loop, beyond the computational resources available to us in this project.

5.2. Modulating the Volume of Weak Labels in Binary Classifier

Finally, we trained the CNN classifier model by sweeping over two parameters: learning rate, and the number of weakly labeled data points added to the training dataset (referred to as weak volume). The core 30 labeled videos (the labeled validation set from Reef) was augmented with either 100, 200, or 300 additional weakly labeled points to create a much larger dataset. In addition, as a baseline model, we also trained a model using no weakly labeled points at all. For each model, we recorded the accuracy of the labeling (the percentage of the data points which were predicted correctly), the training loss (the cross-entropy loss being maximized in training), the validation cross-entropy loss, and ROC curves for the validation datasets. The learning rate invariance shown below indicates that with a relatively small dataset, the loss landscape is simple. Note the gains in accuracy that are evident once weakly-labeled points are introduced.

Weak Volume	lr = 1e-5	lr = 1e-4	lr = 1e-3	lr = 1e-2
0	0.500	0.667	0.667	0.333
100	0.769	0.808	0.769	0.769
200	0.739	0.739	0.739	0.739
300	0.712	0.712	0.712	0.712

Figure 5. Validation accuracies.

Weak Volume	lr = 1e-5	lr = 1e-4	lr = 1e-3	lr = 1e-2
0	0.3	0.357	0.557	0
100	0.557	0.528	0.557	0.557
200	0.557	0.557	0.557	0.557
300	0.557	0.557	0.557	0.557

Figure 6. Test accuracies.

The best-performing model achieves an AUROC of 0.773 and the receiver operating characteristic curve is shown:

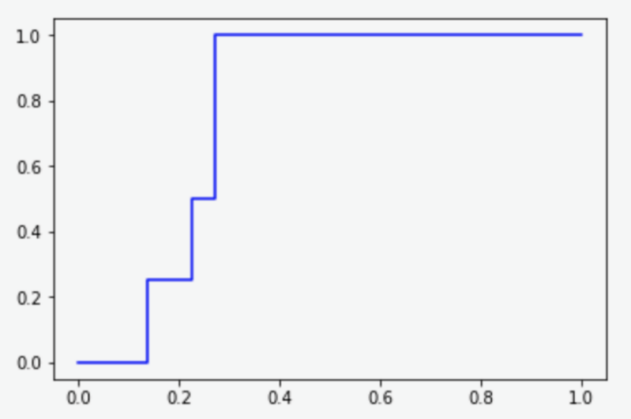

Figure 7. ROC curve with AUC of 0.773.

6. Discussion

It is necessary to contextualize these results with two points. First, the primary goal of this paper, after completing a visual weak supervision pipeline, was to maximize the performance of a classifier with weakly-supervised data relative to a model that did not utilize any weakly-supervised data. As indicated in the tables above, these objectives were clearly met. Nevertheless, binary classification in video is far from an intractable problem and has been addressed extensively in the field. Though the datasets we used were extremely small (drawing on a training set of only 24 videos when not augmented by weakly-labeled points), the performance of the more complex models lagged behind benchmarks, at times performing not much better than a random model.

In order to address the weaknesses of the classifier, a few different ideas come to mind. First of all, training the autoencoder for more epochs might have enabled the extraction of richer features from the matrix, in turn improving the accuracies of models later in the pipeline. Next, it is possible that a CNN was not the best classifier to use on a two-dimensional matrix where one dimension was time, and the other dimension stored a variety of high-level features. For instance, an LSTM-based model may have been equally if not more appropriate for capturing relationships between the features over time. Further, obtaining a larger dataset would have avoided problems of overfitting involved with deeper neural networks in this paper. Finally, actually simplifying the classifier may have helped in this particular case where data was limited.

As for the first objective of the paper, the accuracy obtained with the best model generally outperforms the baseline model without weak supervision. Further accuracy gains might be possible with improvements to the weak supervision framework, in particular generating candidate heuristics meant for images with more of a spatial aspect.

7. Conclusion and Future Work

The highest-performing pipeline we could develop consisted of the following structure: a CAE used for encoding a 1-d feature representation of image frames, followed by k-nearest-neighbors methods used in Reef for generating weak supervision heuristics, followed by the use of a 2-d CNN classifier for the classifier. Key points discovered along the way was the value of using transpose convolutional layers in the decoder of a CAE, the potential of k-nearest-neighbors heuristics for weak labeling, and the value of including weakly-labeled points in the training dataset for a video classifier, especially when very few hand-labeled points were available. Importantly, we successfully adapted the Reef framework in order to use one-dimensional encodings of images as inputs. To this best of our knowledge, this is the first automated visual weak supervision pipeline. In addition, by successfully improving the performance of a classifier with the weakly labeled data, we demonstrated a proof of concept for a fully automated pipeline for generating and using weakly-labeled data in video classification.

There are a variety of experiments which we were unable to run due to constraints on time and compute. Most promising among these experiments is the possibility of using higher cardinalities in the Reef heuristic generator. Perhaps to make this more computationally feasible, we could modify Reef to randomly sample candidate functions from the expanded candidate space when cardinality is greater than 1 rather than exhaustively search the entire space of

candidate functions. Further, Reef, with some small modifications, could potentially handle multi-class classification, not just binary classification. In addition, expanding the set of possible candidate functions to some functions more appropriate for image data might have been helpful as well.

Many of the networks trained in this paper were relatively shallow = especially when constrained by compute and time, the classification and autoencoder models were more used as proof-of-concept for creating an effective weak supervision pipeline than viewed as the objectives themselves. As a result, this paper focuses more on the relative performance of models with and without weakly labeled data rather than on the absolute performance of the models themselves. Nevertheless, we acknowledge the relative non-complexity of the binary classification problem, and look forward to improving its classification capacity beyond weak supervision.

8. Appendices

8.1. Tube Normalization

A direction we initially pursued was action detection from action tubes. Before we decided to redefine our task, we made some progress in that vein. We implemented what we call tube normalization. Tube normalization is a two-step process that reduces varying volume action tubes to a consistent single 20x20 pixel image. First, each frame is padded such that its new dimensions are divisible by 20. Next, spatial max pooling is applied to transform each frame in the tube to a 20x20 pixel image. Finally, we apply temporal max pooling to generate a single 20x20 pixel image that serves as a summary frame of the entire tube (ergo for every unique object). All frames discussed thus far have a third dimension, which contains the RGB channel metadata. With these summary frames we hope to explore object actions in the future. For instance, we can qualitatively see a light streak going from right to left in the summary frame. Indeed, this represents the movement of the car.

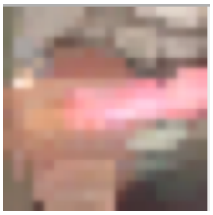

Figure 8: A car and its associated summary frame.

9. Contributions, Acknowledgements

Swetha Revanur and Vishnu Sarukkai both contributed equally to all parts of the project, including but not limited to designing the model architectures, training the autoencoders, adapting Reef for weak supervision of image labeling, and the final classifier. In addition, both contributed equally to the writing of this paper. Credit for the Reef weak supervision module goes to Stanford Infolab.[10] Credit for obtaining the video dataset of an intersection in Jackson, Wyoming with bounding boxes for vehicles goes to Daniel Kang at Stanford Infolab.[11] In addition, Paroma Varma from Stanford Infolab provided guidance by suggesting we retrofit the Reef infrastructure for our project. Finally, credit is due to the CS231N class for providing us with enough Google Cloud credits to conduct our research and run our experiments.

References

[1] L. Zhang et al. "Real-time Object Classification in Video Surveillance Based on Appearance Learning". In: *2007 IEEE Conference on Computer Vision and Pattern Recognition*. June 2007, pp. 1–8. DOI: 10.1109/CVPR.2007.383503.

[2] Alexey Dosovitskiy et al. "Discriminative Unsupervised Feature Learning with Convolutional Neural Networks". In: *Advances in Neural Information Processing Systems 27*. Ed. by Z. Ghahramani et al. Curran Associates, Inc., 2014, pp. 766–774. URL: http://papers.nips.cc/paper/5548-discriminative-unsupervised-feature-learning-with-convolutional-neural-networks.pdf.

[3] Andrej Karpathy et al. "Large-scale Video Classification with Convolutional Neural Networks". In: 2014.

[4] Deepak Pathak, Philipp Krähenbühl, and Trevor Darrell. "Constrained Convolutional Neural Networks for Weakly Supervised Segmentation". In: *CoRR* abs/1506.03648 (2015). arXiv: 1506.03648. URL: http://arxiv.org/abs/1506.03648.

[5] G. Ye. *Large-Scale Video Event Detection*. 2015.

[6] Joe Yue-Hei Ng et al. “Beyond Short Snippets: Deep Networks for Video Classification”. In: *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2015.

[7] L. Wang. *UntrimmedNets for Weakly Supervised Action Recognition and Detection*. 2017.

[8] L. Wang et al. “Video Object Discovery and Co-Segmentation with Extremely Weak Supervision”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 39.10 (Oct. 2017), pp. 2074–2088. ISSN: 0162-8828. DOI: 10.1109/TPAMI.2016.2612187.

[9] U. Ahsan. *DiscrimNet: Semi-Supervised Action Recognition from Videos using Generative Adversarial Networks*. 2018.

[10] Stanford Infolab. *Reef*. https://github.com/HazyResearch/reef. 2018.

[11] Daniel Kang. *Jackson Hole Video Dataset*. 2018.