Automatic Drift Detection for Online Prediction Systems

Joseph Ganley, *University of California Los Angeles*

Abstract

Widespread deployment of video cameras has fueled the growth in research into efficient video analytics systems. Many of these systems involve the use of specialized neural networks which are trained on the data coming from a specific video feed. One of the primary limitations of these systems is that the video feed can change, leaving the specialized model ill-equipped to make predictions. This paper proposes and motivates further research into systems that can cheaply track model drift and indicate when the specialized neural network needs to be retrained.

1. Introduction

The pervasive deployment of video cameras has provided ample data for growing research into video analysis techniques. Many of these techniques involve the use of specialized neural networks which are cheaper to run than more general object detection models such as YOLOv3 [1, 2]. These models are able to achieve similar accuracies to the more general models because they are trained on data specifically from the video feed that they are tasked to analyze. While this model specialization is incredibly effective in reducing computational cost, accuracy can be reduced when the video data drifts too far from the video data used to train the model initially. This phenomenon is referred to as model drift or concept drift and is a known problem with online prediction systems [3, 4].

Model drift can occur gradually as a result of lighting and coloring changes in a video feed or it can change more abruptly as a result of a sudden weather changes such as rainfall or lightning. An example of model drift is provided in figure 1, which shows the difference in coloring and lighting for two video frames at different times throughout a day. Whether the change in video characteristics is gradual or abrupt, it is likely that the model accuracy will suffer if the characteristics of the new video feed are too far removed from the characteristics of the video data used to train the model.

The primary challenge is that it is generally quite expensive to detect accuracy degradation in a specialized model as a result of model drift. The easiest way to detect accuracy degradation is to periodically apply a YOLOv3 model to the new video data to serve as a ground truth label. The ground truth data can then be used to determine the accuracy of the specialized model. However, periodically running a YOLOv3 model on the new data is quite computationally expensive.

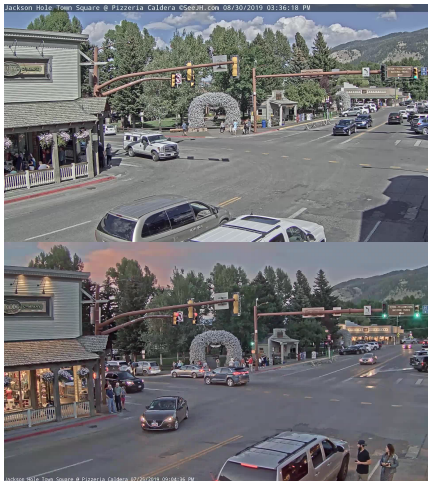

Figure 1: This depicts two images from the same video feed at Jackson Hole Town Square in Wyoming [5]. The upper picture was taken at 3:36 p.m. while the bottom picture was taken at 9:05 p.m. The differences in lighting and coloring between these two frames is quite clear.

I propose a cheaper way to detect model drift which makes use of cheap frame filtering techniques to determine when the new video frames have drifted too far from the training data. This technique can be used in place of a full YOLOv3 model which will result in dramatic computational savings.

2. Related Work

2.1. Drift Detection

Much of the current work in drift detection involves online systems that receive feedback on their predictions [3, 4]. This feedback allows the system to cheaply track the accuracy of the model over time. The systems typically define two thresholds, a warning threshold and a drift threshold. When the accuracy degradation passes the warning threshold, the system starts collecting data which will be used to retrain the model. Once the accuracy degradation passes the drift threshold, the model is retraining offline and then redeployed.

Tiny YOLO Accuracy Values (mAP)	
Dataset	**IOU 0.5**
1530 (training)	58.11%
1900	54.14%
2000	46.44%
2105	50.35%
2115	42.46%
2200	38.27%
2300	14.31%

Figure 2: mAP accuracy values for a specialized Tiny YOLO model. The model was trained on data from 3:30 p.m. and tested on a range of data throughout the late afternoon and evening.

While these systems are useful, they are not necessarily applicable in situations where immediate feedback is not available. In order to mimic these systems for video analytics, a full YOLOv3 model would need to be run periodically to obtain accuracy measurements. As stated previously, this is computationally expensive. This paper seeks to find a solution that does not require a full YOLOv3 model to be run on the new video data.

2.2. Frame Filtering

Reducto [6] is a system designed to predict changes in query results based on cheap frame differencing techniques that can be run on edge devices. Reducto was able to find a very high correlation between particular low-cost frame differencing scores and changes in query results, enabling cost-effective frame filtering on edge devices.

While this paper and Reducto are tackling different problems, the lessons learned from Reducto can be applied effectively to this problem. Specifically, this paper looks to find low cost frame differencing techniques that correlate with accuracy degradation in specialized neural network models.

3. Motivation Results

3.1. Setup

This paper attempts to correlate low-cost frame differencing scores with accuracy degradation in a specialized Tiny YOLO model. To accomplish this, I trained a Tiny YOLO model with data from the Jackson Hole Town Square dataset [5]. I used Darknet [7] to train the Tiny YOLO model to detect and classify specifically cars within the video frames. The model was trained with data from a 30-minute time frame starting at 3:30 p.m., and then tested on data from different periods throughout the day.

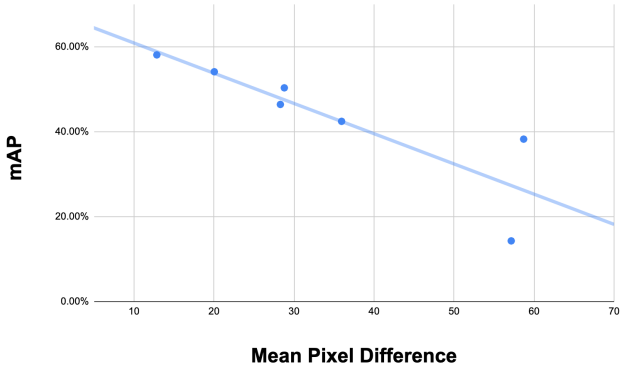

Figure 3: A graph plotting mAP scores against frame differencing scores for different periods throughout the day. The frame differencing technique was mean pixel difference applied to the top half of each image.

3.2. Results

The first step was to test the accuracy of the Tiny YOLO model for different period throughout the day to see how the accuracy degraded. The results for this test are included in figure 2. The model achieves an accuracy of 58 percent on the data it was trained with. When tested on periods throughout the data, the accuracy degrades almost linearly with the increase in time. The only exception is 8:00 p.m. which has a particularly low accuracy.

The next step was to find a cheap frame differencing technique that has a high correlation with the accuracy degradation of the model. I tried many different frame differencing techniques including frame subtraction (both color and grayscale), histogram differencing (chi square, intersection, and Bhattacharyya), and SIFT feature detection. However, the most effective frame differencing technique was frame subtraction applied to only the top half of the frame. The intuition for this is that the top half of the frame was primarily sky, so it would serve as a good predictor for lighting and coloring changes.

In order to obtain a frame differencing score, I sample a random 10% of frames from the training data and a random 10% of frames from the testing data. I then apply the frame differencing technique across each pair of sampled training and testing frames. The score for each pair is then averaged into one overall differencing score.

The correlation results for frame subtraction over the top half of the image is shown in figure 3 above. As the mean pixel difference increases, the accuracy generally decreases. The Pearson correlation coefficient between the mAP accuracy scores and the mean pixel difference scores is a respectable -0.8615. Excluding the two data points with the highest mean pixel difference, the Pearson correlation coefficient increases to -0.9666. Therefore, this particular frame

differencing technique is incredibly effective for predicting fine grained changes in accuracy of the model. While there is less predictive correlation with large changes in accuracy, the frame differencing technique can still provide clues that a retrain might be necessary.

4. Challenges

There are multiple challenges that need to be addressed before this system can be deployed effectively. The three primary challenges are the following.

3.1. Most effective frame differencing technique.

Our results proved that frame differencing can be an effective predictor for model drift. However, it is unclear how this particular frame differencing will generalize to other datasets. Therefore, it will be necessary to do more testing to determine if there is a cheap frame differencing technique that can generalize to multiple video feeds. It may also be possible that the best frame differencing technique will be different for each unique video feed. If this is the case, then a system will have to be developed that can dynamically choose the best frame differencing technique for a particular video feed.

Another issue is that these techniques may become less effective over time. This technique was motivated by showing that the model will degrade in accuracy over a single day, but it is unclear whether the technique will be as effective over a larger time period, such as a year. This will need to be evaluated further before this system is able to be deployed.

3.2. Choosing retraining thresholds.

This system will need to define a warning threshold and a drift threshold. Like the systems discussed in related work, the warning threshold will trigger data collection while the drift threshold will trigger an offline retraining of the specialized model.

Choosing the drift thresholds is not necessarily a trivial process. The thresholds will need to be based on a user defined accuracy goal. This accuracy goal will then have to be translated into a frame differencing threshold. Like the frame differencing technique, it is possible that the proper thresholds will differ for each video feed which will have to be addressed.

3.3. New data / old data for retraining.

Another issue will be how to choose data for retraining. In most systems, it will probably be appropriate to simply add the new data to the training data and retrain based on this dataset. However, it is also possible that the model has drifted so completely, that the initial training data is no longer applicable at all. If this is the case, there needs to be a policy that gradually phases out old training data and replaces it with more recent and more relevant training data.

5. References

[1] D. Kang, J. Emmons, F. Abuzaid, P. Bailis, and M. Zaharia. NoScope: Optimizing deep CNN-based queries over video streams at scale. *PVLDB,* 2017.

[2] Daniel Kang, Peter Bailis, and Matei Zaharia. Blazeit: Fast exploratory video queries using neural networks. arXiv preprint arXiv:1805.01046, 2018.

[3] J. Gama, P. Medas, G. Castillo, and P. Rodrigues, "Learning with drift detection," in Advances in Artificial Intelligence–SBIA 2004. Springer, 2004, pp. 286–295.

[4] M. Baena-Garcia, J. del Campo-Avila, R. Fidalgo, A. Bifet, R. Gavalda, and R. Morales-Bueno, "Early drift detection method," 2006.

[5] Jackson Hole Wyoming USA Town Square Live Cam. https://www.youtube.com/watch?v=1EiC9bvVGnk.

[6] Reducto.

[7] Joseph Redmon. Darknet: Open Source Neural Networks in C. http://pjreddie.com/darknet/.