Weather Detection Type and Time of Day on Video from Car's DVR

1st Edwin G. Carreño Lozano
Information Security Systems Department
Kazan National Research Technical University
named after A.N.Tupolev-KAI
Kazan, Russia
KarrenoLE@stud.kai.ru

3rd Nikita S. Rerikh
Information Security Systems Department
Kazan National Research Technical University
named after A.N.Tupolev-KAI
Kazan, Russia
rerikhns@gmail.com

2nd Rail R. Shakurov
Information Security Systems Department
Kazan National Research Technical University
named after A.N.Tupolev-KAI
Kazan, Russia
shakurov-98@mail.ru

4th Nadezhda E. Sitdikova
Information Security Systems Department
Kazan National Research Technical University
named after A.N.Tupolev-KAI
Kazan, Russia
nadyasitdikova@yandex.ru

***Abstract*—Recognition tasks are intensive areas under research, specially in Driver-less car applications or Unmanned Aerial Vehicles operations in order to guarantee security, reliability, and efficiency. One of those important recognition tasks is related to weather and time of the day recognition. Present work aims to walk through some famous Convolutional Neural Network architectures(ResNet, Inception, Xception, etc.) in order to build a raw classifier with an accuracy greater than 80% for four categories (DayCloudy, DayRainy, NightClear, NightRainy).**

***Index Terms*—convolutional neural networks, computer vision, digital video recording, tensorflow, weather recognition.**

I. Introduction

This work, it was proposed as an initial starting point from scratch into weather and time of the day recognition in digital video recordings, provides a review in the *State-of-art* of recognition tasks using Convolutional Neural Networks and different architectural approaches to resolve the problem of infer information in images. Contrary to those works mentioned ahead, the main idea for lectures is to find a practical approach that connect basics in Deep Learning with real problem without the urgency of putting them in production instantly. Several improvements are needed to find the best of the best of the model that could figure initial problem out with outstanding results, however in this work those improvements were applied but at the end of this document are discussed for future works in this actively area of Computer Vision.

II. State-of-the-art

Convolutional Neural Network has proven to be useful for computer vision tasks related to image classification, object detection, segmentation, etc. Despite the variety of datasets used for resolving those tasks, most of them do not explode more complex scenarios for self-driving applications such as weather detection, road structure identification, places, etc. In [1] not only has been proposed the DrivingScene dataset but also a multi-label architecture based on GoogleNet and Adaboost algorithm to improve the classification ability. Moreover, the join between the proposed dataset and multi-label architecture has achieved greater performance compared with previous related works. Another approach for resolving the task of weather detection has been proposed [2], specifically for traffic management case uses. Deep Supervised Convolutional Neural Network has been called the method to address the task of identifying weather conditions in surveillance systems. In this work, a dataset has been created with more than 2500 images to classify weather conditions in five categories (sunny, overcast, rainy, snowy, and foggy). At the same time, it was compared the performance of various classification algorithms such as Random forest, VGGnet-19, Inception-V3, Resnet-101, Densenet-161, and DS-CNN, measuring typical classification metrics such as precision rate, recall rate, and f-measure, as a result, the best score was achieved by the DS-CNN architecture model. In order to understand why CNNs are a useful approach to identify weather conditions compared to other types of pattern recognition models, it could be found in [3]. This work presents a concise guide and explanation of the use of the ImageNet dataset and a CNN architecture based on the work of Krizhevsky et al [4] as an evolution of the well-known Yann LeCun's architecture. Despite classifying the weather conditions in two classes (rainy, sunny), outstanding results were achieved with metrics such as Accuracy and Normal Accuracy. Their results have been between 85% and 93% respectively compared with previous related works with SVMs (Support Vector Machines), accuracy: 42%, and normal accuracy: 70.6%. Until now, previous works have not mentioned how to deal with weather detection when the

images or videos presented have been taken under the presence with day/night variations. Related work to weather conditions detection was presented in [5] to activate a windshield in the presence of rainy conditions. This work used a ResNet with 18 layers to identify conditions (rainy or not rainy), a dataset with an appropriate classification of images labeled in two main groups day and night with under different classes of rain. Videos were sampled to 30 frames per second but a sub-sampling was made until 1 frame per second to train the ResNet more rapidly. In this work was demonstrated that accuracy is higher if the dataset size increases; another metric used in this work was the precision/recall as is usual in classification tasks. Understanding a surrounding, foreground, or background context is important to achieve higher accuracy results in other tasks related to image/video classification. For instance, in [6] it is mentioned that the task of identifying cars under night conditions was hard due to overexposure of lights or missing cars in some regions with a lack of luminescence. Weather detection is a useful tool for different tasks because the weather condition has a strong effect on outside devices; operability as cameras. In [7] the authors describe how the weather influences autonomous vehicles' work correctness. They consider rain as an example of obstacles to present how it decreases an opportunity for object detection. A pedestrian is detected worse than a sedan because these objects have different radar cross-section (RCS) areas. Or if the rain intensity increases, these objects are less able to detect, and consequently, the sensing technologies may miss them. Simulations are made by one of the sensing technologies and by different scenarios. Also, the results of simulations show how important to determine the type of weather and its impact to set radar settings correctly. As cameras are used in contemporary vehicles, they have to work not only in good weather conditions but conditions of poor visibility too. Paper [8] presents a method for pedestrian detection in such conditions. The authors used the YOLO (You Only Look Once) approach for object detection that operates faster than its predecessors. The main feature of YOLO is CNN is applied for the whole image only once on the opposite other networks do it for separate regions of the image several times [9]. Based on this approach, the authors proposed three models that may be used during haze. They developed a weighted combination layer in one of the approaches by combining multi-scale feature maps and a squeeze and excitation block. As a result, the paper shows that proposed methods are more effective than existing pedestrian detection algorithms.

III. Motivation

As more analytical and make-decision technologies appear, and more they enter various business and social areas, as more their work should be accurate. For example, nowadays, vehicles are autonomous and can move without human participation. This became possible because of cameras and neural network usage, which detect objects on the road and, analyzing them, make decisions of the moving direction. Obviously, throughout the trip and under any conditions, the vehicle must do its job properly. Otherwise, it can lead to incorrect processing of the images received from the cameras and incorrect responses from the vehicle equipment. Similar technology is used not only in cars but in other various places. The picture can come from outdoor or indoor cameras, and the logic for processing this data can be very different but anyway, the conditions in which the camera operates must not influence its operating quality.

In this way, determining the time of day or weather in the picture could be used in these conditions in the problem-solving. A picture operating logic must consider additional restrictions and have instructions on how it must work for fog, snow, rain conditions, or at night. Thus, using a neural network to determine the time of day or weather conditions can be useful in many areas for various tasks and problems.

IV. System design

A. Prerequisites and Infrastructure

Before discussing any result, it is fundamental to indicate what are those tools, libraries and infrastructure used in this paper in order to replicate all the exposed work. In terms of infrastructure, I/O read operations and GPU (Graphical Processor Unit) show better performance in Kaggle instead Google Colab. In addition, training Convolutional Neural Networks require a big quantity of computing processing that a normal laptop could not do, for this reason Kaggle definitely has been the better choice for the purpose of this work.

In terms of software, we decant for using Keras as a top library to process Tensorflow objects. As a consequence of using Keras, all the architecture models revised here could handle big numerical arrays, distribute them between GPU cores and most importantly to create useful and completely functional Neural Networks with few lines of code. On the other hand, it is challenging training directly with videos as inputs for any model, for this part the FFMPEG software has been used for extract frames from input videos. Therefore, any model could use images instead videos and for this work it is sufficient. Finally, Python 3.8 has been used to program and integrate pre-processing, training, validation and testing data purposes as it will be explained in next section.

B. Pipeline Design

Figure 1 shows the basic plan to figure the problem out of identifying weather conditions and time of day taking into the account videos as inputs. Accordingly, with the Figure 1, the first step is to tag every single video in the complete dataset (90 videos in total). Using this dataset, tagging the videos could be possible indicating four different categories that at the same time (1 for weather and 1 for time of the day), those categories are: DayCloudy, DayRainy, NightClear and NightRainy. In addition, the selection of those categories were part of the process of evaluating how the length of the dataset limit the goal of this project. Following step was to convert those different videos into single image pieces in order to be processed, for this purpose was used the software FFMPEG

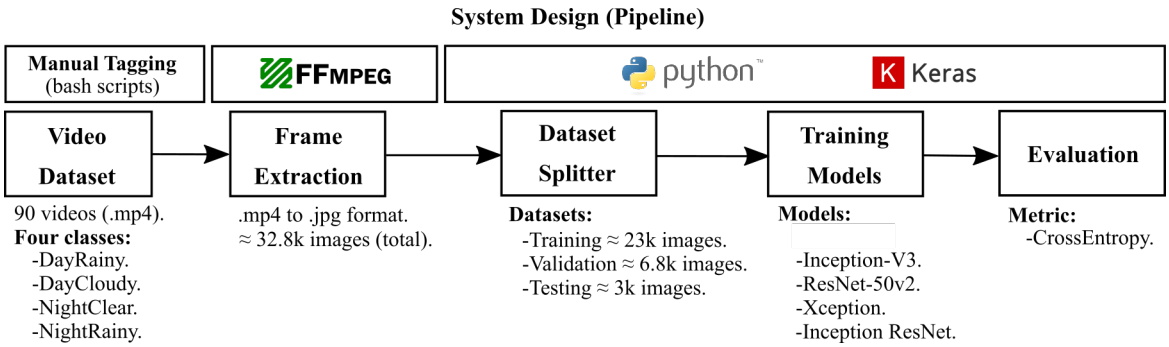

Fig. 1: System design pipeline for weather and time of day recognition in digital video recordings.

to extract frames with a result of approximately 32000 images in total for four categories.

However, the real core if this work is related to testing different Convolutional Neural Network Architectures to understand what could be an initial point to start with more robust applications. As a consequence, the idea of training, validating and testing over different slices of the complete dataset is still working in this approach. A portion of the dataset was used for training (70%) the different models, a remaining of 20% and 10% were used to validate (tuning hyper parameters) and report final results (best model and its accuracy). Finally, using Keras was possible to program and test complex CNN architectures using few lines of code, following the approach of describing layer, choosing optimizer, compiling the model and evaluating performance with built-in metrics such as Cross Entropy for classifiers with more than two categories.

V. Results

A. Comparison between CNN architectures

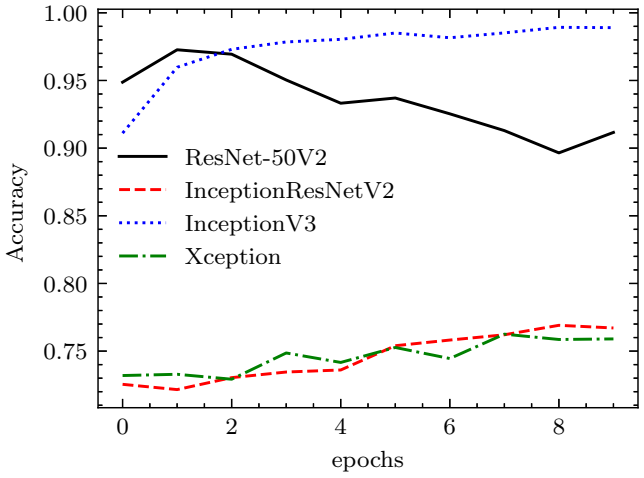

Fig. 2: Training accuracy results for different Convolutional Neural Network Architectures.

Table I summarizes the different results for every single architecture used to train the classifier (four categories) in this work. According to the table, best accuracy and lower loss was achieved by the *InceptionV3* architecture, reaching an accuracy of 98.90% and 2.28 units of loss. Figure 2 shows the accuracy evolution over every epoch used for training, note again, best results were achieved by the *InceptionV3* architecture pre-trained with the *ImageNet* dataset and deployed by Keras using their API(Application Program Interface). In addition, tuning hyperparameters is one of those activities out of the scope of this work, however, Figure 3 shows several peaks for accuracy and loss function, those are guidelines that the first tuning move it is related to adjust optimizer (Adam, SGD, etc.), learning rate and epsilon for the *InceptionV3* model.

TABLE I: Table Type Styles

Architecture	Train Acc.	Train Loss	Val. Acc.	Val. Loss
InceptionV3	98.90%	2.28	99.67%	2.32
ResNet-50V2	91.16%	6.34	97.74%	6.65
Xception	75.90%	6.67	76.35%	6.76
Inception ResNet	76.70%	5.31	76.87%	5.24

B. Evaluation and Testing results

Before going further predictions it is useful to refresh what is the loss function for classification purposes:

$$J(\mathbf{w}) = -\frac{1}{N}\sum_{i=1}^{N}[y_i \log(\hat{y}_i) + (1-y_i)\log(1-\hat{y}_i)]$$

Above formula in essence compare real tagged value (DayCloudy, DayRainy, NightClear, NightRainy) with the predicted values using an ordinal conversion of categories into numbers. At the end, if a predicted category mismatch the tagged value this loss function penalizes and occurs the opposite effect of minimization. Currently, those functions are implemented into the Keras API.

Figure 4 shows the final result of applying the Cross Entropy Sparse evaluation, it is the same loss function for classifiers, achieving 80.25% using the test dataset that previously was splitted from the complete dataset. Clarifying, achieving a high result as we did, it is not a evident our classifier will be work properly, however it is good starting point to experiment with more advanced topics such as hyperparameter tuning.

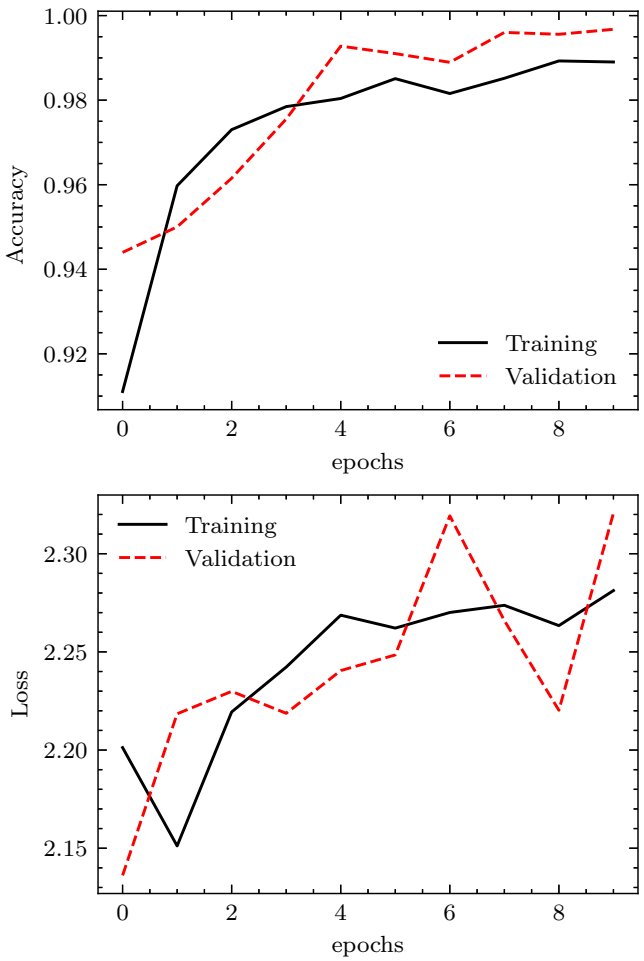

Fig. 3: Accuracy(top) and loss(bottom) for InceptionV3 architecture.

```
Model under evaluation: InceptionV3
48/48 [==============================] - 50s 1s/step - loss: 7.8016 - accuracy: 0.8026
        Test Accuracy:  0.802566647529602
```

Fig. 4: Test accuracy reported for *InceptionV3* using the Test dataset.

Fig. 5: Accuracy(top) and loss(bottom) for InceptionV3 architecture.

```
============================================================
=====        Plotting InceptionV3
============================================================

Path:  ../input/weatherdataset20200105-version3/Test/DayCloudy/DayCloudy_test00001.jpg
List predictions:  [[1. 0. 0. 0.]]
List after softmax:  0
Prediccion: DayCloudy

Path:  ../input/weatherdataset20200105-version3/Test/DayCloudy/DayCloudy_test00002.jpg
List predictions:  [[1. 0. 0. 0.]]
List after softmax:  0
Prediccion: DayCloudy
```

Fig. 6: Prediction of two images. Note, prediction coincide with the tag in the name.

In Figure 5 it is shown two different photos, correlated in time. Contrary to other recognition tasks, we could infer that for future works or deployed models in production, those models could apply sub-sampling in order to reduce computing complexity and having the same results. Finally, Figure 6 shows the result of predicting classes for the images in Figure 5, those results show our classifier is working as we expected in a first test.

C. Conclusions and future work

As a first approach, this work have reached remarkable results in two ways, what is the starting point model we should investigate and what are those following steps to improve it in order to increase performance. InceptionV3 has been model with best result (Val. accuracy 99.67% and Test Accuracy 80.25%) in our raw classifier, in order to improve accuracy and reduce loss, a future work should propose a new architecture based on elements that make this model different from others. On the other hand, optimizers play an important role for decreasing loss function to the lowest point possible avoiding meta-stability, divergence and most common case finding local minimum instead global minimum. Sweeping different optimizers, values for learning rate parameters and epsilon should be sufficient to calibrate the model in terms of optimization(finding minimum value for loss function). Classical approaches such as K-fold should be follow to reduce variance/bias. However, most importantly is the construction of a big dataset with different images equally distributed for each class, it does not matter if we have several videos, once frames are extracted those frames are highly correlated in time and it causes our model memorize data instead learn. Future work,

should use OpenCV as engine to recognize and predict in real time weather and time of the day, those implementations join with our raw classifier [10]. Finally, weather recognition does not require using each frame of a video in order to give and useful result, it is sufficient sub-sampling the total quantity of frames, contrary to other recognition tasks such as plate recognition.

REFERENCES

[1] L. Chen, W. Zhan, W. Tian, Y. He and Q. Zou. "Deep integration: a multi-label architecture for road scene recognition," In Proc. IEEE Transactions on Image Processing '10, 2019, pp. 4883-4898.

[2] Z. Sun, P. Wang, Y. Jin, J. Wang and L. Wang. "A practical weather detection method built in the surveillance system currently used to monitor the large-scale freeway in China," In Proc. IEEE Access, vol.8, 16 June 2020, pp. 112357- 112367.

[3] M. Elhoseiny, S. Huang and A. Elgammal, "Weather classification with deep convolutional neural networks," 2015 IEEE International Conference on Image Processing (ICIP), Quebec City, QC, 2015, pp. 3349-3353, doi: 10.1109/ICIP.2015.7351424.

[4] A. Krizhevsky, I. Sutskever, G. E Hinton, "Imagenet classification with deep convolutional neural networks," in Advances in neural information processing systems (NIPS), 2012, pp. 1097–1105.

[5] C. Lai and C. G. Li, "Video-Based Windshield Rain Detection and Wiper Control Using Holistic-View Deep Learning," 2019 IEEE 15th International Conference on Automation Science and Engineering (CASE), Vancouver, BC, Canada, 2019, pp. 1060-1065, doi: 10.1109/COASE.2019.8843331.

[6] C. Tsai, C. Tseng, H. Tang and J. Guo, "Vehicle Detection and Classification based on Deep Neural Network for Intelligent Transportation Applications," 2018 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC), Honolulu, HI, USA, 2018, pp. 1605-1608, doi: 10.23919/APSIPA.2018.8659542.

[7] S. Zang, M. Ding, D. Smith, P. Tyler, T. Rakotoarivelo, M. A. Kaafar, "The impact of adverse weather conditions on autonomous vehicles", IEEE Vehicular technology magazine, 2019, vol.14, pp 103-111.

[8] G. Li, Y. Yang, X. Qu, "Deep Learning Approaches on Pedestrian Detection in Hazy Weather", IEEE Transactions on Industrial Electronics, 2019, vol. 67, 8889 - 8899 pages

[9] J. Redmon, S. Divvala, R. Garshick, A. Farhadi, "You Only Look Once: Unified, Real-Time Object Detection", IEEE Conference on Computer Vision and Pattern Recognition, 2016, 779-788 pp.

[10] Carreño, E., Shakurov, R., Rerikh, N. and Sitdikova, N., 2021. "BNNs-Paper Project Notebook". [online] Kaggle.com. Available at: ¡https://www.kaggle.com/egcarren/bnns-paper-project-notebook/edit/run/50829801¿ [Accessed 28 January 2021].