ARIS: A Real Time Edge Computed Accident Risk Inference System

Pretom Roy Ovi*, Emon Dey*, Nirmalya Roy, Aryya Gangopadhyay
Center for Real-time Distributed Sensing and Autonomy (https://cards.umbc.edu)
Dept. of Information Systems, University of Maryland, Baltimore County, USA
{povi1, edey1, nroy, gangopad}@umbc.edu
*Authors have equal contribution

***Abstract*—To deploy an intelligent transport system in urban environment, an effective and real-time accident risk prediction method is required that can help maintain road safety, provide adequate level of medical assistance and transport in case of an emergency. Reducing traffic accidents is an important problem for increasing public safety, so accident analysis and prediction have been a subject of extensive research in recent time. Even if a traffic hazard occurs, a readily deployable structure with an accurate prediction of accident can contribute to better management of rescue resources. But the significant shortcomings of current studies are the use of small-scale datasets with minimal scope, being based on extensive data sets, and not being applicable for real-time purposes. To overcome these challenges, we propose *ARIS*: a system for real-time traffic accident prediction built on a traffic accident dataset named *'US-Accidents'* which covers 49 states of United States, collected from February 2016 to June 2020. Our approach is based on a deep neural network model that utilizes a variety of data characteristics, such as time-sensitive weather data, textual information, and discerning factors. We have tested *ARIS* against multiple baselines through a comprehensive series of experiments across several major cities of USA, and we have noticed significant improvement during inference especially in detecting accident classes. Additionally, to make our model edge-implementable we have compressed our model using a joint technique of magnitude-based weight pruning and model quantization. We have also demonstrated the inference results along with power consumption profiling after deploying the model on a resource constrained environment that consists of Intel Neural Compute Stick 2 (NCS2) with Raspberry Pi 4B (RPi4). Our investigation and observations indicate major improvements to predict unusual traffic accident event even after model compression and deployment. We have managed to reduce the model size and inference time by $\approx$ 6x, and $\approx$ 70 % respectively with insignificant drop in performance. Furthermore, to better understand the importance of each individual type of variables used in our analysis, we have showcased a comprehensive ablation study.**

***Index Terms*—accident risk prediction, real-time, quantization, pruning, device implementation**

I. Introduction

Traffic accident is one of the most dangerous threats to human life and statistically it has risen to the 8th leading cause of death globally. Almost 1.35 million people die in traffic accidents each year that stands for 3700 people everyday [1]. On average, there is a 3 percent of the gross domestic product loss due to traffic accidents in the majority of the countries around the world [2]. In fact even in US, despite having one of the best traffic infrastructures, traffic accidents are a serious problem. For example traffic accidents were responsible for the death of 23,708 vehicle occupants in 2017 [3], [4]. Early prediction of traffic hazards, one of the prerequisites in accident prevention, provides emergency responders with valuable information to determine the seriousness of injuries, assess the possible effects of the accident, and develop appropriate protocols for traffic system management.

At present, deep learning is attracting a great deal of attention from the research community owing to the flexibility it offers to address complex problems in text, image or even speech medium relating to a variety of applications including classification, recognition, forecasting etc. In a similar vein, the issue of traffic accident prediction can also have a deep learning based solution. To this extent, efforts to develop a comprehensive dataset for traffic hazard prediction started as soon as the concept of smart city development came into the limelight. For instance, some dataset of this kind for United Kingdom [5], Seattle [6], New York [7], Maryland [8] etc. are publicly available to help the research community come up with innovative technologies to reduce the severity and number of accidents as a whole. However, almost all available datasets till now provide only a fraction of the information related to accident ferocity, location and environmental influence, road and weather conditions etc. In general these datasets fail to provide a comprehensive breakdown which encapsulates the whole scenario of a traffic hazard. So, most of the existing studies are based on small-scale datasets which covers only one state or city, and in relation to the studies where detailed data is available, the authors refrain from making them publicly available due to the sensitive nature of the information. To address these issues, our work is done on "US-Accidents" dataset, which was collected from Feb, 2016 to July, 2020 and covers accident details of 49 states in the US. This dataset is unique in the sense that it covers majority of the states in a country (US) and has almost all the possible geo-spatiotemporal features including a brief description of the incident to clearly illustrate the event.

Even after developing an efficient traffic hazard prediction model, it is important to test whether the model is implementable in real environments. Most deep learning models suffer from huge computation requirement and high inference time. These make it difficult for real-time deployment. Hence, generating edge-device friendly deep models, compressing

978-1-6654-1252-0/21/$31.00 ©2021 IEEE
DOI 10.1109/SMARTCOMP52413.2021.00027

the models to run within the limited resources of edge-devices is essential. Many previous studies proposed different strategies to overcome this limitation. For example, low rank factorization [9], distributed processing [10], weight pruning [11] based approaches were introduced for different kinds of applications like computer vision [12], human activity recognition [13], audio classification [14], video processing [15], etc. Also, compression of different kinds of deep models i.e., Convolutional Neural Network (CNN) [16], Long Short Term Memory (LSTM) [17], Generative Adversarial Network (GAN) [18], etc. are active research fields these days. Moreover, replicating those compressed models into resource constrained devices like Nvidia Jetson Nano [19], Intel Edison [20], Snapdragon processor [21], etc. are drawing attention of research communities of these respective fields.

The goal of this work is to develop a system that would raise awareness among the drivers in certain accident-prone areas particularly in bad weather or road conditions. We propose an Accident Risk Inference System (ARIS); where we have introduced a pruning, and quantization based joint compression scheme to solve the issues with tight-resource edge devices. The conventional models for this purpose are required to run on heavy computing devices because of their computational complexity. The use of model compression technique in our case enables traits like low-power consumption, faster inference, and less memory usage with insignificant drop in performance. We utilize the Intel Neural Compute Stick (NCS2) with Raspberry Pi 4B (RPi4) for our edge deployment. To the best of our knowledge, this is the first study to incorporate the features of model compression, and edge-implementation for a real-time accident prediction application. **Our main contributions are:**

- *Accident Risk Inference System (ARIS):* We propose ARIS, which uses a variety of data including text description of traffic events, weather events, discerning factors, and time to provide real-time traffic accident prediction for every 15 minute time interval. The model consists of LSTM and dense layers where we have introduced a mixed approach with magnitude based weight pruning, and quantization aware training to bring down the size of our model (approximately one-sixth) in order to make it deployable on low power edge devices without compromising accuracy.
- *Edge-devices implementation:* To showcase the adaptability of our system with real-time deployment, we prototype ARIS using Intel Neural Compute Stick 2 (NCS2); one of the tightest inference engines available with Raspberry Pi 4B (RPi4) as test bed. We have tested the pliability of our system in terms of inference time, memory requirement while running, and also provided necessary power profiling. Across all these measuring parameters, our system fulfills required qualities to be considered as a real-time implementable one.
- *Empirical evaluation against several baseline models and ablation study:* We have evaluated the performance of our implemented models and illustrate comparisons with relevant baseline models including both the statistical, and DNN to illustrate the validity of our approach. We have considered several widely accepted performance metrics i.e., Area Under the Curve (AUC), F1-score, etc. in addition to accuracy to corroborate the efficacy of our model with skewed dataset. Detailed ablation study on the importance of various attributes on prediction results are also provided.

II. Related Work

In this related work section we will talk about similar datasets that were analyzed for these kinds of tasks, different accident prediction approaches, and finally some review on the previously explored compression techniques.

A. Literature on accident prediction

To analyze this kind of dataset, several kinds of techniques were explored. Among them, machine/deep learning based approaches are dominant. To provide some examples, [22] used a SVM based analysis which takes only numerical input and does a binary classification. Attention based works are demonstrated by [23] which looks for the relevant data points to predict the traffic incidents. Computer vision based [24] techniques are also popular and they exploits the use of real time camera even LiDAR sensor. Use of deep models like autoencoder [25], DNN [26], LSTM [27] etc. are being popular now-a-days due to their capability of extracting the insights from data with different modality. Additionally, supervised machine learning algorithms, such as AdaBoost, Logistic Regression (LR), Naive Bayes (NB), and Random Forests (RF) are also implemented on traffic accident data [28]. The data used in this study was provided by the Office of Highway Safety Planning (OHSP) and contains information about road crashes occurred on Michigan during the period ranging from 2010 to 2016. In [29], the authors proposed three different network architectures based on a simple NN, CNN, and RNN models for prediction of traffic accidents. The 2009–2015 traffic accident data for the North–South Expressway (NSE), Malaysia were used in this study. The authors deals with the dataset of road traffic accidents that occurred in Seoul, Korea, 2017-2018 [30]. They plot the importance of variables using Random Forest and represents that Victim's car, Accident Type, Attacker's car, Block and Violation are significant variables. In the above works, the prediction is mainly the percentage of chance of accident occurrence; the specific cases are not taken into account. So, accurately predicting a possible severe accidental event can lead to efficient use of resources and can finally result in fulfilling the goal of sustainable development. Our plan is to combine the analytical results of both numeric and textual parts of the main dataset so that we can showcase a robust model for real-time accident prediction.

B. Model compression

As an attempt to bring the extraordinary characteristics of deep learning like extracting features without domain knowledge, capturing complex structure of dataset, accurate

prediction, etc. into edge, the attempts are going on among researchers for a while. Due to insufficient resources to host a deep model in its original extent, it is imperative to find a way to reduce the computation complex, and model size while keeping the performance as intact as possible. Several compression techniques have emerged in the recent years to make the deep models to be more usable and making the inference using a wearable device. For instance: strategically reducing some of the model's structure components i.e., weights, filters was successfully documented in some research works. It is called pruning [11] technique which search for most valuable nodes responsible for better performance, and eliminate the others through different ranking methods. But pruning with some common regularization methods like L1 or L2, requires more iterations to converge than general. Also, the pruning criteria are required to be set manually for different layers to achieve the auspicious result. Reducing the required bit-size to represent the model weights and parameters is another notable approach. This is named quantization [31] and this wording is modified based on the used bit size like, 2-bit representation is known as binarization [32], technique. One possible shortcoming of this binary thresholding methods is based on simple matrix approximations that may not capture the proper effects of this imposed thresholding on the achievable accuracy loss. In low-rank matrix factorization [9], the parameters of one layer were set and the layers above were fine-tuned based on a criterion for reconstruction error. Several different approaches in this domain are undertaken to furnish the procedure, for example, truncated Singular Value Decomposition (SVD) [33], Batch Normalization (BN) based decomposition [34], etc. One common disadvantage of these approaches is that structural constraints may introduce bias which can degrade the model's performance.

In ARIS model, we have utilized a combined scheme of pruning, and quantization aware training. This approach helps us to apply two separate compression strategies simultaneously checking the performance curve which in terms aids us to select the most efficient structure.

III. Dataset Description

For our work we choose to use **'US-Accidents'** dataset [35] which is kind of self-explaining, and covers accident data of 49 states of the United States. The data is continuously being collected from February 2016, using several data providers, including two APIs which provide streaming traffic event data. These APIs broadcast traffic events captured by a variety of entities, such as the US and state departments of transportation, law enforcement agencies, traffic cameras, and traffic sensors within the road-networks. Currently, there are about 3.5 million accident records in this dataset. In our knowledge, it is the first dataset containing these comprehensive amount of information on accidents of US. It has 49 variables in total and some of the important variables are source of accident report, accident description, roundabout zone, no exit zone, junction, railway crossing in nearby location, temperature, wind flow, wind speed, wind direction, air pressure, weather condition, etc.

A. Feature Selection

In order to better utilize the features listed in our chosen dataset, we have categorized them as time-variant features and time-invariant features in a broad manner. We have tagged time, and weather as time-variant features whereas the natural language description, and discerning factors (e.g. stop sign, junction) were categorized as time-invariant features. We will now briefly describe those components.

- *Time-Variant Features*

  Time and Weather: Time indicates the period of the day such as weekday/weekend, hour-of-day and daytime or night. On the other hand, weather is mainly a vector representing 10 weather attributes including temperature, pressure, humidity, visibility, wind-speed, precipitation amount; and four special events such as rain, snow, fog, and hail.
- *Time-Invariant Features*

  Textual Description: This feature refers to the natural language description of the different traffic events. It portraits the road scenario in a definitive manner which we convert into a embedding vector of size 50.

  Discerning Factors: Discerning factors are nothing but the spatial characteristics of the roads. These factors are used to analyze the effect of amenity, speed bump, crossing, give-way sign, junction, no exit sign, railway, roundabout, station, stop sign, traffic calming, traffic signal, and turning loop on traffic condition. The vector size used to represent these factors is 13.

B. Data Preprocessing

With the help of the formulation stated above, we incorporate 15 time-variant (i.e., time, and weather) and 63 time-invariant (i.e., Discerning Factors and Description) attributes. In order to predict the label of incident (accident/non accident) during a given time, we use a vector representing the last 6 time intervals (last one and half hours), including one instance of time-invariant attributes (63 features), and 6 instances of time-variant attributes (6×15 features). In the original form of the dataset, the label distribution was among four categories. The authors of the dataset [35] have enlisted four difference severity levels. But we have experienced minute distinction among these four labels. So, in order to reform the dataset for our application, we have inspected the textual descriptions with respective severity levels reported in the dataset. We have found that severity level 1 and 2 stand for insignificant traffic events that results in short delay, these events are labeled as non-accident in our modified version of the dataset. Similarly, severity levels with 3 and 4 are categorized as accident events corroborated by their textual descriptions.

IV. Methodology

This section includes the detailed description of the process of developing ARIS model as a real-time. First, the major components of the base model will be covered, followed by the particulars of the employed compression technique.

A. Model Development

To make prediction with these heterogeneous types of data, we have developed a deep neural network consists of Multi Layer Perceptron (MLP) and Long Short Term Memory (LSTM) architecture. The model diagram can be found in figure 1. The major components of this illustration will be described subsequently.

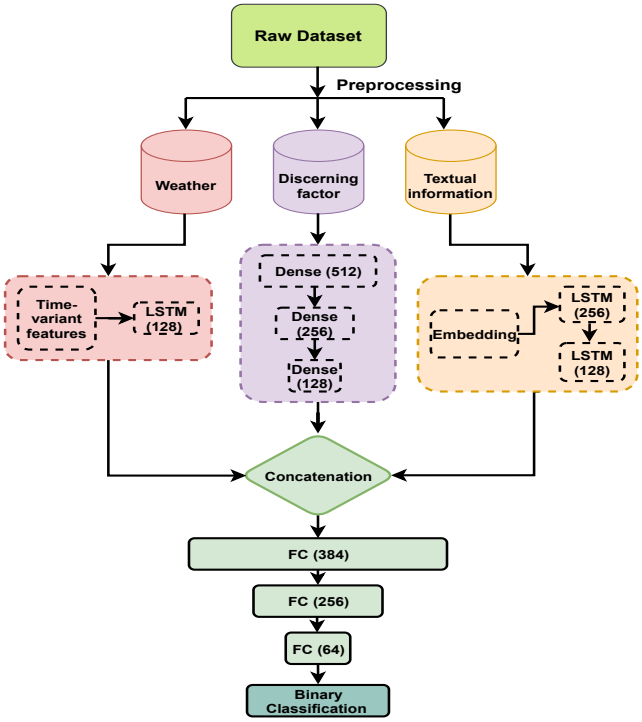


Fig. 1: Workflow of ARIS model.

- *Analyzing time-variant features*: Weather with specific time-stamps is considered as time variant feature. These attributes are fed to this network in their temporal order as they can be treated as a sequence. This component consists of a long-short-term-memory (LSTM) model with 128 LSTM cells. Thus, the output is a vector of size 128.
- *Analyzing discerning factors:* This component utilizes the spatial characteristics of the roads. The vector size here is 13 as par prior description . This vector is also processed with three feed-forward layer of size 512, 256 and 128 respectively. Thus, the output is a vector of size 128. 'ReLU' was used as the activation function in the first two dense layers and 'Sigmoid' was used on the last layer.
- *Analyzing textual description:* This part of the model converts the natural language description of an event into a embedding vector size of 50. This vector is then fed to two LSTM layers with size of 256 and 128 respectively. Thus, the output is a vector of size 128.
- *Concatenation and classification:* After gaining extracted features from three different components, a concatenation operation is done in order to utilize the captured inherent structure of data from multiple types. The output will go through the fully-connected component to make the final prediction. We have used four dense layers of size 384, 256, 64, and 1, respectively. 'ReLU' was used as activation function in the first three danse layers,and 'Sigmoid' was used in the output layer (last layer).

B. Compression

As shown in figure 1, our ARIS model contains several neural network layers i.e., Long Short Term Memory (LSTM), Fully Connected (FC) layers. So, in order to reform our model into a less computationally complex one, and with lesser parameters, we can exploit the features of model compression into these neural layers. In ARIS, we have used a combined compression scheme consists of magnitude-based weight pruning, and quantization aware training. In magnitude-based weight pruning, unlike dropout, we have the control over selecting the important nodes because we first let the model to reach to certain accuracy before setting a percentage of weights to zero. It also enables us to look into accuracy of the model after discarding a certain portion of connections. We have used two hyperparameters named sparsity, and scheduling. With sparsity, we can determine what percentage of nodes we want to keep, in our case we have used 60% sparsity. Scheduling helps us in setting up the interval between two subsequent pruning operations.

In addition to pruning, we have used quantization aware training. There are two popular methods for quantization: post training quantization, and quantization aware training. Quantization aware training ensures that forward pass matches the precision for both training and inference time. It is possible to generate the quantize awareness for the entire model or only parts of it. On the other hand, in post-training quantization, weights are quantized post-training, and the activations are quantized dynamically at inference time. As using quantization aware training gives a chance to look into the training performance directly while applying quantization, we have chosen this one. Quantization aware training emulates inference-time quantization which, through downstream tools, can be used to produce actually quantized models. After going through extensive empirical analysis to select suitable bit size for quantization, we find that 16-bit quantized model works best for our system. Also, we didn't apply quantization on the layer with data input and the last layer as it brings up the concerns of important raw information. It aids us to achieve more accurate results.

V. Experimental Setup and Evaluation

This section will walk you through the procedures followed to set up the experiments, and the evaluation results.

A. Experimental Setup

We have demonstrated our results for six major cities from four different time zones, they are: Atlanta, Seattle, Detroit, Miami, Denver, and Chicago. For each of them, each entry of

TABLE I: Specifications of Intel NCS2 with RPi4 as host board.

Metrics	Specifications
Processor	Intel Movidius Myriad X VPU with 16 SHAVE cores (128-bit VLIW Vector Processors)
Operating System	Raspbian Buster
Host Interface	USB 3.0 Type-A port
Memory	2 GB LPDDR4 with 512 MB LPDDR4 + 2.5 MB centralized on chip
CPU	Quad-Core Cortex A 72

data is represented by 63 time-invariant and 6×15 time-variant features.

For our experiment, we have chosen Logistic Regression, Decision Tree, Random Forest, and Deep Neural Network (four layers) as baseline models. These baseline models are implemented using Scikit-learn library and our main deep learning based ARIS model is implemented on Keras running on Tensorflow backend.

The model compression segment of ARIS model is carried out using the TensorFlow framework. We have modified the TensorFlow Runtime package according to our input types and application scenario. For pruning, we have chosen 'Magnitude based Weight Pruning' technique and tweaked two hyperparameters named 'Sparsity', and 'Scheduling'. For quantization bit selection, we have achieved the best results with 16 bits. The final model is saved into '.tflite' format to load and perform the inference procedure in edge device.

B. Device configuration and setting

To evaluate our model's compatibility with resource constrained environment, we have showcased the deployment results on a low power inference engine named 'Intel Neural Compute Stick 2 (NCS2) with Raspberry Pi 4B (RPi4) as carrier board. Intel has made the latest Movidius deep learning System-on-Chip (SoC) to be used for test bed designing. This chip is adequate in the sense that it enables both the parallel operation, and inference. It's processing speed can be brought up to 4 Tera Operations per Second (TOPs) within a power consumption of 1.5-watt. This is mainly specialized with Visual Processing Unit (VPU) and can be used with two different RAM settings: without additional RAM in-package and with 4 Gbits (512 MBytes) in-package RAM. The host interface of the hardware version of this package (NCS2) is a USB 3.0 dongle that houses the Movidius Myriad X VPU with 4 Gbit of RAM. This USB stick can be used as an add-onto boost the inference speed. Once the model is trained, the optimization technique can be modified using the OpenVINO framework before performing the inference operation. As a host device of this inference engine, we have chosen Raspberry Pi 4B (RPi4). The floating-point number for the input tensor is converted into FP16 as it is required for compatibility issue. The whole device implementation setup is illustrated in figure 2, and the configuration of this aforementioned edge composition is detailed in table I.

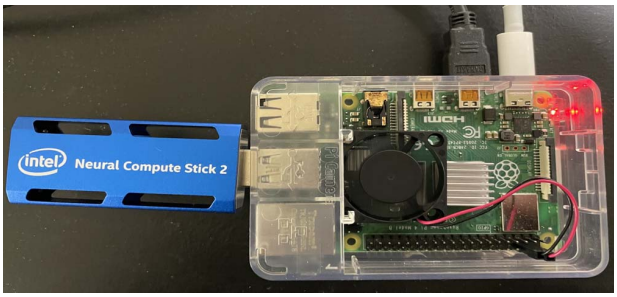


Fig. 2: Intel NCS2 with Raspberry Pi 4B as host board.

C. Evaluation

It is worth mentioning that the dataset used in our work is highly skewed, and may predict better for non-accident class. So, to deal with this issue, we have imposed class weights, and gone through stratified cross-validation approach. We have achieved significant improvement after adopting those strategies. We have evaluated the performance of the implemented models based on AUC (Area Under Curve) score, Precision, Recall, and F1-score reported for each class separately.

For AUC score the area under the curve is computed using the trapezoidal rule for each class (accident and non-accident). We specifically use AUC score since, the dataset we work with is skewed with respect to the distribution of samples and this metric generates a more holistic insight regarding the efficacy of our model. The respective ROC curves for each cities can be found in figure 3. Analyzing this figure it is evident that, AUC region for our ARIS model is always higher for all the six cities when compared to other baseline models. Also, the highest AUC score is achieved for the city named 'Seattle' and 'Denver' whereas the AUC scores for all cities range from 0.77 to 0.91 for our proposed ARIS model.

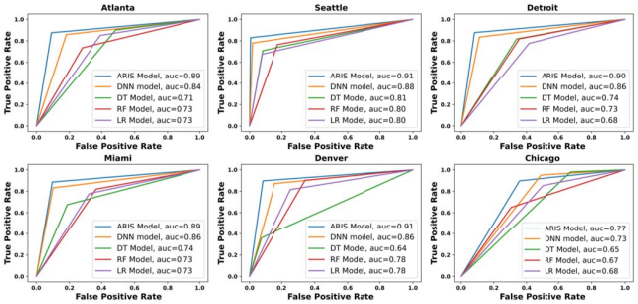


Fig. 3: Comparison of ARIS model with implemented baselines in terms of ROC curves for both accident and non-accident classes.

Also, the respective precision, recall and F1-score for each city for accident class and non-accident class is plotted in figure 4 and figure 5 respectively. The comparison of our model with the five baselines is also shown in Figure 4 and 5. Clearly, the ARIS model is performing better in almost all the cases. Analyzing the F1-scores for two classes it is expected that the results for non-accident class should be better than

the accident class. This is because the majority of data points belongs to non-accident class. One interesting thing to say, for Chicago city, our model's performances (Precision) degrades for non-accident class compared to the baseline models. But as described earlier, it is important to minimize type II error that means reducing the error of predicting accident event as non-accident event. In practical terms, the consequence of this error is the most severe for this predictive model. By looking into the results, we can claim that, our model successfully meets this requirement by showing superior results for both accident and non accident class.

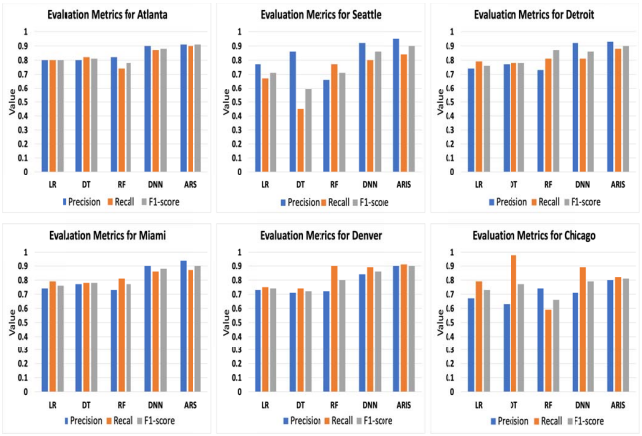


Fig. 4: Comparison between the ARIS model and baseline models based on Precision, Recall, and F1-score for Accident Class.

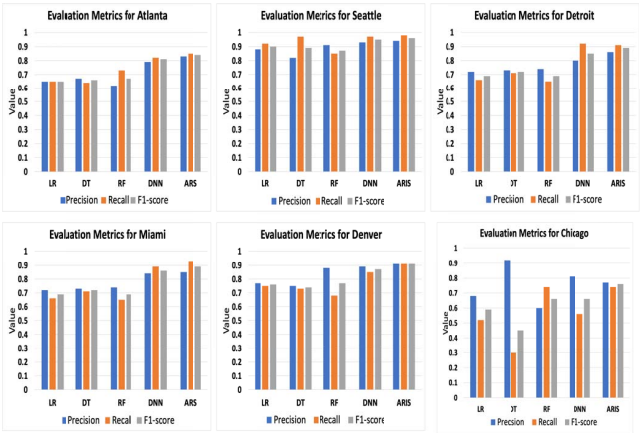


Fig. 5: Comparison between the ARIS model and baseline models based on Precision, Recall, and F1-score for Non Accident Class.

We have tabulated a compilation of all the results from the six cities taken into account (F1-Score) for both baseline machine learning and deep learning models along with our proposed ARIS model in table II. On average our model shows a 23.9% improvement for non-accident class and 18.6% improvement for accident class when compared to traditional machine learning models. Finally, when compared to deep learning frameworks, our model demonstrates 6% and 4.7% F1-score improvement for non-accident and accident class respectively which further justifies the effectiveness of our optimized model.

D. Ablation Study

To better understand the importance of each individual type of variable used in our analysis, we have conducted an extensive ablation study. This enlightens us with the effect of a certain variable on the output. This part is divided into two segments. First we analyzed the model performance for both accident and non accident class by excluding one variable each time. The resultant graphs of these experiments for the ARIS model are shown in figure 6a. This plot signifies that, removing text description from the model may harm the performance in broader magnitude for both the accident and non accident classes. The possible reason for that is using the textual description the model can better grasp the traffic conditions. On average, the combination of weather, and textual description provides better results for both accident and non-accident classes. Minor deviations can be noticed in case of Seattle, and Miami for accident, and non-accident classes respectively.

Secondly, for cross-verification with the previous settings, we have implemented ARIS model with only one variable at a time. This actually correlates with the previous results with exclusion of one variable. Using either text description of traffic event or weather attribute results in satisfactory results as these two variables go hand in hand, whereas using only discerning factor results in severe degradation on performance. The respective plots can be found in figure 6b.

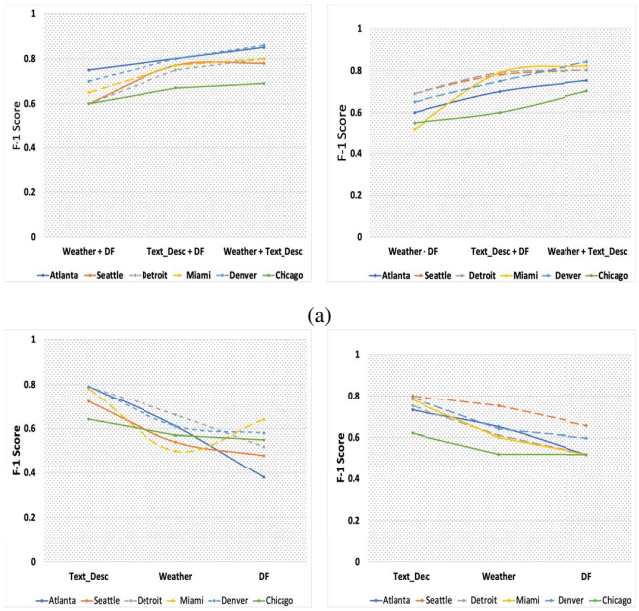
VI. Edge-device implementation

To make this system real-time, it is imperative to make the inference stage implementable on edge devices. We envision that this can be introduced in the form of an app on drivers' mobile phone or smartwatch, a new dashboard feature in vehicles or can be established as overhead billboards, containing a resource constrained processor itself, showing prediction ahead of time. We hypothesize that the ARIS if implemented on the cloud compared to the edge can incur delay in prediction by sending data to the cloud, performing computing there, and finally sending back the information to the drivers while rendering it to a local display.

The performance evaluation of *ARIS* model after implementing it on NCS2, we have considered three significant metrics namely: inference time, required memory while model running, and power usage during prediction. The inference time or running time validates the real-time capability of any model. In our case, this inference time includes data loading time, model loading time, and final result displaying time. We have used the 'time' function of Raspbian OS to measure the inference time. For the reported inference time here, we have considered batch size of 1 that means time required

TABLE II: Result compilation of all implemented models based on F1-score for both accident and non-accident classes.

City Name	LR		DT		RF		DNN		ARIS	
	Non Accident	Accident	Non Accident	Accident	Non Accident	Accident	Non Accident	Accident	Non Accident	Accident
Atlanta	0.65	0.8	0.66	0.81	0.67	0.78	0.81	0.88	**0.84**	**0.91**
Seattle	0.9	0.71	0.89	0.59	0.87	0.71	0.95	0.86	**0.96**	**0.9**
Detroit	0.69	0.76	0.72	0.78	0.69	0.87	0.85	0.86	**0.89**	**0.9**
Miami	0.69	0.76	0.72	0.78	0.69	0.77	0.86	0.88	**0.89**	**0.9**
Denver	0.76	0.74	0.74	0.72	0.77	0.8	0.87	0.86	**0.91**	**0.9**
Chicago	0.59	0.73	0.45	0.77	0.66	0.66	0.66	0.79	**0.76**	**0.81**



(a)

(b)

Fig. 6: (a) Output of ARIS model using all excluding one variable (The leftmost plot is for accident class, the rightmost one is for non-accident), and (b) Output of ARIS model using only one variable at a time (The leftmost plot is for accident class, the rightmost one is for non-accident).

to process individual data point. Another important metric for limited resource model evaluation is the memory size allocation while running the inference operation. The peak memory required during inference should be within the reach of resource constrained setup. We have defined the difference between the idle memory value, and highest value of memory while inference procedure operation as required memory in our work. The results are shown in Megabytes (MB). The power profiling of any model is also one vital factor while we proceed for real world deployment. The running power of any deep model should be well within the sustainable limit of the device capability. Here, the reported value for power consumption is achieved by subtracting the idle power value from the peak power value displayed during inference operation. The unit used for reporting is Watt (W), and we have used a USB power monitor for this purpose. The achieved values in terms of these three metrics and comparison among base, pruned, and final pruned with quantization can be found in figure 7. For the sake of edge-device implementation, we optimized our software framework and compressed the size of our base model to one-fifth. At first, we performed pruning on our base model, then did quantization on the pruned model and achieved reasonable model size. Besides achieving compressed size, our pruned and quantized tflite model takes less time, lower inference power and less memory to execute on the edge devices. Noticeably, the inference time of the final pruned and quantized model is less than 1.5 mS which suggests the requisite harmony for a real-time implementable model. The results depicted in the figure 7 clearly indicate that we gained improvement of 1.9 times on running time, 1.8 times on power consumption, 2.8 times on memory use in RPi4 device to execute the inference.

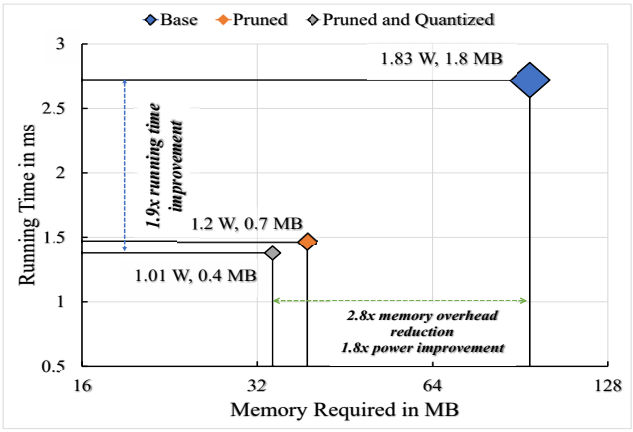


Fig. 7: Comprehensive illustrative comparison among ARIS model (pruned and quantized), pruned only, and uncompressed base model with respect to running time, memory required, and power consumption during inference. The descending shape of legends from base to pruned and quantized model indicates their relative difference based on model size.

VII. Conclusion

In this work, we have presented an edge implementable deep learning-based technique to model the real-time traffic accident prediction. We have used different data modality like numerical and textual description to make our model more prevalent. We have also compared our ARIS model with four of the baseline models (Logistic Regression, Decision Tree, Random Forest, and Deep Neural Network) and experimental results demonstrate that ARIS model outperforms them in almost all the cases for accident class prediction and also minimizes false negative rate significantly. The documented ablation study in this work articulates the significance of each individual variable

for accident prediction framework. Furthermore, we have achieved noticeable improvement among some major metrics for resource-constrained device implementation. Especially, the achieved inference time of 1.38s, and low power requirement (1.01W) using our model without significant drop in prediction performance makes ARIS a suitable one to be implemented in real-time traffic environment. In future, we hope to develop a dedicated hardware accelerator for real-time traffic monitoring, and accident prediction purposes including the feature of secure communication between deployed accelerators for a comprehensive and more confident prediction. We hope that this research will pave the way of designing more interactive and efficient traffic structure which is one of the key features of smart city development.

ACKNOWLEDGMENT

This research is partially supported by NSF grant number 1923982, "HDR DSC: Collaborative Research: Creating and Integrating Data Science Corps to Improve the Quality of Life in Urban Areas".

REFERENCES

[1] "Casuality statistics of accidents in us," https://policyadvice.net/car-insurance/insights/how-many-people-die-in-car-accidents/., accessed: 2020-10-02.

[2] "Casuality statistics of accidents in us," http://www.who.int/news-room/fact-sheets/detail/road-traffic-injuries, accessed: Jul. 18, 2018.

[3] "Us accident statistics," https://www.statista.com/statistics/191521/traffic-related-fatalities-in-the-united-states-since-1975/, accessed: 2020-10-02.

[4] "Fatality statistics of accidents in us," https://www.statista.com/statistics/192097/number-of-vehicles-involved-in-traffic-crashes-in-the-us/, accessed: 2020-10-02.

[5] "Traffic accident data of uk." https://www.kaggle.com/daveianhickey/2000-16-traffic-flow-england-scotland-wales/., accessed: 2020-10-05.

[6] "Traffic accident data of seattle." https://data-seattlecitygis.opendata.arcgis.com/datasets/5b5c745e0f1f48e7a53acec63a0022ab_0/., accessed: 2020-10-05.

[7] "Traffic accident data of new tork." https://data.ny.gov/Transportation/Motor-Vehicle-Crashes-Vehicle-Information-Three-Ye/xe9x-a24f/., accessed: 2020-10-05.

[8] "Traffic accident data of maryland." https://catalog.data.gov/dataset/maryland-statewide-vehicle-crashes-cy2017-quarter-1-c6361/., accessed: 2020-10-05.

[9] C. Tai, T. Xiao, Y. Zhang, X. Wang *et al.*, "Convolutional neural networks with low-rank regularization," *arXiv preprint arXiv:1511.06067*, 2015.

[10] F. Tung and G. Mori, "Clip-q: Deep network compression learning by in-parallel pruning-quantization," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 7873–7882.

[11] S. Han, J. Pool, J. Tran, and W. Dally, "Learning both weights and connections for efficient neural network," in *Advances in neural information processing systems*, 2015, pp. 1135–1143.

[12] A. Mathur, N. D. Lane, S. Bhattacharya, A. Boran, C. Forlivesi, and F. Kawsar, "Deepeye: Resource efficient local execution of multiple deep vision models using wearable commodity hardware," in *Proceedings of the 15th Annual International Conference on Mobile Systems, Applications, and Services*. ACM, 2017, pp. 68–81.

[13] E. Dey and N. Roy, "Omad: On-device mental anomaly detection for substance and non-substance users," in *2020 IEEE 20th International Conference on Bioinformatics and Bioengineering (BIBE)*. IEEE, 2020, pp. 466–471.

[14] N. D. Lane, P. Georgiev, and L. Qendro, "Deepear: robust smartphone audio sensing in unconstrained acoustic environments using deep learning," in *Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing*. ACM, 2015, pp. 283–294.

[15] V. Kastrinaki, M. Zervakis, and K. Kalaitzakis, "A survey of video processing techniques for traffic applications," *Image and vision computing*, vol. 21, no. 4, pp. 359–381, 2003.

[16] S. Albawi, T. A. Mohammed, and S. Al-Zawi, "Understanding of a convolutional neural network," in *2017 International Conference on Engineering and Technology (ICET)*. Ieee, 2017, pp. 1–6.

[17] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[18] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Advances in neural information processing systems*, 2014, pp. 2672–2680.

[19] S. Bhattacharya and N. D. Lane, "From smart to deep: Robust activity recognition on smartwatches using deep learning," in *2016 IEEE International Conference on Pervasive Computing and Communication Workshops (PerCom Workshops)*. IEEE, 2016, pp. 1–6.

[20] ——, "Sparsification and separation of deep learning layers for constrained resource inference on wearables," in *Proceedings of the 14th ACM Conference on Embedded Network Sensor Systems CD-ROM*. ACM, 2016, pp. 176–189.

[21] N. D. Lane, Y. Xu, H. Lu, S. Hu, T. Choudhury, A. T. Campbell, and F. Zhao, "Enabling large-scale human activity inference on smartphones using community similarity networks (csn)," in *Proceedings of the 13th international conference on Ubiquitous computing*. ACM, 2011, pp. 355–364.

[22] D. Al-Dogom, N. Aburaed, M. Al-Saad, and S. Almansoori, "Spatio-temporal analysis and machine learning for traffic accidents prediction," in *2019 2nd International Conference on Signal Processing and Information Security (ICSPIS)*. IEEE, 2019, pp. 1–4.

[23] L. Zhu, T. Li, and S. Du, "Ta-stan: A deep spatial-temporal attention learning framework for regional traffic accident risk prediction," in *2019 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2019, pp. 1–8.

[24] E. P. Ijjina, D. Chand, S. Gupta, and K. Goutham, "Computer vision-based accident detection in traffic surveillance," in *2019 10th International Conference on Computing, Communication and Networking Technologies (ICCCNT)*. IEEE, 2019, pp. 1–6.

[25] D. Singh and C. K. Mohan, "Deep spatio-temporal representation for detection of road accidents using stacked autoencoder," *IEEE Transactions on Intelligent Transportation Systems*, vol. 20, no. 3, pp. 879–887, 2018.

[26] S. Moosavi, M. H. Samavatian, S. Parthasarathy, R. Teodorescu, and R. Ramnath, "Accident risk prediction based on heterogeneous sparse data: New dataset and insights," in *Proceedings of the 27th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, 2019, pp. 33–42.

[27] H. Ren, Y. Song, J. Wang, Y. Hu, and J. Lei, "A deep learning approach to the citywide traffic accident risk prediction," in *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*. IEEE, 2018, pp. 3346–3351.

[28] R. E. AlMamlook, K. M. Kwayu, M. R. Alkasisbeh, and A. A. Frefer, "Comparison of machine learning algorithms for predicting traffic accident severity," in *2019 IEEE Jordan International Joint Conference on Electrical Engineering and Information Technology (JEEIT)*. IEEE, 2019, pp. 272–276.

[29] M. I. Sameen, B. Pradhan, H. Shafri, and H. B. Hamid, "Applications of deep learning in severity prediction of traffic accidents," in *Global Civil Engineering Conference*. Springer, 2017, pp. 793–808.

[30] R. Garrido, A. Bastos, A. de Almeida, and J. P. Elvas, "Prediction of road accident severity using the ordered probit model," *Transportation Research Procedia*, vol. 3, pp. 214–223, 2014.

[31] J. Wu, C. Leng, Y. Wang, Q. Hu, and J. Cheng, "Quantized convolutional neural networks for mobile devices," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 4820–4828.

[32] M. Courbariaux, Y. Bengio, and J.-P. David, "Binaryconnect: Training deep neural networks with binary weights during propagations," *arXiv preprint arXiv:1511.00363*, 2015.

[33] Y. Lu, A. Kumar, S. Zhai, Y. Cheng, T. Javidi, and R. Feris, "Fully-adaptive feature sharing in multi-task networks with applications in person attribute classification," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 5334–5343.

[34] V. Lebedev, Y. Ganin, M. Rakhuba, I. Oseledets, and V. Lempitsky, "Speeding-up convolutional neural networks using fine-tuned cp-decomposition," *arXiv preprint arXiv:1412.6553*, 2014.

[35] S. Moosavi, M. H. Samavatian, S. Parthasarathy, and R. Ramnath, "A countrywide traffic accident dataset," *arXiv preprint arXiv:1906.05409*, 2019.