TrafficNNode: Low Power Vehicle Sensing Platform for Smart Cities

Justin Nguyen
Electrical and Computer Engineering
Carnegie Mellon University
Mountain View, CA USA
justin.nguyen@sv.cmu.edu

Reese Grimsley
Electrical and Computer Engineering
Carnegie Mellon University
Pittsburgh, PA USA
reeseg@cmu.edu

Bob Iannucci
Electrical and Computer Engineering
Carnegie Mellon University
Mountain View, CA USA
bob@sv.cmu.edu

***Abstract*—Automating traffic monitoring and management can materially contribute to the realization of Smart Cities, but this demands detailed, accurate and timely characterization of traffic flows. Current methods employ video capture (high installation and operating costs), pneumatic-tube-based counters (limited detail about vehicle types and often only installed temporarily), or manual data capture (high human cost).**

We have developed an intelligent, inexpensive, pavement-mountable device capable of collecting information about vehicle types and speeds using an embedded, power-optimized neural network. The device is designed to fit within a raised pavement marker (RPM). RPMs are deployed throughout the world as lane markers. Packaging our sensing technology into RPMs offers the potential to significantly improve the spatial and temporal resolution of traffic flow information across a city.

We outline our methodology for energy-optimized machine learning on a small, resource-constrained sensor device. We present the results of our work in terms of accuracy (96% classifying vehicle type and 89% classifying vehicle speed) and battery life (three years).

***Index Terms*—tinyML; embeddedML; traffic sensing; Smart Cities; IoT; RNN; Low-Power.**

I. Introduction

Smart Cities employ pervasive sensing networks that enable intelligence in city infrastructure, resource management, and policy to improve the well-being of the city and its inhabitants [1]. Roads are a critical component of city infrastructure, and applying concepts from smart cities to their improvement can yield substantial, widespread benefit.

Cities need fine-grained measurements, *i.e.* precise in space and time, of traffic volumes, the types of vehicles, and their speed [2] to better understand roadways and their use. Armed with this knowledge, signage, traffic signal re-timings and other interventions can be implemented to reduce the congestion and re-balance traffic flows. Such improvements benefit drivers, residents, business-owners, and policymakers [3].

To facilitate fine-grain data collection at a city-scale, tens of thousands of sensors (or more) will be needed [4]. Existing traffic monitoring solutions (video, inductive loop, pneumatic tube, etc. [2]) would incur high installation and operating costs to achieve the necessary level of detail. Suitable traffic sensors need to be inexpensive to manufacture, deploy, and maintain, and should last as long as the rest of the city's roadway system, *i.e.*, multiple years without constant attention.

A long-lasting, battery-powered solution is the most cost-effective one. Battery-powered platforms are cheaper to deploy because they avoid the cost of installing mains power. Low unit cost permits more sensors and higher spatial resolution. Battery-powered sensors with short lifetimes incur significant maintenance costs, especially when situated in the roadways, as technicians must visit thousands of devices to swap batteries, temporarily closing roadways in the process. Such sensors must last for years between required maintenance, just like a stop sign or a traffic light. This requires careful optimization of compute, network, and peripheral usage on the sensor. Given the power constraints, low-powered MEMS transducers like magnetometers are the most suitable sensing modalities.

In this work, we demonstrate that it is possible to extract vehicle speed and type from three-axis magnetometer sensing with embedded machine learning, all of which can be packaged in a compact, battery-operated device that can live on road surfaces for three to five years without maintenance. We present the somewhat counter-intuitive result that performing this computation in the device and only sending summary information wirelessly actually runs at lower average power than streaming sensed readings and performing the neural network computations at the edge or in the cloud.

In this work, we develop a gated recurrent unit (GRU) [5] classification model which can be deployed on a microcontroller. We evaluate this on our custom hardware (based on a commodity ARM Cortex-M4 processor) using vehicular data collected for this purpose. The entire sensor–the *TrafficNNode*–includes a low-power wide-area network radio and a suitable battery, and is designed to fit within a commodity Raised Pavement Marker (RPM). RPMs are widely deployed on roads world wide and are excellent vantage points for observing traffic flows. Our classification model is capable of counting vehicles classifying them by type and speed range with 96% and 89% accuracy, respectively. Our device has an estimated life time of up to three years for a 2 Ah $Li - SOCl_2$ battery.

Our contributions include:

- Feasibility analysis of a battery-powered Smart City traffic sensor using embedded machine learning.
- Domain-specific optimization techniques and methodologies to improve the battery life for embedded machine learning without sacrificing accuracy.

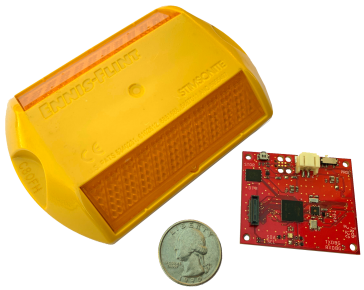

Fig. 1: CommonSense Low-Power Sensing Hardware

II. Related Work

Traffic measurement systems (TMS) for Smart Cities are the building blocks upon which we build our work. TMS's and their impact are discussed in [2], [6]. These works examine various measurement methods, the majority of which do not scale for fine-grained measurements since they rely on existing infrastructure or their sensing modalities are too power-hungry for cheap, battery-powered sensors.

Prior literature has identified magnetic flux as a useful modality for sensing vehicles given the large amount of ferromagnetic material present in automobiles [7], [8]. A vehicle speed estimation study using magnetoresistive sensors [8] implemented a graph-based approach to determine vehicle speed on an RPM-like wireless sensor. However, they are unable to determine the traffic composition, *i.e.,* vehicle types, and their system will only last one year on a 5 Ah battery. We build upon these studies with a neural-network based approach that was infeasible until recent advances in tinyML and modern microcontrollers.

The field of tinyML, *i.e.*, machine learning on deeply embedded devices, is relatively new. Tools like TensorFlow Lite for Microcontrollers (TFLM) [9] make the deployment of neural networks on microcontrollers computationally feasible by reducing the model size and instruction count without impacting accuracy. Several studies look at how neural network architecture can modified to improve energy efficiency, generally by reducing the computation time, which has a linear relationship with the total energy consumed [10]–[12]. This can be further optimized by leveraging application knowledge, which we describe in Section IV.

III. Sensor Hardware and Dataset

For our system, we target a custom hardware design, CommonSense, that is built for prototyping low-power IoT sensors with substantial compute capabilities. CommonSense, shown in Figure 1 uses a Cortex-M4F processor, a 3-axis magnetometer for sensing vehicles, and a LoRa radio for communication. Table I describes parameters of this platform with respect to computation and energy consumption.

We built our training and test datasets for classification of vehicle types and speeds across three separate events [13]. Drivers were recruited and test tracks were set up at two safe, off-highway locations. Several prototype TrafficNNodes were mounted to the pavement in the center of and offset from the centerline of travel. The vehicles were queued a few hundred meters from the sensors to give ample distance to stabilize at the target speed. A radar gun captured actual speed as each vehicle transited the sensor area. Each TrafficNNode continuously recorded three-axis magnetometer traces to capture each passing event. A camera recorded the radar gun and sensor area to ease future labeling; passing events were timestamped. This was repeated for each of the five speed classes for each of the eight vehicle six times identically on two separate days. The vehicles utilized consists of three SUVs, four sedans, and one police cruiser (a sedan that is heavier than all other vehicles in the dataset) for a total of eight vehicle types. In total, the dataset consists of 500 vehicle measurements, evenly split among the five speed classes and eight vehicle types, which we augment with AWGN noise, time-shifts, and amplitude scaling to increase the dataset size by a factor of 20 for a total of 10,000 measurements.

Processor Speed	120 MHz
RAM	256 kB
FLASH	1024 kB
Active Current	21.26 $\pm$ 0.072 mA
Sleep Current	35.8 $\pm$ 1.80 µA
Magnetometer Active Current (400 Hz)	575 µA
Magnetometer Sleep Current (12.5 Hz)	40µA
LoRa Transmit Power (SF10+14dBm)	33.5 mA

TABLE I: CommonSense Hardware Parameters

IV. Methodology

This section follows the step-by-step approach that we took to design and optimize the classification model for energy efficient deployment of TrafficNNode. We begin from the baseline model with no constraint on network size, computation time, or memory footprint. We implement optimizations to reduce power without sacrificing more than a few percent of recognition accuracy. We start with standard tinyML optimizations like reducing network size and sparsifying the model [12].Next, we perform application-specific optimizations, such as reducing the sample rate, to further reduce energy usage on a single sensor. Finally, we propose several network-level optimizations, which leverages load-balancing across several sensors to reduce energy usage even further.

A. *TrafficNNode Classification Model*

Our traffic-sensing model is built using Tensorflow and TFLM [9]. TFLM provides a portable runtime which reads in a compiled Tensorflow Graph binary. TFLM then constructs the graph and executes it at runtime.

The network is a many-to-one recurrent classifier where the recurrent neural network (RNN) is applied over the entire sequence of 3-axis magnetometer samples and the final hidden state is passed to a fully-connected layer with softmax activation to perform inference. We chose a gated-recurrent unit (GRU) [5] as the recurrent node based on accuracy vs. computation time; our initial RNN achieved 99.2% accuracy where the RNN hidden layer contains 18 neurons.

B. *Metrics*

To evaluate performance, we will use accuracy and computation time as our primary metrics. Computation time is a

direct proxy for energy usage because it has a direct linear correlation as shown by [11]. Intuitively, more time spent computing means more time spent in a high-powered active state *vs.* putting the device into an ultra-low power sleep state. Equations 1 and 2 translate the computation time into energy usage over the course of a day.

$$t_{comp/day} = \text{AADT} \times (Lt_{RNNstep} + t_{projection}) \quad (1)$$

$$E_{comp/day} = V_{batt} I_{comp} \frac{t_{comp/day}}{86400 \frac{s}{day}} \quad (2)$$

$$E_{sleep/day} = V_{batt} I_{sleep} (1 - \frac{t_{comp,perday}}{86400 \frac{s}{day}}) \quad (3)$$

where AADT is the annual average daily throughput (a metric describing vehicles per day), L is the sequence length and t_{layer} is the time in seconds to compute a single forward propagation for the $layer$ in the model. I_{comp} is the average measured current consumption of the processor when performing the inference and V_{batt} is the supply voltage from the battery, which we assume to be constant throughout the vast majority of the battery's life cycle. I_{comp} and I_{sleep} are shown in Table I. $E_{comp/day}$ and $E_{sleep/day}$ represent the amount of energy spend over the course of a day performing computation to classify passing vehicles and sleeping, respectively.

The batteries lifetime is represented as:

$$t_{lifetime} = \frac{E_{batt}}{(E_{comp/day} + E_{sleep/day})} \quad (4)$$

where $t_{lifetime}$ is in days and E_{batt} is the battery capacity in Watt-hours, although we assume stable 3.6V operating voltage for the battery rated in Ah. As we optimize the classification model with respect to energy per vehicle classification, we expect to see a trade off with accuracy.

C. Model Size Tuning

Our first energy optimization is a byproduct of ordinary network hyperparameter tuning, which is necessary regardless of our system constraints. Specifically, we tune the hyperparameter for the number of hidden units, *i.e.,* the size of the GRU cell's state vector. A high dimension hidden state may lead to over fitting, and a low dimension will incur under fitting. We prune the network size to reduce overfitting and minimize inference time. There is a linear relationship between the recurrent hidden dimension and time-complexity as shown in Figure 2, and thus, a linear relation between the number of hidden neurons and the energy cost of a single inference.

We empirically select the best value for the recurrent hidden dimension by sweeping this hyperparameter and analyzing the accuracy and computation time as shown in Figure 2. The accuracy drops as this hyperparameter decreases, but is nonlinear: we observe a 'knee' in this curve around $hiddendim = 6$ and 12, where the accuracy starts to drop more sharply. A $hiddendim = 8$ offers a good tradeoff of high accuracy and low computation time, although we could use a smaller dimension to improve the energy efficiency further at the loss of accuracy. The rest of the paper will use $hiddendim = 8$.

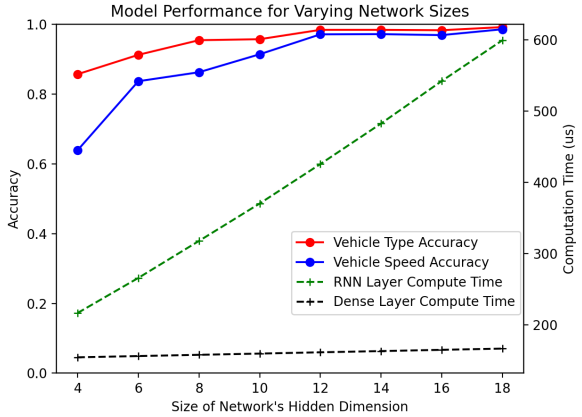

Fig. 2: Classification Performance for Varied Network Size

D. Minimal Sufficient Sampling Rate

Another opportunity to reduce computation time, and therefore save energy, is to reduce the sampling rate. This would also result in shorter sequence lengths. Our baseline model used every sample, which was collected at 800 Hz, the maximum sampling rate of the magnetometer. However, spectral analysis of our data showed these signals have little information above 80 Hz, even for vehicles faster than 60 mph; the majority of energy is within 0-50 Hz. Therefore, we know that the data sampled at 800 Hz is higher than necessary; a sampling rate of 160 Hz is sufficient based on the Nyquist rate.

This can be achieved by training and running the network on subsampled data. When applying this transformation to the data, it is crucial that the downsampling operation does not reduce the data to a point that the model cannot generalize due to the reduced volume of input data, resulting in an overfit model. With this in mind, we ran a series of experiments where our network was trained on downsampled variants of the input data, synthesized using a decimation filter that applies an anti-aliasing filter before downsampling such that the apparent sampling rate is no lower than 160Hz.

Since we have know all signal energy must resides below 80Hz, downsampling by up to a factor of four should not reduce the information content within the signal. This is observed in Table II where the network accuracy is consistent across a range of decimation factors and the computation time decreases linearly.

As energy consumption is linearly related to L as shown in Equation 1, downsampling the signal to the shortest possible representation will save energy with respect to the original signal. To reduce concerns of overfitting the data, we select for a sampling rate of 200 Hz, *i.e.*, a decimation factor of 4, to ensure all spectral content is retained. This reduces the energy consumption per inference by a factor of 4, although the gains in battery life are smaller than this due to power usage by other components, as will show in Section V.

Decimation Factor	Label Acc.	Speed Acc.	Compute Time (s)
1	0.99496924	0.9860256	0.19136
4	0.99105644	0.98658466	0.09601
8	0.99385130	0.98937952	0.04812

TABLE II: Classification accuracies and compute time for an inference of a single vehicle over a range of decimation factors. The accuracies are for the model predicting vehicle type and speed.

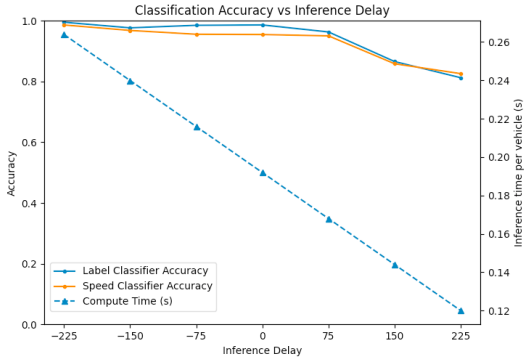

Fig. 3: Classifier accuracy when trained on delay inference from -225 to 225 samples. The delay is performed prior to the downsampling operation.

E. Delayed Inference

To achieve a long battery life, the processor needs to be duty-cycled such that the hardware spends as much time in the low-powered sleep state as possible. Measuring the entire magnetic signature of a passing vehicle is impractical, since the sensor would need to poll continuously to find the first moment the vehicle nears the magnetometer.

The processor should be active only when a car is passing. Ideally, the magnetometer too would be duty-cycled to use a lower sampling rate when cars are not passing, but react to significant changes in the magnetic field such that it wakes itself and the processor. The selected magnetometer implements this functionality via a programmable threshold and debounce timing, which in combination, allow for energy efficient vehicle detection. Low-power operation will prevent us from seeing the entire signature of the vehicle; however, we find it is sufficient to classify using a subset of that signal. We introduce a degree of freedom, "delayed inference", where the recurrent classifier begins inference a shortly after the vehicle begins to pass the sensor, which provides time to debounce the signal and wake the processor. We ran an experiment the model's performance while varying this delay from -225ms to 225ms from the vehicle passing until we start using current magnetometer samples within the recurrent model. A negative delay is meant to emulate the case where we begin running the model before the vehicle affects the magnetic field. A large positive delay may miss the vehicle entirely (depending on the vehicle speed). Figure 3 shows the impact on accuracy.

Since there is a linear relationship between energy and sequence length, L, as shown in Equation 1, a larger inference delay should be selected that still exhibits acceptable classification performance. For our network, we observe that an inference delay of 75 samples (with respect to the full 800Hz sampling rate, this is 94ms of delay) would result in the best tradeoff of performance to energy efficiency. Independent of other optimizations, using this 75 sample inference delay would reduce the energy consumption per inference by 12.5%.

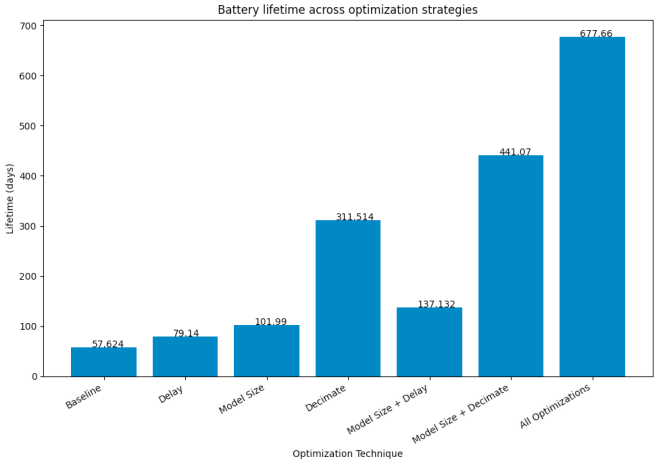

Fig. 4: Battery Life impacts of our optimizations.

Roadway Type	AADT
Highway	>15000
Arterial	10000 - 15000
Collector	5000 - 10000
Local	1000 - 5000

TABLE III: AADTs the network was evaluated on.

V. Results

For our dataset, we believe the number of measurements is small (500 measurements total, pre-augmentation) such that overfitting is difficult to avoid. We qualify that our accuracy results are questionable. Instead, we focus on our methodology for reducing energy consumption of on-board, time-series machine learning, specifically for traffic measurement. We believe these strategies for reducing sampling rate, early interference, and tuning model size are general to other time-series tinyML applications, although the exact parameters are highly application dependent.

We estimate the battery life of our system using Equation 4. Figure 4 shows the impact of our optimizations, particularly sampling rate reduction and delayed inference, on the battery life. Reducing the sampling rate has the largest impact, and has less effect on accuracy than delayed inference.

The battery life is dependent on the number of vehicles we perform inference on, *i.e.* the AADT, so we consider several roadway types following the Functional Classifications guidelines outlined by the U.S. Federal Highway Administration. States and local governments have slight variations of these guidelines, but in general, roadways are classified via travel characteristics, *i.e.,* distance served, access points, speed limit, distance between routes, usage (AADT/DVMT), significance, and number of lanes, as summarized in [14]. We use several AADT ranges from CalTrans and the City of San Jose, California to aid our analysis of battery life over a feasible set of AADTs as measured in the real world. These ranges are described in Table III.

The estimated battery life of our fully optimized RNN is shown across a range of feasible AADTs in Figure 5.

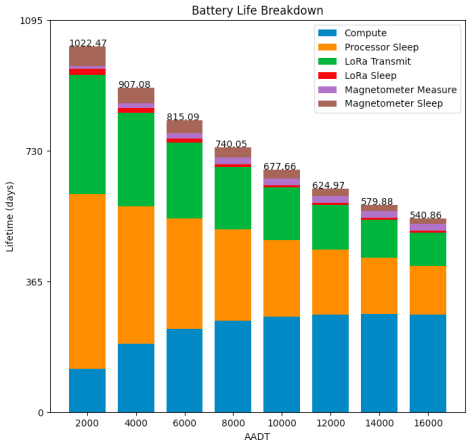

Fig. 5: Estimated lifetime for a 2Ah LiSoCl2 battery at 90% usable capacity; a LoRA radio transmits aggregates every 15 minutes @ 1 kbps (SF10, +14dBm)

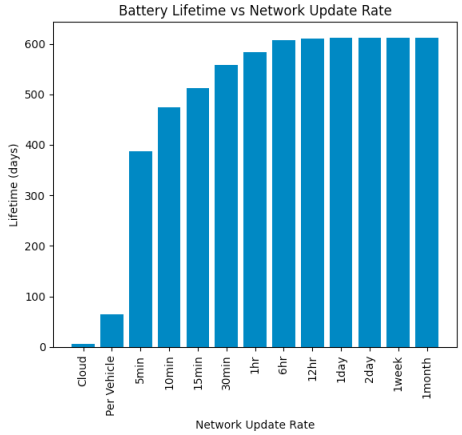

Fig. 6: Battery lifetime for varied communication intervals

This figure shows where energy is spent in the device, in which processor sleep state, active (compute) state, and LoRa transmission consume the bulk of the energy. For low AADT, the sleep state will dominate the battery life, and regularly scheduled LoRa transmissions will have an increased contribution throughout the entire lifetime. For high AADT, computation takes on a larger portion and the sleep state will have less impact as we spend less time sleeping, more time computing. If not specified, the reader may assume that we are modeling after an AADT of 10,000, as this is the midpoint between arterial and collector road traffic volumes per day.

Figure 6 shows the impact of network usage on the battery life. We base this on a LoRa radio transmitting at SF10 and TX power +14dBm. We show two degenerate cases, in which all samples are uploaded to the Cloud for inference, and when every vehicle detection is uplinked. The battery life in these cases is quite poor compared to uplinking aggregates of all vehicles measured in a recent time-frame, such as 15 minutes or a day. This impacts the battery life noticeably until the aggregates are sent on a several-hour basis, after which the average energy impact of communication diminishes.

We envision our sensor being used primarily on collector and arterial roads, in which AADT is likely within 4000 to 16000 cars per day. If sensors are deployed individually, then we would expect 1.5 - 2.8 years of lifetime per sensor.

VI. Conclusion

In this study, we have designed a battery-powered, pavement-mounted sensor that classifies vehicle type and a speed range using three-axis magnetometer readings with a recurrent neural network (RNN). We deploy this RNN to our microcontroller -based system, which must last several years without recharging or maintenance. To achieve this lifetime, we built a methodology to apply standard and application-specific optimizations to reduce the overall energy cost of vehicle detection and classification. The energy cost per inference is reduced by decreasing the sampling rate and delaying inference beyond the initial detection. These optimizations improve battery lifetime by 11.8x over the original baseline model to achieve a total lifetime of up to 2.8 years. We have demonstrated that is feasible to deploy an inexpensive, battery-powered, intelligent sensor directly to the roadways.

References

[1] H. Chourabi, T. Nam, S. Walker, J. R. Gil-Garcia, S. Mellouli, K. Nahon, T. Pardo, and H. Scholl, "Understanding smart cities: An integrative framework," *45th Hawaii International Conference on System Sciences*, pp. 2289–2297, 01 2012.

[2] K. Nellore and G. P. Hancke, "A survey on urban traffic management system using wireless sensor networks," *Sensors*, vol. 16, no. 2, 2016.

[3] S. Djahel, R. Doolan, G.-M. Muntean, and J. Murphy, "A communications-oriented perspective on traffic management systems for smart cities: Challenges and innovative approaches," *IEEE Communications Surveys Tutorials*, vol. 17, no. 1, pp. 125–151, 2015.

[4] B. Iannucci and A. Rowe, "Crowdsourced Smart Cities," in *Intelligent Transportation Systems (ITS) World Congress*, (Montreal, Quebec, Canada), 2017.

[5] J. Chung, Ç. Gülçehre, K. Cho, and Y. Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling," *CoRR*, vol. abs/1412.3555, 2014.

[6] S. Djahel, R. Doolan, G.-M. Muntean, and J. Murphy, "A communications-oriented perspective on traffic management systems for smart cities: Challenges and innovative approaches," *IEEE Communications Surveys Tutorials*, vol. 17, no. 1, pp. 125–151, 2015.

[7] S. Oh, S. G. Ritchie, and C. Oh, "Real-time traffic measurement from single loop inductive signatures," *Transportation Research Record*, vol. 1804, no. 1, pp. 98–106, 2002.

[8] Z. Zhang, T. Zhao, and H. Yuan, "A vehicle speed estimation algorithm based on wireless amr sensors," in *Big Data Computing and Communications* (Y. Wang, H. Xiong, S. Argamon, X. Li, and J. Li, eds.), (Cham), pp. 167–176, Springer International Publishing, 2015.

[9] R. David, J. Duke, A. Jain, V. J. Reddi, N. Jeffries, J. Li, N. Kreeger, I. Nappier, M. Natraj, S. Regev, R. Rhodes, T. Wang, and P. Warden, "Tensorflow lite micro: Embedded machine learning on tinyml systems," *CoRR*, vol. abs/2010.08678, 2020.

[10] E. Liberis, u. Dudziak, and N. D. Lane, "nas: Constrained neural architecture search for microcontrollers," in *Proceedings of the 1st Workshop on Machine Learning and Systems*, EuroMLSys '21, (New York, NY, USA), p. 70–79, Association for Computing Machinery, 2021.

[11] L. Heim, A. Biri, Z. Qu, and L. Thiele, "Measuring what really matters: Optimizing neural networks for tinyml," *CoRR*, vol. abs/2104.10645, 2021.

[12] C. R. Banbury, C. Zhou, I. Fedorov, R. M. Navarro, U. Thakker, D. Gope, V. J. Reddi, M. Mattina, and P. N. Whatmough, "Micronets: Neural network architectures for deploying tinyml applications on commodity microcontrollers," *CoRR*, vol. abs/2010.11267, 2020.

[13] E. Teng, C. G. Ramirez, and B. Iannucci, "Swatt: Synchronized wide-area sensing and autonomous target tracking," in *2018 IEEE International Symposium on Technologies for Homeland Security (HST)*, pp. 1–7, IEEE, 2018.

[14] "Highway functional classification concepts, criteria and procedures: 2013 edition," Tech. Rep. FWHA-PL-13-026, U.S. Department of Transportation Federal Highway Administration, Salt Lake City, UT, 2013.