Energy and Emission Prediction for Mixed-Vehicle Transit Fleets Using Multi-Task and Inductive Transfer Learning

Michael Wilbur[✉][1], Ayan Mukhopadhyay[1], Sayyed Vazirizade[1], Philip Pugliese[2], Aron Laszka[3], and Abhishek Dubey[1]

[1] Vanderbilt University, Nashville TN 37203, USA
[2] Chattanooga Area Regional Transportation Authority, Chattanooga TN, USA
[3] University of Houston, Houston TX, USA
michael.p.wilbur@vanderbilt.edu

Published in the proceedings of the 2021 European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML PKDD).

Abstract. Public transit agencies are focused on making their fixed-line bus systems more energy efficient by introducing electric (EV) and hybrid (HV) vehicles to their fleets. However, because of the high upfront cost of these vehicles, most agencies are tasked with managing a mixed-fleet of internal combustion vehicles (ICEVs), EVs, and HVs. In managing mixed-fleets, agencies require accurate predictions of energy use for optimizing the assignment of vehicles to transit routes, scheduling charging, and ensuring that emission standards are met. The current state-of-the-art is to develop separate neural network models to predict energy consumption for each vehicle class. Although different vehicle classes' energy consumption depends on a varied set of covariates, we hypothesize that there are broader generalizable patterns that govern energy consumption and emissions. In this paper, we seek to extract these patterns to aid learning to address two problems faced by transit agencies. First, in the case of a transit agency which operates many ICEVs, HVs, and EVs, we use multi-task learning (MTL) to improve accuracy of forecasting energy consumption. Second, in the case where there is a significant variation in vehicles in each category, we use inductive transfer learning (ITL) to improve predictive accuracy for vehicle class models with insufficient data. As this work is to be deployed by our partner agency, we also provide an online pipeline for joining the various sensor streams for fixed-line transit energy prediction. We find that our approach outperforms vehicle-specific baselines in both the MTL and ITL settings.

Keywords: Energy Prediction · Smart Transit · Transfer Learning · Multi-task Learning

1 Introduction

Context: Public transit agencies are focused on finding ways to make their fixed-line bus systems more energy efficient by introducing electric vehicles (EVs) and hybrid vehicles (HVs), which have reduced impacts on the environment in

comparison to traditional vehicles with internal combustion engines (ICEVs). However, EVs and HVs are expensive and in practice, transit agencies have to manage a mixed-vehicle fleet, requiring complex scheduling and assignment procedures to maximize the overall energy efficiency [13, 20, 23] while satisfying the expectations of transit demand. This is turn requires the ability to estimate energy and emissions of vehicles on assigned routes and trips. Energy prediction models can be categorized based on their modeling scale. Microscopic models aim to estimate vehicle energy consumption at a high frequency [2]; however, this comes at the cost of reduced accuracy. For most system level optimization, macroscopic models that aim to predict energy consumption at an aggregated spatial or temporal span are sufficient [1, 2].

State of the Art: There has been significant research on macroscopic models for EVs over recent years. For example, De Cauwer et al. used a cascade of ANN and linear regression models for energy consumption prediction for EVs using vehicle speed, voltage, current, SoC, road network characteristics, altitude, and weather [6]. Their model, however, did not use traffic data and the approach had a mean absolute error (MAE) of 12-14% of average trip consumption. Vepsäläinen et al. used a linear model using temperature, driver behavior, and roadway characteristics and found that EV energy consumption was 15% lower on suburban routes compared to city routes. A recent study by Pamula et al. used a DNN with stacked autoencoders and an multi-layer perceptron to predict energy consumption between stops. Their model used travel time, elevation change, and modeled weather as categorical variables [17]. However, most of the prior work relied on learning separate models for each vehicle class [1, 3, 17].

Challenges: There are several unresolved challenges for public transit operations teams. First, modern public bus fleets include not only a mix of vehicle classes (ICEV, HV, and EV), but also different vehicle models within each class. For example, out partner agency, Chattanooga Area Regional Transportation Authority (CARTA), manages a total of six ICEV models, two HV models, and two EV models. Training separate models for each type of vehicle ignores generalizable information that is not explicitly modeled in the feature space. For example, Ayman et al. modeled EVs and ICEVs without sharing model parameters between classes [1]. Second, the number of vehicles in each class varies greatly, which leads to an uneven distribution of data available for training the energy or emission prediction models. Third, and similar to the second problem in principle, when a new vehicle class is added to an existing fleet, the agency must deploy *some* vehicles, obtain data, and then learn a new predictive model from scratch.

Our Contributions: We address these challenges as multi-task learning (MTL) and inductive transfer learning (ITL) problems. Although different vehicle classes' energy consumption depends on a varied set of covariates through different non-linear functions, we hypothesize that there are broader generalizable patterns that govern the consumption of energy and vehicle emission. That is, if an agency has access to *many* vehicles, and consequently data, from each vehicle class (ICEVs, HVs, and EVs), we formulate emission (and energy) forecasting as an

MTL problem. We show that this approach improves the predictive accuracy for all vehicle classes compared to a baseline where separate networks are trained to predict emissions (and energy) for each class. In a situation with imbalanced data or when an agency introduces a new model or class, we show that it is possible to learn a model for classes with sufficient data, and *transfer* the learned abstraction to improve the predictive accuracy for the class with insufficient data. The benefit of ITL is the ability to deploy the model earlier than the time required to collect enough samples to train a separate model for the new class. Finally, we highlight that real-world transit problems require collecting, cleaning, and joining data from various sources, formats, and precision. We provide a general online pipeline for joining the various sensor streams (vehicle telemetry and trajectory data with external data sources such as weather, traffic, and road infrastructure) for training and maintaining the fixed-line transit energy prediction models. We evaluate our MTL and ITL models using real-world data from our partner agency's mixed-fleet of EVs, HVs, and ICEVs. We show that in both the MTL and ITL settings, our approach outperforms state-of-the-art methods. The greatest improvements over baselines were in the ITL setting when the target vehicle class suffers from a lack of data. However, we also find that in some cases ITL does not work well, such as when transferring learned abstractions from EV to ICEV.

2 Model

2.1 Predicting Energy Consumed and Emissions

Transit agencies are concerned with reducing a) costs by limiting energy used, and b) the impact of their vehicles on the environment by reducing emissions. For ICEVs and HVs, energy expended by a vehicle is a function of the fuel consumed, measured in liters. On the other hand, the energy expended by an EV is a function of the dissipated charge of its battery, which is the change in its state-of-charge (SOC). This presents a problem since transit agencies primarily use prediction models to optimize the assignment of vehicles to trips. As a consequence, they require a common metric to compare across vehicle classes in their mixed-fleet for both energy consumed and emissions. For energy, we use kWh. For ICEVs and HVs, we convert liters of diesel fuel consumed to kWh using a conversion rate of 10.639 kWh/liter [7]. For EVs, we multiply the change in SOC and the capacity of the battery. We measure emissions as kg of CO_2. For ICEVs and HVs, fuel consumed in liters can be converted to emissions (kg CO_2) at a rate of 2.689 kg/liter [8]. For EVs, dissipation in charge can be converted to emissions (kg CO_2) at a rate of 0.707 kg/kWh [8]. As shown in Figure 1, the function $g_i(\hat{Y}_i)$ represents the linear conversion between the predicted target (emission) and energy consumed for an arbitrary vehicle class denoted by the index i.

2.2 Preliminaries and Model Formulation

Our goal is to learn energy consumption and emissions in a mixed fleet of vehicles conditional on a set of relevant determinants (Figure 1). We refer to learning

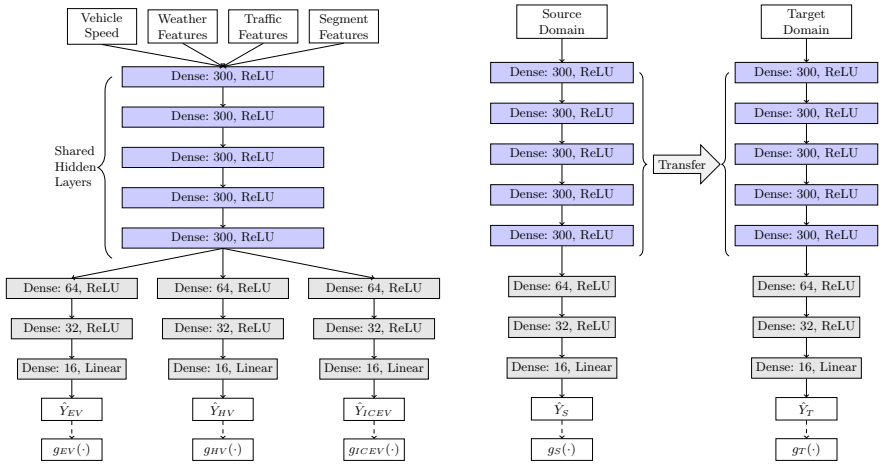


Fig. 1: (a) MTL Model: DNN with hard parameter sharing for predicting emissions (kg CO_2) of EVs ($\hat{Y}_{EV}$), HVs ($\hat{Y}_{HV}$) and ICEVs ($\hat{Y}_{ICEV}$). (b) ITL Model: shared-hidden layer parameters are frozen and transfered to the target model. Energy consumed (kWh) is a linear function $g_i(\cdot)$ for vehicle class i, separate of the neural network per the conversion discussed in Section 2.1.

prediction models as *tasks*, consistent with the terminology in the area of transfer learning [18]. We introduce the formalism for our problem next. We define a *domain* $\mathcal{D}$ as the combination of a feature space $\mathcal{X}$ and a probability distribution $P(X)$, where $X = \{x_1, x_2, \dots\} \in \mathcal{X}$. For example, $\mathcal{X}$ can include features like vehicle speed and weather. Given a specific domain, a *task* is then defined as $\mathcal{T} = \{\mathcal{Y}, f(\cdot)\}$ where $\mathcal{Y}$ is the space of output labels, and f is a predictive function over $y \in \mathcal{Y}$ conditional on x. Probabilistically, f denotes the probability of a realization of y given x ($P(y \mid x)$). For example, Y can denote the energy consumed by a vehicle, and subsequently, the function f can be used to denote a distribution on the energy consumed conditional on the determinants. Typically, f is unknown; instead, we assume access to observations (data) in the form of input-output pairs $\{(x_1, y_1), (x_2, y_2), \dots, (x_m, y_m)\}$. We deal with a scenario with multiple tasks (and associated domains). Specifically, there are three vehicle classes, and therefore three domains $\mathcal{D}_{EV}, \mathcal{D}_{HV}, \mathcal{D}_{ICEV}$ representing the domains of EVs, HVs and ICEVs, respectively. Similarly, we have three output label spaces $\mathcal{Y}_{EV}$, $\mathcal{Y}_{HV}$, and $\mathcal{Y}_{ICEV}$, and three predictive functions f_{EV}, f_{HV}, and f_{ICEV}, which need to be learned.

The functions f_{EV}, f_{HV}, and f_{ICEV} are parameterized by a set of parameters θ, that we seek to learn by minimizing a predefined loss function given the observed data. The input features for each of the vehicle class domains are derived from the characteristics of the road segments, weather, traffic features, and vehicle dynamics. Therefore, we can state that the feature spaces are equivalent, $X_{EV} =$

Table 1: Data description; data collected from Jan 1 2020 to July 1 2020.

Data Source	Description	Features	Frequency	Scope
ViriCiti - ICEVs	vehicle telemetry	fuel level, GPS	1 Hz	3 vehicles
ViriCiti - HVs	vehicle telemetry	fuel level, GPS	1 Hz	4 vehicles
ViriCiti - EVs	vehicle telemetry	current, voltage, GPS	1 Hz	3 vehicles
Clever Devices	automated vehicle location	trip ID, vehicle ID	0.1 Hz	all vehicles
HERE	traffic (per TMC)	jam factor, current speed, free flow speed	0.0166 Hz	major roads, highways
DarkSky	weather	visibility, wind speed, precipitation intensity, humidity, wind gust, temperature	0.0033 Hz	whole city
Static GTFS	transit schedule	routes, trip IDs, stop sequences, stop locations (latitude, longitude), schedule trip times, trip shape (GeoJSON)	static	whole city
GIC - Elevation	LiDAR elevation	location, elevation (meters)	static	whole city
Trip Segments	multiple sources	segment length, time to travel, average speed, roadway type	static	whole city

$X_{HV} = X_{ICEV}$. Additionally, the marginal probability distributions over the features are independent of vehicle class and therefore, the marginal probability distributions over the features are equivalent, $P(X_{EV}) = P(X_{HV}) = P(X_{ICEV})$. Finally, given that the feature spaces and marginal probability distributions are the same for all vehicle classes, we have that $\mathcal{D}_{EV} = \mathcal{D}_{HV} = \mathcal{D}_{ICEV}$.

As the energy consumed for ICEV and HV vehicles are measured in fuel (liters) consumed, the space of output labels $\mathcal{Y}$ is the set of positive real numbers $\mathbb{R}_+$. On the other hand, EV vehicles have regenerative braking, therefore the energy consumed can take negative values and the task space for EV vehicles is $\mathbb{R}$. Additionally, since the performance of the three vehicle classes varies greatly, we consider that the predictive functions for each vehicle class are different; as the conditional probability distributions are not equal and $P(Y_{EV}|X_{EV}) \neq P(Y_{HV}|X_{HV}) \neq P(Y_{ICEV}|X_{ICEV})$. Finally, we can generalize the problem to n classes of transit vehicles in the fleets (e.g. the classes can be categorized based on the model and year as well); such a generalization will focus on learning the tasks $\{\mathcal{T}_1 \neq \mathcal{T}_2, \cdots, \neq \mathcal{T}_n\} \in \mathcal{T}$, given the domains $\{\mathcal{D}_1 = \mathcal{D}_2, \cdots, = \mathcal{D}_n\} \in \mathcal{D}$.

3 Approach

We now discuss our approach to learning the energy prediction functions (f_{EV}, f_{HV}, and f_{ICEV}). In order to perform data-driven learning, we first need to accumulate data from various sources. In real-world problems pertaining to public transportation, creating a data pipeline is often an arduous task due to the variety of data sources, formats, recording precision, and data collection frequency. As a result, we begin by discussing the data sources (Table 1) and the data pipeline. We gather data from 3 ICEVs, 4 HVs, and 3 EVs from our partner agency for a period of six months from January 1, 2020 to July 1, 2020. Each vehicle has a telematics kit produced by ViriCiti LLC [22], that provides speed and GPS

positioning at a minimum of 1Hz resolution. In addition, for ICEVs and HVs the sensors provide fuel consumed (liters) while for EVs we collect current and voltage levels which are used to calculate energy consumed as well as emissions. Each vehicle is equipped with a kit from Clever Devices [4]. The Clever Devices feed provides a unique vehicle ID corresponding to the vehicle ID in the ViriCiti feed, as well as the unique trip ID which maps to scheduled trips in the static General Transit Feed Specification (GTFS) [21].

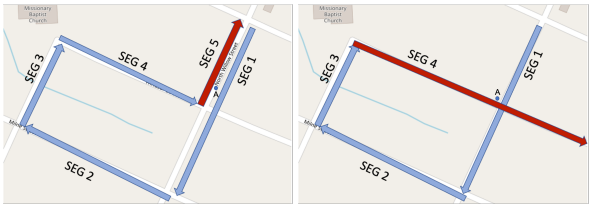
We collect weather data from multiple weather stations within the transit region at 5-minute intervals using the DarkSky API [5], including temperature, humidity, wind speed, and precipitation. Traffic data was collected at 1-minute intervals using the HERE API [11], which provides speed recordings for segments of major roads. The traffic data is reported per *TMC* (Traffic Message Channel), which is a custom geographical mapping unit. We perform map matching similar to prior work [1] to obtain traffic data for each road segment of interest to us. Road network map data was collected from OpenStreetMaps [9]. Lastly, we collect static GIS elevation data from the state Geographic Information Council which provides high-resolution digital elevation models (DEMs) derived from LiDAR elevation imaging, with a vertical accuracy of approximately 10cm.

Fixed line transit vehicles travel at pre-determined times (trips) covering a sequence of stops along a route. The latitude and longitude of each stop and the geographical shape of the path (the route segment) that the vehicles travel by visiting each stop is specified using the static GTFS schedule published by CARTA. Using this information, it is straightforward to divide the path taken by a bus during a given trip into a sequence of segments $\langle SEG \rangle$, where each segment is marked by a start stop and an end stop. As the specific characteristics of segments are important, a unique segment is created for every spatial path that exists between a pair of stops. Note that effectively, each SEG_i is described using a discrete sequence of points (latitude and longitude), close enough to draw the shape of the road on the map. We use these segments as the fundamental spatial unit for which we predict emissions (or energy). This has two advantages: first, the generation of route segments for prediction can be derived directly from a transit agency's schedule, rather than relying on external infrastructure data such as OSM [1] or time intervals [3], and second, segments can be shared between trips thereby providing additional data for learning.

3.1 Mapping Vehicle Trajectories to Route Segments

To generate the joined data samples, we first map the vehicle trajectories to segments. By joining the ViriCiti and Clever Device feeds, we determine a set of GPS points that a vehicle traverses. We refer to this ordered sequence of points as a trajectory T consisting of spatial points $\{l_1, l_2, \dots\}$. Consider that the trajectory T serves the trip R. The goal of the mapping process is to label each location $l_i \in T$ to a corresponding segment $SEG_j \in R$, thereby representing the specific segment that each vehicle traverses at a specific point in time.

In principle, it is possible to perform an exhaustive search on the segments to identify the one that matches (or is the closest to) each point in a trajectory.



(a) (b)

Fig. 2: (a) Overlapping segments. Segments 1 and 5 traverse the same section in opposite directions. (b) Intersecting segments. Vehicle locations near the intersection of segments 1 and 4 can lead to incorrect mapping. Stops not shown.

However, such an approach does not work in practice with real-word trajectory feeds due to two reasons. First, routes often traverse segments between spatial points close to each other during trips. For example, consider the overlapping segments in Figure 2a in which the vehicle passes through SEG_1 relatively early in the trip and through SEG_5 later. Due to noise in the measurement, a point early in the trip can erroneously get mapped to SEG_5, resulting in incorrect representation of the features that are induced by the segment. Similar problems arrise when segments cross each other as shown in Figure 2b. Our exploratory analysis on the data obtained from our partner agency showed several examples of such incorrect mappings. Second, the mapping of trajectory data to segments is computationally challenging for transit agencies. As an example, consider our partner agency CARTA, which operates a total of 60 vehicles. The number for bigger cities is larger in orders of magnitude; for example, the New York Metropolitan Transit Authority (NY-MTA) operates more than 5000 buses [16]. Considering location data collected at the frequency of 1 Hz for 3 years, the matching must be done for over 3.5×10^9 spatial locations, each of which could potentially be mapped to one out of hundreds of segments (for a larger city like New York, the number of matches is 3×10^{12}).

To alleviate these concerns, we propose an algorithm for mapping vehicle trajectories to route segments (Algorithm 1). The algorithm takes the trajectory T of the vehicle traversing the sequence of segments $\langle SEG \rangle$ of trip R. During matching, we maintain a lookahead window, denoted by W, that represents the number of segments to consider for the match. For example, if a location $l_i \in T$ is already matched to segment SEG_c in a route, then for matching the next location $l_{i+1} \in T$, we consider the set $\{SEG_c, \ldots, SEG_{c+W}\}$. By maintaining a short lookahead, we alleviate duplicate matches from segments further away in the route. Also, a shorter lookahead provides computational efficiency as opposed to an exhaustive search. We maintain a tolerance distance B for matching where a segment is matched to a location from a trajectory only if the distance between them is less than or equal to B. The function $dist(SEG_j, l_i)$ is used to calculate the minimum distance between segment SEG_j and GPS point l_i.

Algorithm 1: Mapping Trajectories to Route Segments

Input:
$R \leftarrow$ sequence of segments $\{SEG_0, \ldots, SEG_N\}$ for each trip
$T \leftarrow$ set of vehicle GPS locations l along the trip
$W \leftarrow$ number of segments to lookahead
$B \leftarrow$ max distance between segment and vehicle GPS
Output:
$TrajSegMap \rightarrow$ list of segments for each SEG in R
Initialization:
$c \leftarrow 1$, index of current segment
$TrajSegMap \leftarrow []$
for $i \in \{1, \ldots, |T|\}$ **do**
  $SegWindowDist \leftarrow []$
  for $j \in \{c, \ldots, c + W\}$ **do**
    if $j \leq |R|$ **then**
      $SegWindowDist.push(dist(SEG_j, l_i))$
  if $min(SegWindowDist) \leq B$ **then**
    $c \leftarrow c + argmin(SegWindowDist)$ $TrajSegMap[i] \leftarrow SEG_c$
  else
    $TrajSegMap[i] \leftarrow None$

3.2 Generating Samples

To generate the joined data samples, we split each of the trajectories T based on the locations mapped to trip segments. We create one data sample per continuous travel on a trip segment, providing average speed and the total fuel/energy consumption and emission on that segment. For ICEVs and HVs, the fuel consumed is provided in liters. While EVs provide state-of-charge (SOC) readings, the precision is too low to use for representing energy consumed. Therefore, we estimate the amount of energy from the battery current A and voltage V. The energy used between consecutive data points is given by $E_i = A_i \cdot V_i \cdot (TS_i - TS_{i-1})$, where E_i, A_i, and V_i are the consumed energy (Joule), current (Ampere), voltage (Volt) at time step i, respectively, and TS_i is the timestamp (in seconds) at time step i. To get the energy on a segment, the energy consumed between each sample is accumulated for all locations of the vehicle mapped to that segment.

Weather features for each sample are taken from the weather reading closest to the time at which the vehicle starts traversing a segment. For traffic features, we take the average jam factor (JF) and speed ratio (SR) of all TMCs mapped to the segment traversed by the vehicle when the vehicle enters a segment. Speed ratio is defined as the traffic speed divided by the free flow speed.

3.3 Learning

Recall that our goal is to address two specific problems. The first scenario is where a transit agency has access to *many* vehicles, and consequently data, from each

vehicle class. In this case, our goal is to improve the predictive accuracy of f for all tasks. One method of addressing this problem is to learn a predictive model f over each vehicle class. However, we hypothesize that there are generalizable patterns between vehicle classes that can be leveraged to aid learning. Consequently, we formulate a MTL model as shown in Figure 1a. We use hard parameter sharing to learn a common representation of the input features which enables us to extract generalizable patterns across the tasks. Additionally, each task (vehicle class) has a vehicle-specific set of hidden layers which outputs the predicted energy consumed/emissions for EVs ($\hat{Y}_{EV}$), HVs ($\hat{Y}_{HV}$), and ICEVs ($\hat{Y}_{ICEV}$) along route segments. At each training iteration, a batch of samples from EVs, HVs, and ICEVs is fed through the network and mean-squared error (MSE) loss is calculated between the predicted target and true target for each vehicle class. The gradient of the loss is then propagated back through the network.

The second problem we seek to address is where an agency has significant variation in the number of vehicles from each class. In such a case, while a common model can be learned using the MTL framework, the tasks with a significantly larger number of samples are likely to *dominate* learning. Also, learning a model solely for the task with few samples can result in overfitting. In this case, we seek to learn f for classes with sufficient data (source model) first, and *transfer* the learned abstraction to improve the predictive accuracy for the class with insufficient data (target model). Our ITL framework is shown in Figure 1b. When training the target model, the transferred layers are frozen and only the vehicle-specific layers are updated during training.

4 Experiments and Results

Vehicle telemetry, weather, and traffic data is collected for a six-month period between January 1, 2020 and July 1, 2020 for 10 vehicles as shown in Table 2. We include two post-processing steps in generating the final datasets for each respective vehicle class. First, we remove partial trajectories by eliminating samples where the total distance traveled was less than 50% of the segment length and greater than 150% of the segment length. Second, to address outliers and potential errors in the mapping process, we remove samples with the target value (energy/emission) in the bottom 2% and top 2% quantiles. The final data size is shown in last column of Table 2.

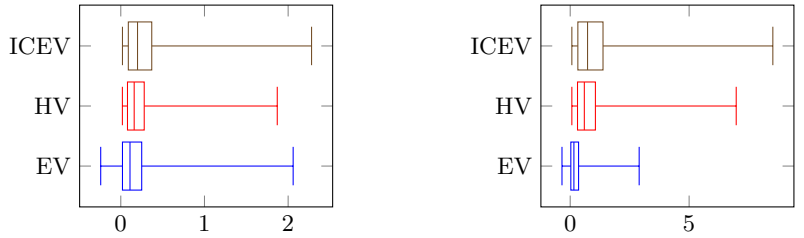
The distributions of emissions (kg CO_2) and energy (kWh) consumption are shown in Figure 3. As energy consumption for ICEVs and HVs is derived from liters of diesel fuel consumed, emissions must be greater than 0 kg CO_2 for these vehicle classes. The EVs in the fleet have regenerative braking, which allows for energy consumed, and thus emission, to be negative. We predict energy/emissions per route segment. The distributions over energy and emission for each of the vehicle classes has a long right tail and the average varies between vehicle classes. Therefore, the *task* of energy/emission prediction is also different, by virtue of having a different distribution over the space of output labels $\mathcal{Y}$.

Table 2: Data processing summary.

Class	Model Year	Vehicles	Raw Samples	Distance Filtering	Final Samples
ICEV	2014	3	139,652	127,212	114,348
HV	2014	4	235,671	223,913	201,491
EV	2018	3	48,969	47,804	43,022

Table 3: Pearson's correlation coefficient of input features with emissions.

Class	Length Segment	Time to Travel	ΔElevation	max ΔElevation	Speed Ratio	Visibility	Wind Speed	Precipitation	Humidity	Wind Gust	Jam Factor	Temperature	Avg Speed
EV	0.860	0.752	0.523	0.222	0.038	0.008	-0.002	-0.003	-0.009	-0.012	-0.015	-0.037	-0.093
HV	0.916	0.838	0.505	0.135	0.038	0.006	0.004	-0.008	-0.008	-0.002	-0.026	0.013	-0.134
ICEV	0.886	0.865	0.539	0.103	0.028	0.004	0.011	-0.005	0.001	≈ 0	-0.016	-0.005	-0.262



(a) Emissions (kg CO_2) (b) Energy consumption (kWh)

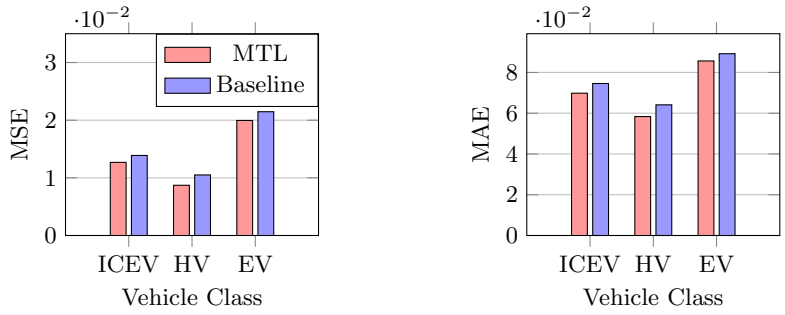
Fig. 3: Distribution of (a) emissions (kg CO_2) and (b) energy (kWh) consumption per trip segment for each vehicle class.

The Pearson correlation coefficient between input features and emissions is provided in Table 3. Distance traveled and time to traverse the segment have a strong positive correlation with emissions. Δ Elevation, which is the change in elevation from the start to the end of the segment, also has a strong correlation with emissions for all vehicle types. Max Δ elevation, which is defined as the difference between the maximum and minimum elevation along the segment, has a relatively weaker correlation. Additionally, the average vehicle speed has a stronger negative correlation of -0.262 with emissions for ICEVs than with HVs (-0.134) and EVs (-0.093).

4.1 Hyperparameter Tuning and Baseline Models

We randomly select 43,022 samples from each vehicle class. For each vehicle class, we use 80% of the samples for training and 20% for testing. Of the training samples, 10% are withheld from training and used as a validation set to identify the best set of hyperparameters for the subsequent analyses. We perform the hyperparameter search using the model derived from the MTL formulation.

We tested shared hidden layer widths of {200, 300, 400} and shared hidden layer depths of {3, 4, 5}. We use 3 vehicle-specific layers and tested the configurations of {128, 64, 32} and {64, 32, 16}. Mean-squared error (MSE) is used for the loss function and the networks are optimized using the Adam algorithm [12].



(a) MSE (b) MAE

Fig. 4: (a) MSE and (b) MAE of MTL model compared to vehicle-specific neural network models (baseline) on testing set. Prediction target: emissions (kg CO_2).

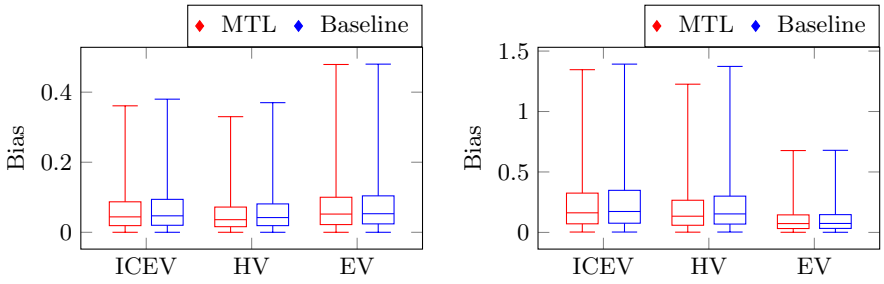
We test learning rates of {0.01, 0.005, 0.001, 0.0005, 0.0001} and batch sizes of {64, 128, 256, 512}. The best performing configuration is shown in Figure 1a, which consists of 5 shared hidden layers of 300 fully connected neurons with ReLU activation functions [15], and 3 vehicle-specific hidden layers of 64, 32, and 16 hidden neurons respectively. For the output layer we test using ReLU as well as linear activation functions for ICEVs and HVs and linear activation function for EVs, however we find that using a linear activation function as the output layer for all 3 vehicle classes provides the best performance. An early stopping strategy was performed, where we stopped training if MSE on the validation set did not improve for 10 epochs. The best performing learning rate was 0.0005 and the best batch size was 256.

In the baseline model no layers are shared between vehicle-classes resulting in a separate neural network for each vehicle class. The same grid search from the proposed models was used to find the hyperparameters of the baseline models. In all experiments, we use Kaiming initialization [10] to initialize the weights of the networks.

4.2 Multi-task Model Evaluation

First, we investigate the performance of the MTL model compared to vehicle-specific baseline models. To evaluate the robustness of the models, we train 10 MTL models (30 vehicle-specific models, 10 for each vehicle class) and present the average MSE and MAE in Figure 4. Models are trained for up to 150 epochs. We find that for all vehicle classes, the MTL model outperforms the vehicle-specific baseline models. The mean percent improvement in MSE is 8.6%, 17.0%, and 7.0% for ICEVs, HVs, and EVs, respectively. The mean percent improvement in MAE is 6.4%, 9.0% and 4.0% for ICEVs, HVs, and EVs respectively.

Even with improved accuracy, it is important to investigate the bias and the variance of the proposed approaches. Therefore, we repeat the entire evaluation using 30 datasets creating through bootsrapping [19] from the original data. At



(a) Emissions (kg CO_2) (b) Energy (kWh)

Fig. 5: Distribution of MTL and baseline model bias per sample for each vehicle class from bootstrap evaluation, 30 bootstrap iterations. Prediction target: (a) emissions and (b) energy.

each iteration, we sample a training set, with replacement, from the ICEV, HV, and EV datasets. The samples not selected for each training set are used as the testing set for that iteration. For each iteration, we train a single MTL model and vehicle-specific baseline models on the training set and evaluate on the testing set. The distribution of empirical bias per sample for the MTL and baseline models is presented in Figure 5. We observe that the MTL model results in a lower bias for each vehicle class compared to the baseline models. The MTL model also results in lower median variance per sample.

4.3 Inductive Transfer Learning Evaluation

Next, we evaluate the performance of the ITL model formulated in Figure 1b. To train the ITL models, we use data from all of the three vehicle classes, each of which contains 43,022 samples, as outlined in section 4.1. For each pair of source and target task, we first train the source model, freeze the shared hidden layers, and transfer to the target model. Then, we optimize the target model's vehicle-specific layers. For each model, the available sample size to train the target model is varied from 2%, 5%, 10%, and 15% of the total number of available samples to investigate the influence of sample size in training of the target models. This is consistent with what transit agencies might face in practice; as a new vehicle is introduced, agencies gradually collect more data from it. We test our approach for all pairs of vehicle classes.

To compare the performance of the models, we train baseline models that only use the training data from the target domain. For example, while evaluating inductive transfer from EV to ICEV with 2% of the target data available, the baseline model is trained exclusively on the same amount data from ICEV class. In order to consider the randomness in training process, when evaluating the target and baseline models, we trained each model 10 times on 10 random samples from the target domain's dataset and 10 different initial values for the parameters using Kaiming initialization [10].

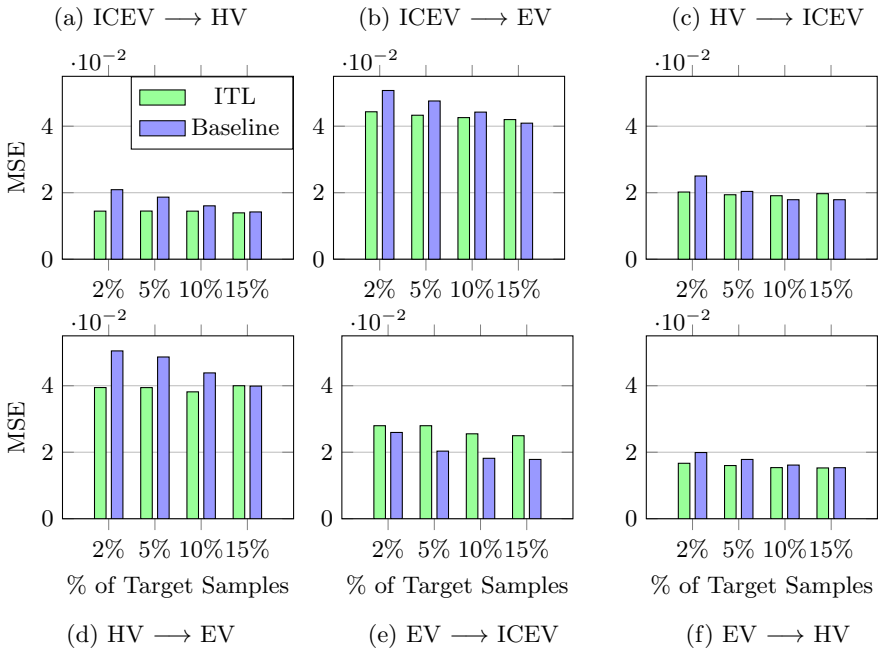


Fig. 6: ITL models compared to corresponding baselines. ITL model is trained on full dataset in the source vehicle class and is evaluated on the target vehicle class (source ⟶ target). Average MSE compared to fraction of data samples used for training in the target vehicle class. Prediction target: emissions (kg CO_2).

We provide the results of the proposed ITL approach in Figure 6. We observe that in general, the proposed approach results in improved forecasting accuracy across the tested scenarios (except when EV is used as source and ICEV is used as target). We also observe that as the amount of data from the target domain increases, both the ITL and the baseline method show improved forecasting accuracy; however, the baseline methods shows relatively higher improvement, to the extent of outperforming the ITL framework in some cases (15% data from target domain in Figure 6 b, c, e and f).

Additionally, we seek to understand the role of the shared-hidden layers in our proposed approach. Conceptually, the role of such layers in the target model is to extract generalizable patterns across the spectrum of tasks to aid learning in the target task. We use t-distributed stochastic neighbor embeddings (t-SNE) [14] to visualize the separation of multi-dimensional information in a two-dimensional space. In Figure 7, we show t-SNE on the raw input features of the three vehicle classes color coded by emissions (kg CO_2). All three plots are very similar, thereby corroborating our assumption that the input features are similar across the tasks ($\mathcal{D}_{EV} = \mathcal{D}_{HV} = \mathcal{D}_{ICEV}$). We separately apply t-SNE on the output of the shared-hidden layers across all pairs of source and target

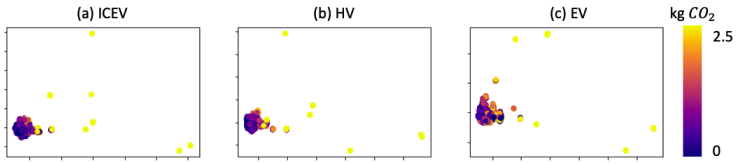


Fig. 7: t-SNE on raw input features for each data sample from the source domain. t-SNE parameters: number of components=2, perplexity=10, initialization=PCA, number of samples=860 (2% of dataset)

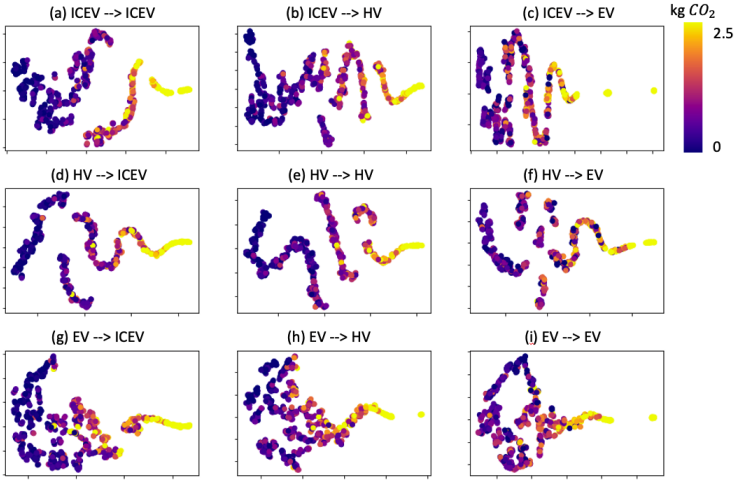


Fig. 8: t-SNE on the output of shared-hidden layers for each data sample from the target domain. t-SNE parameters same as Figure 7.

tasks and show the results in Figure 8. We observe that the ICEV source model and HV source model (plots (a) to (f) of Figure 8) effectively discriminate the samples with high emissions and low emissions (increasing the distance between light points and dark points). On the other hand, EV source model (plots (g) to (i) in Figure 8) shows poor discrimination, reflecting the negative transfer.

4.4 Discussion

We now present the key takeaways from the experiments. First, we observe that in general, both the MTL and the ITL framework outperform the baseline methods, thereby resulting in improved emission (and consequently energy) predictions for transit agencies that operate mixed-fleet vehicles. Second, we observe that the MTL, ITL, and baseline models are less accurate in predicting EV emissions compared to HV and ICEV, most likely due to the complexity of the energy cycle in EV engines. Third, a key finding for practitioners is that the greatest improvements over baselines are seen when the target vehicle class suffers from

lack of data. However, it is important to switch to standard models once sufficient data is collected for the class. The point at which such a switch should be made depends on the specific task and data at hand. In our work with CARTA, we implement a periodic check to facilitate such a switch. Fourth, we find that when the goal is to predict the emissions for ICEV class using a source model trained based on EV class dataset (this situation rarely arises in practice due to precedence of the ICEV class), ITL models underperformed baseline models, irrespective of the size of training data from the target domain. This indicates negative transfer between the EV domain and the ICEV domain.

Lastly, while this work is a general approach that can be used by cities to improve their energy prediction models there are a couple limitations agencies should be aware of. First, our models were trained on data from Chattanooga, TN, which is a mountainous city in the southern United States with a warm climate and limited snowfall or freezing temperatures. Therefore any direct transfer of our pre-trained models to other cities should take into account potential biases in these determinants. Second, like most macroscopic energy prediction models we do not take into account the impact of delays at stops or the number of passengers on the vehicles. We intend on incorporating these parameters into future work.

Code, data, and supplementary results of this study are available at `https://github.com/smarttransit-ai/ECML-energy-prediction-public`

5 Conclusion

By framing emission (and energy) forecasting as an MTL problem, we showed that an agency with access to *many* vehicles can improve the predictive accuracy for EVs, HVs, and ICEVs over current state-of-the-art, vehicle-specific models. We also showed that in a situation with imbalanced data the predictive accuracy of classes with insufficient data can be improved by transferring a learned abstraction from vehicle classes with sufficient data through ITL. Lastly, we provided a general online pipeline for joining the various sensor streams for emission and energy prediction of mixed-vehicle transit fleets.

Acknowledgments

This material is based upon work supported by the National Science Foundation under grant 1952011 and Department of Energy, Office of Energy Efficiency and Renewable Energy (EERE), under Award Number DE-EE0008467. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation or the Department of Energy.

References

1. Ayman, A., Sivagnanam, A., Wilbur, M., Pugliese, P., Dubey, A., Laszka, A.: Data-driven prediction and optimization of energy use for transit fleets of electric and ICE vehicles. ACM Transations of Internet Technology (2020)

2. Chen, Y., Wu, G., Sun, R., Dubey, A., Laszka, A., Pugliese, P.: A review and outlook of energy consumption estimation models for electric vehicles. International Journal of Sustainable Transportation, Energy, Environment, & Policy (2021)
3. Chen, Y., Zhu, L., Gonder, J., Young, S., Walkowicz, K.: Data-driven fuel consumption estimation: A multivariate adaptive regression spline approach. Transportation Research Part C: Emerging Technologies **83**, 134–145 (2017)
4. Clever Devices API documentation. https://www.cleverdevices.com/ (2020)
5. Dark Sky API documentation. https://darksky.net/dev/docs (2019)
6. De Cauwer, C., Verbeke, W., Coosemans, T., Faid, S., Van Mierlo, J.: A data-driven method for energy consumption prediction and energy-efficient routing of electric vehicles in real-world conditions. Energies **10**(5), 608 (2017)
7. EIA energy conversion calculator. https://www.eia.gov/energyexplained/units-and-calculators/energy-conversion-calculators.php (2021)
8. EPA greenhouse gases calculator. https://www.epa.gov/energy/greenhouse-gases-equivalencies-calculator-calculations-and-references (2021)
9. Haklay, M., Weber, P.: OpenStreetMap: User-generated street maps. IEEE Pervasive Computing **7**(4), 12–18 (2008)
10. He, K., Zhang, X., Ren, S., Sun, J.: Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 1026–1034 (2015)
11. HERE API documentation. https://developer.here.com/documentation (2019)
12. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)
13. Lajunen, A.: Energy consumption and cost-benefit analysis of hybrid and electric city buses. Transportation Research Part C **38**, 1–15 (2014)
14. van der Maaten, L., Hinton, G.: Visualizing data using t-SNE. Journal of Machine Learning Research **9**(86), 2579–2605 (2008)
15. Nair, V., Hinton, G.E.: Rectified linear units improve restricted boltzmann machines. In: Proceedings of the 27th International Conference on International Conference on Machine Learning. p. 807–814. ICML'10, Omnipress, Madison, WI, USA (2010)
16. NY MTA subway and bus facts 2019. https://new.mta.info/agency/new-york-city-transit/subway-bus-facts-2019 (2019)
17. Pamuła, T., Pamuła, W.: Estimation of the energy consumption of battery electric buses for public transport networks using real-world data and deep learning. Energies **13**(9), 2340 (2020)
18. Pan, S.J., Yang, Q.: A survey on transfer learning. IEEE Transactions on Knowledge and Data Engineering **22**(10), 1345–1359 (2009)
19. Parr, W.C.: A note on the jackknife, the bootstrap and the delta method estimators of bias and variance. Biometrika **70**(3), 719–722 (1983)
20. Sivagnanam, A., Ayman, A., Wilbur, M., Pugliese, P., Dubey, A., Laszka, A.: Minimizing energy use of mixed-fleet public transit for fixed-route service. In: Proceedings of the 35th AAAI Conference on Artificial Intelligence (AAAI-21) (2021)
21. Static GTFS reference. https://developers.google.com/transit/gtfs/reference (2021)
22. ViriCiti SDK documentation. https://sdk.viriciti.com/docs (2020)
23. Wu, X., Freese, D., Cabrera, A., Kitch, W.A.: Electric vehicles' energy consumption measurement and estimation. Transportation Research Part D: Transport and Environment **34**, 52–67 (2015)