Genetic Programming with Transfer Learning for Urban Traffic Modelling and Prediction

Anikó Ekárt
Computer Science, Aston University
Birmingham, UK
https://orcid.org/0000-0001-6967-5397

Alina Patelli
Computer Science, Aston University
Birmingham, UK
https://orcid.org/0000-0002-8945-6711

Victoria Lush
Computer Science, Aston University
Birmingham, UK
v.lush1@aston.ac.uk

Elisabeth Ilie-Zudor
Institute for Computer Science and Control, Hungarian Academy of Sciences
Budapest, Hungary
https://orcid.org/0000-0001-9362-3538

***Abstract*—Intelligent transportation is a cornerstone of smart cities' infrastructure. Its practical realisation has been attempted by various technological means (ranging from machine learning to evolutionary approaches), all aimed at informing urban decision making (e.g., road layout design), in environmentally and financially sustainable ways. In this paper, we focus on traffic modelling and prediction, both central to intelligent transportation. We formulate this challenge as a symbolic regression problem and solve it using Genetic Programming, which we enhance with a lag operator and transfer learning. The resulting algorithm utilises knowledge collected from other road segments in order to predict vehicle flow through a junction where traffic data are not available. The experimental results obtained on the Darmstadt case study show that our approach is successful at producing accurate models without increasing training time.**

***Index Terms*—Genetic Programming, Transfer Learning, Symbolic Regression, Intelligent Transportation, Traffic Prediction**

I. Introduction

Intelligent transportation [1], [2] is a technology-driven approach to road traffic management, a critical milestone on the path towards realising the smart cities vision [3]. When implemented, Intelligent Transportation Systems (ITS) would feature increased traffic fluidity, ultimately leading to a significant reduction in pollution, delays, infrastructure costs and accident frequency. There are several computational tools with a well documented potential to underpin the practical realisation of ITS, namely, vehicular-infrastructure communications [4]–[7], evolutionary algorithms [8]–[11], machine learning [12]–[14], blockchain [15], [16] and game theory [17], to name just the well established ones.

Amongst these, Genetic Programming (GP) [18], [19] - a particular type of evolutionary algorithm - shows remarkable promise when it comes to urban traffic modelling and prediction, which are central to ITS implementation. Traffic modelling is essentially a symbolic regression problem, thus offering the ideal application space for GP. The crux of its appeal is best communicated by Sipper and Moore, in their analysis of GP's recent history [19]. The very first item they include in a list of themes and challenges extracted from 15 years of GP theory and practice is symbolic regression. This is deemed to be "at the heart of many complex, real-world problems", a category that traffic modelling and prediction fall neatly under. The second item in the list recommends that "in real life one must progress beyond pure, vanilla GP." Our take on this guideline is to enrich traditional GP with transfer learning for traffic modelling.

This work was supported by the European Commission through the H2020 project EXCELL (hp://excell-project.eu/), grant No. 691829.

Transfer learning [20], [21] employs knowledge acquired on a previous problem to solve new problems that are similar to the original one. We apply this to traffic modelling, by training a model on data collected from a *source* junction and using it to predict traffic on a topologically similar, *target* junction, that is geographically distinct from the source. This is necessary in a variety of practical situations, e.g, missing or faulty sensors, leading to the absence or unreliability of traffic flow readings on the target junction. In such cases, the only solution for predicting future traffic through the target junction is to build a model on data collected from topologically similar source junctions. The question arising is whether the models obtained via transfer learning are competitive (of similar accuracy and/or produced in a similar amount of time) when compared to those one could train exclusively on the target junction (*native* or *reference* models), should the necessary data exist. From a methodology perspective, we investigate two approaches to answer the above question:

- Use *one* source junction for training and transfer the fittest model over to the target junction. We will refer to this approach as Single Source Transfer Learning (SSTL).
- Use *several* source junctions for chain training (transfer the fittest model obtained on the first source to the second source, where training is resumed, and then transfer the final model over to the target junction). We will refer to this approach as Multiple Source Transfer Learning (MSTL).

We propose the GENetic Programming with Transfer LEarning (GENTLE) algorithm to generate SSTL and MSTL models. We evaluate GENTLE models on a set of target junctions (for which complete and reliable traffic readings exist) selected from the road network of Darmstadt, Germany. Based on our results, we propose that GENTLE is adopted as a decision support tool for urban traffic prediction, whenever junction data are not available.

The following section presents a concrete example where transfer learning models are the only way to predict traffic through a target junction. The current research on traffic modelling and prediction is discussed in the background section, which also provides a description of transfer learning and its use in GP. Section IV explains the GENTLE algorithm and is followed by the analysis of the experimental results obtained by deploying GENTLE on the Darmstadt case study. The conclusions are outlined in section VI.

II. Motivating Scenario

We consider urban traffic through the road network of Darmstadt, Germany[1]. For this case study, we selected junction A13, located on a busy main road (Fig. 1a), alongside junctions A21, A36 and A51, of a similar structure (Fig. 1b). Normally, a junction would be equipped with sensors (e.g., induction loops) to count vehicles passing through all inflow and outflow lanes. This would make it possible to model the traffic flow through each outflow lane as a function of the traffic flow through the inflow lanes, provided that good quality data are collected by the sensors, over a sufficiently long period of time. For example, the A13 traffic flow model, $F(X0, \dots, X4)$, evolved by our GP based algorithm on the training data available, will produce an output (prediction) which we will compare to the values in the validation data set. A close correspondence will indicate that the model could be reliably used to predict future traffic through A13's lanes, thus supporting long term urban infrastructure planning. We envisage that our proof-of-concept GENTLE method is one of many novel computational tools that can improve urban long-term decision making.

In practice, A13 data may be unavailable, due to sensor related issues (equipment is faulty, improperly installed or missing altogether) or because the junction has not yet been built. A model of traffic flow through A13 is still needed to predict future vehicle counts, in order to support real time decisions, such as modifying planned travel routes due to large volumes of expected traffic, short term decisions, e.g., investing in sensor repairs or replacements, and long term decisions, such as determining where to build a new junction to minimise congestion. The solution is to generate the model using traffic data collected on nearby junctions with a topology similar to A13 (similar number of inflow and outflow lanes). For instance, the model could be trained on junction A21 (Fig. 1b), which thus becomes the *source* junction, and deployed to predict traffic through A13, the *target* junction. This is a case of single source transfer learning. Alternatively, several source junctions (e.g., A21 and A36 or any other combination of junctions with a similar layout) could be used to train the model for target A13, which is a case of multiple source transfer learning.

[1]https://darmstadt.ui-traffic.de/faces/TrafficData.xhtml

Without data available for the target junction A13, the transfer learning models cannot be validated. In other words, there is no reliable basis to determine which junction(s) – A21, A36, A51 or a specific combination of these – make for the best sources. We show – through full factorial experiments (see section V-B), conducted on Darmstadt junctions where reliable data are available – that most GENTLE models are similar quality predictors. Therefore, the choice of source junction(s) can rely solely on the quality of available data.

III. Background

A. *Traffic Modelling and Prediction*

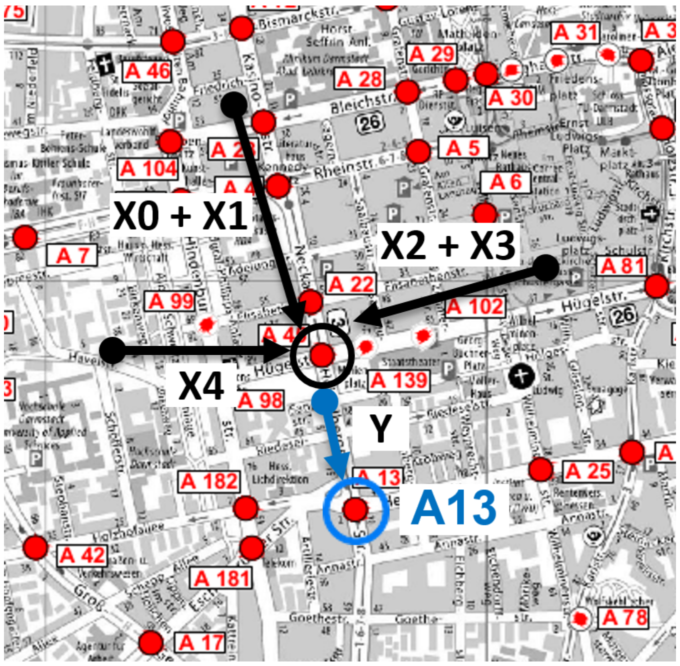
Practical ITS implementations, such as interACT and CoEXist, both EU projects for connected automated driving [22], or the European vehicle-to-vehicle and vehicle-to-infrastructure communications network [23], mostly rely on direct interpretation of sensed data. This enables the development of short-term improvements: notifying drivers of a car stopped ahead, tracking high occupancy vehicles with cameras, providing live updates on traffic density as sensed by road level induction loops, etc. However, the true power of data analytics (data-driven traffic modelling and future pattern prediction for long term decision making, such as infrastructure planning) is not fully embraced, in a strategically relevant way, by the world's major municipalities. This is disappointing, given the wealth of promising research results available, where the proposed algorithms are successfully deployed on publicly available traffic data collected from major cities. In that vein, we mention a few of the most recent efforts: traffic congestion detection in downtown Ottawa [5], road link speed prediction in Rome [12], traffic flow calculation throughout the road network of Malaga (with a customised evolutionary algorithm) [10], traffic assignment in Springfield, IL (employing a Genetic Algorithm in combination with other computational tools) [24], traffic lights control in Stockholm (also with a Genetic Algorithm) [25] and in Bologna (via epigenetics inspired GP) [26], [27] and traffic flow and pollution modelling in Montevideo [28].

Transfer learning can add value to existing approaches by increasing the overall model robustness (accuracy and shelf-life), whilst maintaining (or even reducing) the training time necessary.

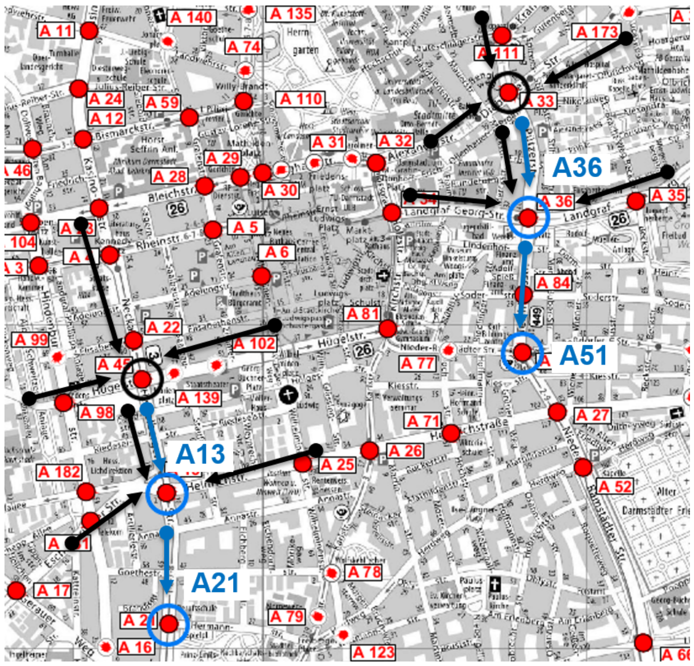
B. *Transfer Learning*

As the term suggests, transfer learning aims to transfer what was learned on a previous problem *in the past* to a new, but similar problem *now*. The goal is to make the learning faster or more effective [20], [21]. The number of real-world applications is growing fast, examples include reinforcement learning for videogame AI [29], text-mining [30], [31], cancer subtype discovery [32], and building-space optimisation [33].

The knowledge is transferred from a so-called source task to a so-called target task, and these may be the same or different. The broad motivation for this area has been to achieve lifelong machine learning, namely, faster and more effective machine



(a) Map view of junction A13. Arms X0 through X4 flow into the junction, arm Y flows out of the junction.



(b) Map view of all Darmstadt junctions used for the case study. Black arrows mark inflow, blue arrows mark outflow.

Fig. 1: Section of the Darmstadt road network retrieved from https://darmstadt.ui-traffic.de/faces/TrafficData.xhtml.

learning built on previous knowledge instead of starting from scratch, similarly to human learning.

Transfer learning is categorised based on the questions of ”what to transfer”, ”how to transfer” and ”when to transfer”. The first two questions are relatively straight-forward to answer, however, the third one requires predicting when the transfer will be beneficial (or not). In inductive transfer learning, the source and target tasks are different, while the source and target domains may be the same or different. In transductive transfer learning, the source and target tasks are the same, while the source and target domains are different [20]. A lot of transfer learning involves initial training on the source domain and is followed by – possibly shorter – training on the target domain, and this relies on the availability of labelled target domain data. Transfer is still possible in the more challenging case where labelled target domain data is scarce or not available at all [34].

The domain is defined as the combination of input space, output space and associated probability distribution. Two domains are considered different if they differ in at least one component. In the general case for transfer learning, the source and target domain may differ in any of the three components. The particular case where the input and output space are the same for source and target domains and only the probability distributions change is called *domain adaptation*.

When transferring models between junctions with the exact same topology, both input and output space are identical between the junctions and our traffic modelling problem is a domain adaptation problem. However, if the topologies are similar, but not identical, the input space of the two domains is not the same and the more general transfer learning term applies.

C. Transfer Learning in Genetic Programming

Investigations on the potential of GP, and in particular symbolic regression, for transfer learning are under way and the results on specifically designed benchmarks for three types of transfer (namely, relational-knowledge transfer, feature transfer and model transfer, relevant to the ”how to transfer” question) are promising [35].

Historically, we can trace back transfer learning in GP to population seeding [36], [37] and layered learning [38]. Instead of random individuals, Langdon and Nordin started GP with a set of perfect individuals, solving all or a subset of the fitness cases and then used a multiobjective approach for the two objectives of fitness and size to obtain generalised solutions [36]. They demonstrated success on three classification problems. Schmidt and Lipson investigated methods for employing expert knowledge in evolutionary search, based on simulating the collection of expert knowledge for problems, by optimising an approximated version of the exact solution to each problem [37]. Their methods included using the entire approximate model at once and breaking it into pieces. They found that using just parts of a solution was often more effective than using the full expert solution.

Layered learning is a technique designed for the effective evolutionary solving of difficult problems, which are decomposed into simpler ones, each associated with one layer. The solution to the complex problem is then incrementally built from a solution to a simple problem. Jackson and Gibbons identified two approaches that do not require prior understanding of a problem's functional decomposition into subgoals. One of these led to a solution-finding performance that is significantly better than that of standard GP systems and of those incorporating automatically defined functions [38]. Thus, in essence, this method achieved positive transfer of knowledge.

One of the earliest explicit applications of transfer learning, as such, in GP involved transferring best solutions or parts of solutions from the final generation of a GP run to a target problem, using various heuristics and hyper-parameters to guide the amount of transfer [39]. This method outperformed standard GP on two sets of symbolic regression problems. O'Neill et al. propose transferring common subtrees from solutions to two source problems to the target problem, by adding these to the function set for the target problem [40]. They demonstrate success on benchmark symbolic regression and Boolean even-N parity problem domains, by reaching either improved or similar quality solutions, when compared to standard GP or Dinh et al's method [39]. Thus, when training data on the target domain are available, GP with transfer learning is a promising avenue to explore.

More recently, Santana et al. have considered classifier transferability in domain adaptation and demonstrated their solution on the real world problem of brain decoding across subjects [41]. Generally, with this sort of application, classifiers trained on some subjects are reused on other subjects. Their approach includes both maximising classifier accuracy in the source domain and maximising a transferability measure that they define for GP programs. Two of their GP-based classifiers, specifically evolved for maximising transferability, proved significantly better than known state-of-the-art classifiers.

The goal of this section was to highlight (1) essential landmarks in the progress of transfer learning in GP so far and (2) some notable successes that motivated our approach. For an up-to-date review of transfer learning in GP, the reader is referred to the study of Muñoz et al. [42]. Following the review of the field, they conduct an extensive study on a variety of source and target tasks [42]. Muñoz et al. study the relationship between success and failure and also analyse the predictability of transfer learning performance on both classification and regression tasks. They find that while some problems are good as sources, others are good as targets and that transferability is not always symmetric between two problems.

IV. The GENTLE Algorithm

We have developed GENTLE as a transfer learning solution for urban traffic prediction, however, its underlying logic may be used to solve any symbolic regression problem. The foundation of the GENTLE algorithm is the $SymbolicTransformer$ class from the $gplearn$ Python library[2], which implements the classic GP loop with a twist: the API provides a way to pause training and resume it for an additional number of generations, which comes in handy when using two source junctions for chain training, as is the case with MSTL. To meet the specifics of the traffic modelling problem space, we have enhanced the $SymbolicTransformer$'s library implementation with:

- *A lag function.* To account for temporal dependencies in road traffic, we have added a lag function to the arithmetic operator set. The lag function delays an input/output, $z(t)$, by one time unit, such that $lag(z(t)) = z(t-1)$. In GP, operators are included recursively in the evolved models (trees), thus creating implicit scope for utilising lags of higher orders (from 1 to the maximum allowed depth, d, of regression trees), $lag^d(z(t)) = z(t-d)$. This is computationally cheaper than including lagged inputs in the terminal set.
- *A memory mechanism.* Let us assume an MSTL scenario, where A13 is the target junction (this is the motivating scenario analysed in section II). There are six possible combinations of source junctions, out of which we will consider two: (A21, A36) and (A21, A51). Once training is paused on A21, the default `SymbolicTransformer` allows it to be resumed on A36. This is done by transferring the final population of models obtained on A21 over to A36, where it becomes the initial population. If we attempt to repeat the process for the (A21, A51) pair of source junctions, the default `SymbolicTransformer` will start training on A21 from scratch. For transfer learning, one should not need to create a new model from the source data every time transfer is applied. For chain training in MSTL, the same A21 model should be available for continued training, irrespective of whether the next junction in the chain is A36 or A51. To facilitate this, we have adapted the library code, in order to "remember" the final population obtained on the first source junction and use it as a unique starting point for continued training on any other junction that may serve as the second source.

Algorithm 1 outlines our full factorial experiment of deploying GENTLE to the Darmstadt case study, consisting of the four junctions shown in Fig. 1b: A13, A21, A36 and A51. There are two required inputs, namely, $target$, representing the junction through which we wish to predict traffic, and a boolean parameter, $MSTL$, indicating the type of transfer learning employed (single, in which case $MSTL$ is false, or multiple, when $MSTL$ is true). The algorithm returns an array, $RMSE$, of Root Mean Squared Error values, one for each single or multiple source transfer learning model produced (the array will contain 3 elements if $MSTL$ is false and 6 otherwise). After initialisation, the algorithm loops through all possible source junctions (line 6) and gathers the readings collected by their sensors. These will be used as training data, to be fed into the $estimator$ (line 8), an instance of the $SymbolicTransformer$ enhanced with "lag" and "memory", as described above. The fit method called on

[2]https://gplearn.readthedocs.io/en/stable/reference.html#symbolic-transformer

Algorithm 1: A full factorial GENTLE application on Darmstadt junctions

```
1:  inputs: $target$, $MSTL$
2:  returns: $RMSE$
3:  $junctions \leftarrow [A13, A21, A36, A51]$
4:  $RMSE$ = []
5:  $(x\_test, y\_test) \leftarrow get\_data(target)$
6:  for $source1$ in $junctions \setminus \{target\}$ do
7:      $(x\_train, y\_train) \leftarrow get\_data(source1)$
8:      $P \leftarrow estimator.fit(x\_train, y\_train, NULL)$
9:      if $MSTL$ == true then
10:         for $source2$ in $junctions \setminus \{target, source1\}$ do
11:             $(x\_train, y\_train) \leftarrow get\_data(source2)$
12:             $estimator.fit(x\_train, y\_train, P)$
13:             $y\_pred \leftarrow estimator.transform(y\_test)$
14:             $RMSE.app(\sqrt{\sum_i (y\_pred[i] - y\_test[i])^2})$
15:         end for
16:     else
17:         $y\_pred \leftarrow estimator.transform(y\_test)$
18:         $RMSE.app(\sqrt{\sum_i (y\_pred[i] - y\_test[i])^2})$
19:     end if
20: end for
21: return $RMSE$
```

line 8 will randomly generate an initial population (hence, the third parameter is NULL) and use the training data to evolve the trees within. After the maximum number of generations is reached, the final population, P, is returned as well as stored internally by the $estimator$. In the case of single transfer ($MSTL$ is false), the algorithm continues with line 17, where the $transform$ method extracts the best model (with the lowest error on training data) from the $estimator$'s internal copy of P and calculates its outputs, *y_pred*, on test data, *y_test*. The differences between the values in the two sets will be aggregated and appended to the $RMSE$ array, by calling function *app* (line 18). In the case of multiple transfer ($MSTL$ is true), an additional loop (line 10) is nested inside the previous one, to run through all possible second sources. The $estimator$ continues training the models in P on each of the available second sources. Once this is complete, the best model from the final population, stored inside the $estimator$, is validated by the $transform$ function (line 13) in the same way as in the case of single transfer.

V. Experimental Analysis and Evaluation

Ideally, data-driven models of traffic through a target junction should be trained on that very junction. Unfortunately, in practice, data may not be available to train "native" models - this is the case if the target junction has not yet been built, or if, for some reason (e.g., related to cost or location), it is not possible to equip it with traffic monitoring equipment, or if existing sensors are faulty. In any of these situations, the solution is to use models trained on similar junctions in order to predict traffic through the target one. Our GENTLE algorithm addresses this challenge, by training models on one (Single Source Transfer Learning) or more (Multiple Source Transfer Learning) nearby junctions with similar topologies. In this section we provide numerical evidence, in the three practical scenarios described above, that GENTLE models are competitive, in terms of both accuracy and training time, when compared against native models, therefore, they are viable in the absence of native models.

A. Darmstadt Case Study and Experimental Setup

The set of junctions from Darmstadt used to evaluate GENTLE is presented in the motivating scenario (section II). We have downloaded and analysed a large volume of the traffic measurements available for the four selected junctions, to find that the data collected in the five weeks between the 28th of August and 1st of October 2017 were most suitable (captured traffic activity that is typical for a major city and featured few missing values) for our experimental analysis. We used the first three weeks of data for training and the final two of the five weeks of data for testing. All data used in this experiment are sampled every 15 minutes. Table I contains statistical information for the test data on each junction, to put into context the reference models' absolute RMSE values, as presented in Table III.

TABLE I: Test data statistics. The values represent the lowest, highest, median and mean number of cars passing through a given junction, across all 15 min intervals in the two weeks' worth of test data.

Junction	Min	Max	Median	Mean
A13	0	67	28	26.31
A21	0	45	24	21.12
A36	0	127	35	33.92
A51	0	45	17	15.99

To ensure the repeatability of the experiments presented in this section, the full set of values used to configure the $SymbolicTransformer$ is given in Table II.

TABLE II: GENTLE configuration

Parameter	Value
population size	1000
generations	20,30
stopping criteria	0.01
crossover prob	0.5
subtree mutation prob	0.3
hoist mutation prob	0.05
point mutation prob	0.1
max samples	0.9
parsimony coefficient	0.005

To give an idea of the internal structure of GENTLE models, we provide the mathematical expression of the A13 reference one: $y = 0.42x_0 + 0.42lag(x_0) + 0.176lag^2(x_0) + 0.074lag^3(x_0) + x_1 + 0.42x_2 + 0.074lag^3(x_2) + 0.596$.

B. Results and Analysis

Table III presents the accuracy, in the form of RMSE values, of the reference, the single source and the multiple source transfer learning models, on the four junctions from the Darmstadt road network. Fig. 2 shows the same data on a linear scale, to highlight the relative difference between the precision of various models. We consider each of the four junctions, A13, A21, A36 and A51, as target junctions and show the RMSE values on unseen test data for (1) the reference model trained on data from the target junction, (2) the SSTL models, when all other junctions are used, in turn, as source junctions, and (3) the MSTL models, when all combination of two junctions different to the target are used as sources. Reference and SSTL models are trained for 30 generations, whereas MSTL ones are trained for 15 generations on each source junction. Conducting a full factorial experiment allows us to understand whether transfer learning models can be reliably deployed in lieu of a native model, in situations where no training data are available on the target junction.

A13 results discussion: The native model's RMSE value is 6.95, meaning that, on average, it yields a traffic volume prediction that is off by ± 6.95 cars, compared to the sensor-measured (real) values in the test data. As expected, SSTL models are less accurate, with the model trained on A21 being closest to the native one - this model's output is 2.31 cars less precise than the reference. MSTL models perform better, with the one trained on A21 and A36 featuring an accuracy loss of only ± 0.66 cars, compared to the reference. If we compare the entire set of MSTL models against that of SSTL ones (Fig. 2), we find that the overlap is quite small. We can therefore infer that, in the absence of data collected from A13, an MSTL model is more likely to yield an accurate prediction for this target junction.

A21 results discussion: The best single source transfer learning model, the one trained on A36, outputs a prediction within a margin of 1.29 cars, relative to the reference. This is slightly more precise that the best multiple source transfer learning model (the one trained on A51 and A36), where the accuracy loss relative to the reference is of 2.01 cars. The least precise models are the one trained on A36 and A51, under-performing by 9.07 cars, and, respectively, the one trained on A51, which lags behind the reference by 9.15 cars. Given the large overlap between the accuracy intervals covered by single source and multiple source transfer learning models for target junction A21 (Fig. 2), we cannot recommend one method over the other, in terms of predicting the flow of traffic.

A36 results discussion: The best MSTL model, trained on A13 and A21, exceeds the precision of the reference by 0.35 cars, which represents an *accuracy gain* of 4.34%. This result is the only one that surpasses the quality of the native model, making it the absolute best prediction, across the entire set of transfer learning models, on all four junctions considered in this experiment. In contrast, the worst multiple source transfer model, trained on A21 and A51, outputs a traffic flow that is 13.84 cars less precise than the reference. This is the worst accuracy obtained in this study. The SSTL models show less variation, in RMSE terms, with the best (trained on A21) lagging 0.3 cars behind the reference and the worst (trained on A51) under-performing by 13.69 cars. Comparing the two categories of models, SSTL and MSTL (Fig. 2), the best MSTL one outperforms the most accurate SSTL one by 0.65 cars, whereas the relative difference at the opposite end of the accuracy spectrum is smaller - 0.15 cars. In other words, in the absence of A36 data, the decision maker runs a slightly smaller risk of selecting an inferior model if picking from the MSTL lot, as opposed to the SSTL one.

A51 results discussion: The best MSTL model, trained on A21 and A13, is 1.84 cars less precise than the reference. The most accurate SSTL model, trained on A13, lags behind the reference by 4.34 cars. The MSTL model trained on A21 and A13 outperforms this SSTL model by 2.5 cars. The least accurate models are the one trained on A36 and A21 (5.02 cars less precise than the reference) and, respectively, the one trained on A21 (5.47 cars less reliable than the reference) - with a difference of 0.45 cars between them, with the MSTL model being more precise. Given the relative differences in accuracy, at both ends of the RMSE axis (Fig. 2), it is more likely that the decision maker will select a superior model if picking from the MSTL lot of predictors.

Overall results discussion: Fig. 2 shows that, for target junctions A13 and A51, multiple source transfer learning models significantly outperform single source transfer learning ones, whereas, for target junction A36, this effect is only marginal. In the case of target junction A21, there is no significant dominance of either category of models. Given that MSTL either outperforms SSTL or is not distinguishable from it, where a sufficient number of topologically similar junctions are available, multiple source transfer learning models are recommended. Another insight that is difficult to glean from Table III, but is clearly revealed by Fig. 2 is the distribution of the models' accuracy. With the exception of A51, most of the transfer learning models are clustered towards the left hand side of the axes - closer to the reference, with a small minority concentrated on the far right. On the upside, knowing that one can expect a larger number of accurate transfer learning models than poor ones is very useful when target junction data are missing and the decision maker needs to "blindly" pick a substitute model. On the flip side, in case a poor model does get selected, the accuracy loss is likely to be substantial. This calls for further investigation into junction topology similarity, with the aim of providing the decision maker with some indication of which transfer learning models are likely to be good (and which are likely to be poor) substitutes for the native model, an aspect we will target in our future work.

VI. Conclusions and Future Work

The United Nations project a growth in the world's urban population from 55% today to 68% by 2050.[3] This increase in urban areas will be accompanied by an expanding road network infrastructure, adding more scope for existing traffic

[3]https://www.un.org/development/desa/en/news/population/2018-revision-of-world-urbanization-prospects.html

TABLE III: Accuracy of GENTLE models on all target junctions. RMSE values are calculated as shown in Algorithm 1, on line 18, where i loops over all 15 min sampling intervals in the test data.

Model type	**Target junction A13**		**Target junction A21**		**Target junction A36**		**Target junction A51**	
	Source junction	***RMSE***	***Source junction***	***RMSE***	***Source junction***	***RMSE***	***Source junction***	***RMSE***
reference	A13	6.95	A21	5.26	A36	8.07	A51	7.44
single source transfer learning	A21	9.26	A13	7.11	A13	9.61	A13	11.78
	A36	12.96	A36	6.55	A21	8.37	A21	12.91
	A51	20.63	A51	14.41	A51	21.76	A36	12.51
multiple source transfer learning	A21, A36	7.61	A13, A36	8.69	A13, A21	7.72	A13, A21	12.24
	A21, A51	9.51	A13, A51	14.18	A13, A51	21.86	A13, A36	12.35
	A36, A21	7.68	A36, A13	6.83	A21, A13	8.47	A21, A13	9.28
	A36, A51	8.55	A36, A51	14.33	A21, A51	21.91	A21, A36	12.02
	A51, A21	7.9	A51, A13	7.31	A51, A13	8.99	A36, A13	9.63
	A51, A36	8.43	A51, A36	6.68	A51, A21	7.82	A36, A21	12.46

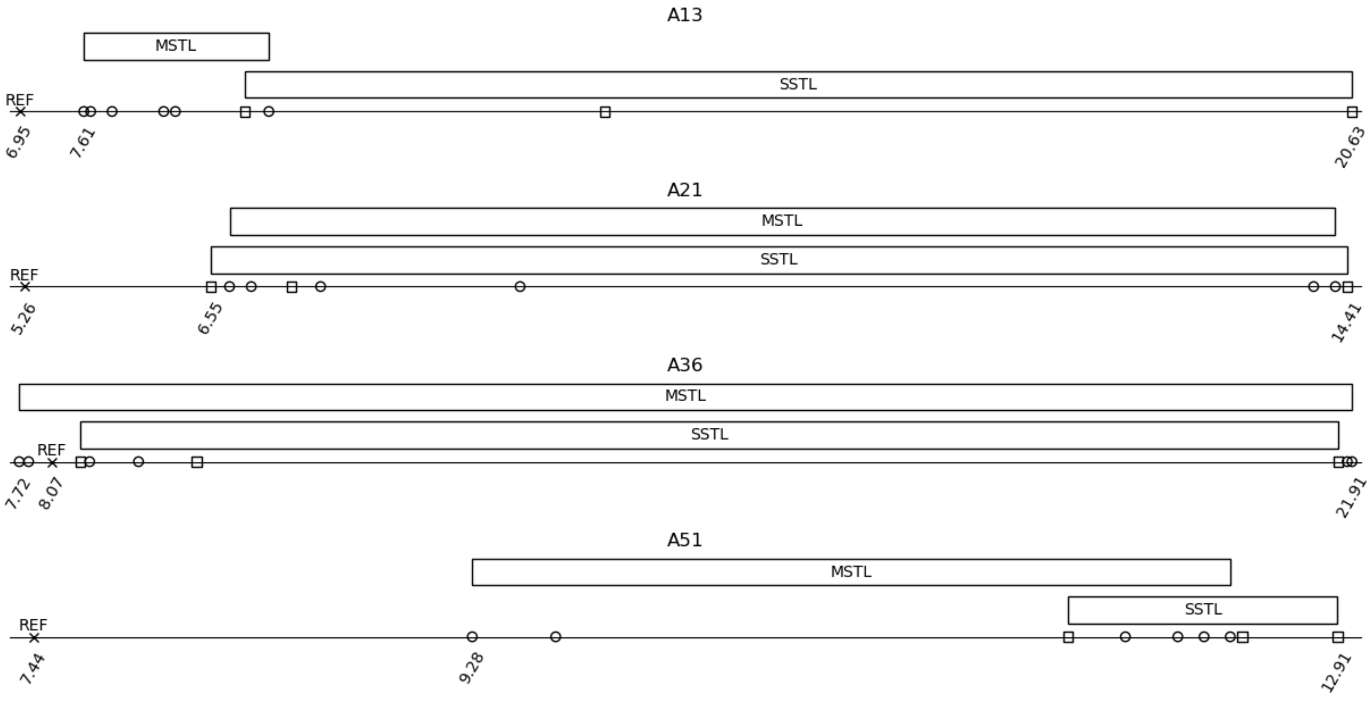


Fig. 2: Relative accuracy of GENTLE models on all target junctions. The RMSE values are represented on the horizontal axes - squares for SSTL models and circles for MSTL ones. The bars run across the accuracy interval - from the lowest to the highest RMSE - covered by SSTL and MSTL models, respectively.

related problems, from pollution and congestion to raising maintenance costs and accidents. This intensifies the urgency of introducing intelligent solutions for road design and exploitation, specifically concerning long-term decision support.

In this context, we proposed the development of transfer learning models, which we demonstrated to be effective at predicting traffic through junctions with no available traffic data. For this, we used real data collected from a set of junctions from the Darmstadt road network. We showed that models transferred from topologically similar junctions, located near the target one, predict traffic with competitive accuracy relative to native models, with no need for supplementary training. To generate transfer learning models, we introduced GENTLE, an algorithm suitable for solving symbolic regression problems, enhanced with a lag operator and a memory for storing the population of models to be transferred from the source junction to the target one.

We envisage that our proof-of-concept GENTLE method is one of many novel computational tools that can improve urban long-term decision making. If successfully integrated, such computational support has the potential of ushering in intelligent transportation on a significantly larger, better coordinated scale than the isolated applications we benefit from today.

REFERENCES

[1] J. Zhang, F.-Y. Wang, K. Wang, W.-H. Lin, X. Xu, and C. Chen, "Data-driven intelligent transportation systems: A survey," *IEEE Transactions on Intelligent Transportation Systems*, vol. 12, no. 4, pp. 1624–1639, 2011.

[2] M. Chowdhury, A. Apon, and K. Dey, *Data analytics for intelligent transportation systems*. Elsevier, 2017.

[3] A. Camero and E. Alba, "Smart city and information technology: A review," *Cities*, vol. 93, pp. 84–94, 2019.

[4] A. Daniel, A. Paul, A. Ahmad, and S. Rho, “Cooperative intelligence of vehicles for intelligent transportation systems (ITS),” *Wireless Personal Communications*, vol. 87, no. 2, pp. 461–484, 2016.

[5] M. B. Younes and A. Boukerche, “A performance evaluation of an efficient traffic congestion detection protocol (ECODE) for intelligent transportation systems,” *Ad Hoc Networks*, vol. 24, pp. 317–336, 2015.

[6] O. Popescu, S. Sha-Mohammad, H. Abdel-Wahab, D. C. Popescu, and S. El-Tawab, “Automatic incident detection in intelligent transportation systems using aggregation of traffic parameters collected through V2I communications,” *IEEE Intelligent Transportation Systems Magazine*, vol. 9, no. 2, pp. 64–75, 2017.

[7] X. Cheng, L. Yang, and X. Shen, “D2D for intelligent transportation systems: A feasibility study,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 16, no. 4, pp. 1784–1793, 2015.

[8] D. H. Stolfi and E. Alba, “Sustainable road traffic using evolutionary algorithms,” in *Sustainable Transportation and Smart Logistics*. Elsevier, 2019, pp. 361–380.

[9] J. Ferrer, M. López-Ibáñez, and E. Alba, “Reliable simulation-optimization of traffic lights in a real-world city,” *Applied Soft Computing*, vol. 78, pp. 697–711, 2019.

[10] D. H. Stolfi and E. Alba, “Generating realistic urban traffic flows with evolutionary techniques,” *Engineering Applications of Artificial Intelligence*, vol. 75, pp. 36–47, 2018.

[11] ——, “Green swarm: Greener routes with bio-inspired techniques,” *Applied Soft Computing*, vol. 71, pp. 952–963, 2018.

[12] G. Fusco, C. Colombaroni, L. Comelli, and N. Isaenko, “Short-term traffic predictions on large urban traffic networks: Applications of network-based machine learning models and dynamic traffic assignment models,” in *2015 International Conference on Models and Technologies for Intelligent Transportation Systems (MT-ITS)*. IEEE, 2015, pp. 93–101.

[13] A. Ferdowsi, U. Challita, and W. Saad, “Deep learning for reliable mobile edge analytics in intelligent transportation systems: An overview,” *IEEE Vehicular Technology Magazine*, vol. 14, no. 1, pp. 62–70, 2019.

[14] Z. Yang and L. S. Pun-Cheng, “Vehicle detection in intelligent transportation systems and its applications under varying environments: A review,” *Image and Vision Computing*, vol. 69, pp. 143–154, 2018.

[15] A. Lei, H. Cruickshank, Y. Cao, P. Asuquo, C. P. A. Ogah, and Z. Sun, “Blockchain-based dynamic key management for heterogeneous intelligent transportation systems,” *IEEE Internet of Things Journal*, vol. 4, no. 6, pp. 1832–1843, 2017.

[16] S. Bao, Y. Cao, A. Lei, P. Asuquo, H. Cruickshank, Z. Sun, and M. Huth, “Pseudonym management through blockchain: Cost-efficient privacy preservation on intelligent transportation systems,” *IEEE Access*, vol. 7, pp. 80 390–80 403, 2019.

[17] R. Hernández, C. Cárdenas, and D. Muñoz, “Game theory applied to transportation systems in smart cities: analysis of evolutionary stable strategies in a generic car pooling system,” *International Journal on Interactive Design and Manufacturing (IJIDeM)*, vol. 12, no. 1, pp. 179–185, 2018.

[18] J. R. Koza, *Genetic Programming: On the Programming of Computers by Means of Natural Selection*. Cambridge, MA, USA: MIT Press, 1992.

[19] M. Sipper and J. H. Moore, “Genetic programming theory and practice: a fifteen-year trajectory,” *Genetic Programming and Evolvable Machines*, vol. 21, p. 169–179, 2020.

[20] S. J. Pan and Q. Yang, “A survey on transfer learning,” *IEEE Transactions on Knowledge and Data Engineering*, vol. 22, no. 10, pp. 1345–1359, 2010.

[21] J. Lu, V. Behbood, P. Hao, H. Zuo, S. Xue, and G. Zhang, “Transfer learning using computational intelligence: A survey,” *Knowledge-Based Systems*, vol. 80, pp. 14 – 23, 2015, 25th Anniversary of Knowledge-Based Systems.

[22] S. Fischer, E. Machek, H. Rakoff, and S. Sloan, “European Union-United States–Japan cooperation on intelligent transportation systems research and deployment: 2017 international accomplishments summary,” United States Department of Transportation. Intelligent Transportation Systems Joint Program Office, Tech. Rep., 2018.

[23] K. Sjoberg, P. Andres, T. Buburuzan, and A. Brakemeier, “Cooperative intelligent transport systems in Europe: Current deployment status and outlook,” *IEEE Vehicular Technology Magazine*, vol. 12, no. 2, pp. 89–97, 2017.

[24] A. Hajbabaie and R. F. Benekohal, “A program for simultaneous network signal timing optimization and traffic assignment,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 16, no. 5, pp. 2573–2586, 2015.

[25] J. Jin, X. Ma, and I. Kosonen, “A stochastic optimization framework for road traffic controls based on evolutionary algorithms and traffic simulation,” *Advances in Engineering Software*, vol. 114, pp. 348–360, 2017.

[26] E. Ricalde and W. Banzhaf, “A genetic programming approach for the traffic signal control problem with epigenetic modifications,” in *Proceedings of the 19th European Conference on Genetic Programming (EuroGP 2016)*, ser. LNCS, M. I. Heywood, J. McDermott, M. Castelli, E. Costa, and K. Sim, Eds., vol. 9594. Springer, 2016, pp. 133–148.

[27] ——, “Evolving adaptive traffic signal controllers for a real scenario using genetic programming with an epigenetic mechanism,” in *16th IEEE International Conference on Machine Learning and Applications (ICMLA)*. IEEE, 2017, pp. 897–902.

[28] M. Péres, G. Ruiz, S. Nesmachnow, and A. C. Olivera, “Multiobjective evolutionary optimization of traffic flow and pollution in Montevideo, Uruguay,” *Applied Soft Computing*, vol. 70, pp. 472–485, 2018.

[29] M. Thielscher, “General game playing in AI research and education,” in *KI 2011: Advances in Artificial Intelligence*, ser. LNCS, J. Bach and S. Edelkamp, Eds., vol. 7006. Springer, 2011, pp. 26–37.

[30] P. H. Calais Guerra, A. Veloso, W. Meira Jr, and V. Almeida, “From bias to opinion: a transfer-learning approach to real-time sentiment analysis,” in *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 2011, pp. 150–158.

[31] C. B. Do and A. Y. Ng, “Transfer learning for text classification,” in *Advances in Neural Information Processing Systems*, 2006, pp. 299–306.

[32] E. Hajiramezanali, S. Z. Dadaneh, A. Karbalayghareh, M. Zhou, and X. Qian, “Bayesian multi-domain learning for cancer subtype discovery from next-generation sequencing count data,” in *Advances in Neural Information Processing Systems*, 2018, pp. 9115–9124.

[33] I. B. Arief-Ang, M. Hamilton, and F. D. Salim, “A scalable room occupancy prediction with transferable time series decomposition of CO2 sensor data,” *ACM Transactions on Sensor Networks (TOSN)*, vol. 14, no. 3-4, pp. 1–28, 2018.

[34] A. Arnold, R. Nallapati, and W. W. Cohen, “A comparative study of methods for transductive transfer learning,” in *Proceedings of the 7th IEEE International Conference on Data Mining Workshops*. IEEE, 2007, pp. 77–82.

[35] E. Haslam, B. Xue, and M. Zhang, “Further investigation on genetic programming with transfer learning for symbolic regression,” in *Proceedings of IEEE Congress on Evolutionary Computation (CEC 2016)*, Y.-S. Ong, Ed. IEEE, 2016, pp. 3598–3605.

[36] W. B. Langdon and J. P. Nordin, “Seeding genetic programming populations,” in *European Conference on Genetic Programming (EuroGP 2000)*, R. Poli, W. Banzhaf, W. B. Langdon, J. Miller, P. Nordin, and T. C. Fogarty, Eds. Springer, 2000, pp. 304–315.

[37] M. D. Schmidt and H. Lipson, “Incorporating expert knowledge in evolutionary search: a study of seeding methods,” in *Proceedings of the 11th Annual Conference on Genetic and Evolutionary Computation*. ACM, 2009, pp. 1091–1098.

[38] D. Jackson and A. P. Gibbons, “Layered learning in Boolean GP problems,” in *European Conference on Genetic Programming (EuroGP 2007)*, M. Ebner, M. O’Neill, A. Ekárt, L. Vanneschi, and A. I. Esparcia-Alcázar, Eds. Springer, 2007, pp. 148–159.

[39] T. T. H. Dinh, T. H. Chu, and Q. U. Nguyen, “Transfer learning in genetic programming,” in *2015 IEEE Congress on Evolutionary Computation (CEC 2015)*. IEEE, 2015, pp. 1145–1151.

[40] D. O’Neill, H. Al-Sahaf, B. Xue, and M. Zhang, “Common subtrees in related problems: A novel transfer learning approach for genetic programming,” in *2017 IEEE Congress on Evolutionary Computation (CEC 2017)*. IEEE, 2017, pp. 1287–1294.

[41] R. Santana, L. Marti, and M. Zhang, “GP-based methods for domain adaptation: using brain decoding across subjects as a test-case,” *Genetic Programming and Evolvable Machines*, vol. 20, no. 3, pp. 385–411, 2019.

[42] L. Muñoz, L. Trujillo, and S. Silva, “Transfer learning in constructive induction with genetic programming,” *Genetic Programming and Evolvable Machines*, Nov 2019.