From *Proceedings of the Fourth International Conference on Knowledge Discovery and Data Mining* (1998). New York: AAAI Press.

Learning to Predict the Duration of an Automobile Trip

Simon Handley
Pat Langley
Daimler-Benz Research and Technology Center
1510 Page Mill Rd., Palo Alto, CA 94304 USA
{handley,langley}@rtna.daimlerbenz.com

Folke A. Rauscher
Daimler-Benz AG
Research & Technology – FT3/KL
P.O. Box 2360, D-89013 Ulm, Germany
folke.rauscher@dbag.ulm.daimlerbenz.com

Abstract

In this paper, we explore the use of machine learning and data mining to improve the prediction of travel times in an automobile. We consider two formulations of this problem, one that involves predicting speeds at different stages along the route and another that relies on direct prediction of transit time. We focus on the second formulation, which we apply to data collected from the San Diego freeway system. We report experiments on these data with k-nearest neighbour combined with a wrapper to select useful features and normalization parameters. The results suggest that 3-nearest neighbour, when using information from freeway sensors, substantially outperforms predictions available from existing digital maps. Analyses also reveal some surprises about the usefulness of other features like the time and day of the trip.

Introduction

Future drivers will expect accurate estimates of the time it will take them to travel along a given route, both for purposes of route planning and for deciding when to start. Of course, existing digital maps provide such time estimates, but these are usually based on the speed limit for each route segment, and thus ignore many information sources that, potentially, could greatly improve accuracy.

Some cities are already collecting data on traffic speeds that could serve as the basis for more informed predictions. Typically, this information comes from sensors embedded in the freeway that measure average velocity and traffic density over short time spans. For example, the freeway system for San Diego, California, includes some 116 such sensors, the values for which are available on the World Wide Web.

Our aim is to use such data to discover useful knowledge about the behavior of traffic. Specifically, given an origin location O, a desired location D, a route from O to D, and possibly other information such as traffic load and starting time, we want to predict the time it will take to drive that route from O to D. The corresponding data-mining task involves learning such a predictor from available traffic data.

We hypothesize that techniques from machine learning, if carefully applied to such data, can improve the prediction of travel times over that are currently available from digital maps. However, as with previous applied efforts in machine learning and data mining (Langley & Simon, 1995), we expect this will require significant efforts in formulating the problem, in collecting and processing the data, and in crafting a representation. We present our responses to these issues in the sections that follow, along with our selection of an induction method. After this, we report experiments on data from San Diego freeways and a detailed analysis of our results. In closing, we discuss related research on predicing travel time and consider directions for future work on this topic.

Problem Formulation

Before we could make progress on our problem, we first had to reformulate it in some way that would let us apply existing techniques for machine learning and data mining. One straightforward way of viewing the problem was as a regression task in which the performance element takes as input attribute values that represent a trip and produces an estimate of that trip's duration. This formulation has the advantage of being simple and mapping readily onto well-understood induction techniques. However, treating trips as atomic entities would limit the ability to generalize between trips. For example, if we were to estimate that the trip A-B-C-D takes 10 minutes and that the trip B-C-D-E takes 15 minutes, then we would also like to estimate accurately the duration of the trip B-C-D. With this formulation, the knowledge learned about trips A-B-C-D and B-C-D-E does not carry over to the trip B-C-D.

We attempted to address this limitation by treating each trip not as a monolithic object but rather by considering the intermediate speeds that constitute them. That is, we tried redefining the task as that of predicting the speed of traffic at a particular time and a particular place, from which we could then compute the desired estimates of trip duration. This formula-

tion can be viewed as a form of time-series problem, in that each location has a speed, s_t, which is a function of the previous speeds, $s_{t-1}, \ldots, s_0$, as well as the data from other locations. In the time-series community, this approach is known as *conditional forecasting* (Pindyck & Rubinfeld, 1991), that is, forecasting that bases its predictions on other unknown variables that must also be predicted.

Unfortunately, preliminary experiments with this time-series formulation encountered difficulties, the reasons for which emerged after simple analysis. Consider the steps involved in calculating the duration of a trip from a collection of speed estimates. We assume that the route is divided into a series of segments and that we estimate the car's speed at the start of each such segment. These estimated speeds are functions of time and we compute the total duration by successively estimating the time at which the car enters each segment. However, the estimate for the car's speed when entering segment i derives from the sum of estimated times taken to traverse segments 0 through $i-1$, which in turn derive from the estimated speeds when entering segment 0 through $i-2$, etc. This routine is unstable in that errors in the speed estimations are magnified by the successive estimations, until the resultant estimate for trip duration has a much higher error than the speed estimates from which it is derived.

Our understanding about how errors are likely to propagate through this prediction method, coupled with the large errors in speed estimates, led us to conclude that the time-series formulation was not a promising approach for this domain. Consequently, we have focused on the simpler regression formulation that involves directly predicting the durations of trips.

Collecting and Processing the Data

Having formulated the problem, our next step was to collect and process traffic data. Because we lacked a large database of observed trip durations on which to test our system, we created one from available data on traffic speeds. We collected these data from 116 sensors, updated periodically, at fixed locations in the San Diego freeway system, that are available through the World Wide Web (`www.maxwell.com/caltrans/sd/sd_transnet.html`).

Each sensor reports four numbers every minute:[1] the 30-second average for traffic speed, the 360-second speed average, the 30-second average for flow, and the 360-second flow average. The distances between adjacent sensors ranged from 0.2 to 9.5 miles with a distribution of 1.6 ± 1.8 miles. We used an automated script to download these sensor readings from the web site over a period of 21 days (September 9, 1997, to October 1, 1997), resulting in about 1.3 million readings.

[1] For one sensor we examined, the actual time between readings was 61.1 ± 20.6 seconds, with a median of 60 and a minimum of 30.

We processed the sensor readings in three ways. First, if a sensor reported all zeros, we copied the speeds and flows from the previous sample for that sensor. Second, we could not determine accurately the location for one sensor (#148), so we discarded all readings for it. Finally, the data contained many entries in which a sensor reading at, say, '`11:58:00PM 09/12/97`' was followed by a reading at, say, '`12:00:30AM 09/12/97`'. The most likely cause of these anomalies was that the date was not being updated correctly, so we incremented the date for the second reading by one day.

Now that we had a database of average traffic speeds at certain locations and times, we proceeded to build a database of trips and durations. We started by enumerating all routes that began and ended at one of the 116 sensors. We then selected just the routes on Interstate 5 southbound, as this was both the longest stretch of freeway available and contained the most sensors. This instrumented section of Interstate 5 is approximately 25 miles in length, starts at Encinitas Boulevard, finishes at 6th Avenue, and contains 12 sensors. We then used available route-planning software to compute the distance between each pair of adjacent sensors. Finally, for each of these $12(12-1)/2 = 66$ routes we uniformly and randomly generated 100 start dates and times.

For each of the computed 6,600 trips, we next computed an 'observed' duration. Since each route is a contiguous subsequence of the 12 sensors on Interstate 5, $S_1 \ldots S_n$, we determined the speed of an average car driving that route and then the duration of the trip from those speeds. Each sensor S_i has a speed, $s_i(t)$, that is a function of date and time. From these we estimated the time, t_i, that the car took to drive from sensor S_i to sensor S_{i+1} by choosing $n-1$ pairs of time and speed, $(t_i, \hat{s}_i)$, that together minimize $\sum error(i)$ where

$$error(i) = |\hat{s}_i - s_i(t_0 + \sum_{j<i} t_j)|$$

and t_0 is the start date and time. The duration of the entire trip is just $\sum t_i$.

We discarded a trip if it required sensor readings outside of the selected dates or if it coincided with a gap of more than ten minutes in the readings for one or more of the required sensors. We replaced each such discarded trip with another trip so that, after discarding malformed trips, there were still 6,600 in total.

Feature Selection and Induction

Having defined the performance task (estimating the duration of an individual trip), as well as the collection and processing of the data, we next chose an induction algorithm and applied it in this context. We selected the k nearest neighbour method because of its algorithmic simplicity and because our intuitions suggested it was a good match for this problem. However, nearest neighbour is actually a class of algorithms, and we needed to make other decisions before we could apply it to our problem. For example, we decided to set the

number of neighbours k to 3, since preliminary studies suggested that higher k values did not aid prediction.

The choice of features was less easy, since a number of them had intuitive appeal. Features that were readily available from our data included the time of day, the day of week, whether it was a weekday or the weekend, and the current sensor values. The latter were really a collection of attributes, one for each sensor on the route, reflecting the 30-second average for traffic speed that the sensor reported at the date and time when the trip started. One representational complication was that different routes could have different numbers of associated sensors. Thus, we modeled each of the 66 routes with a separate set of stored cases, which limited generalization but ensured an unambiguous feature mapping.

Finally, nearest neighbour is often combined with a *normalization* scheme that maps the values of numeric attributes into a common range; the idea here is to prevent some features from dominating others in the distance metric through accidental choices like differences in measurement scale. Because we did not know whether normalization would aid learning in this domain, we considered three alternatives: no normalization, mapping the instance space onto a unit cube, and transforming each feature to have a mean of zero and variance of one.

The choices about predictive features and normalization schemes give $2^4 \times 3 = 48$ combinations of model parameters. Rather than exploring this space manually, we automated the process by using ten-fold cross validation to estimate the performance for each combination of parameters, but ruling out parameter combinations that made no sense. For example, if there is only one feature then normalizations have no effect. This eliminated 11 possibilities, resulting in 37 parameter settings for 3-nearest neighbour.

The Control Predictor

We decided to compare the behavior of our learned predictors with that of a control predictor that uses speed information available from digital maps to estimate a trip's duration. This information takes the form of a single 'typical' speed for each road segment. This control predictor operates in a manner similar to the route planners currently used by in-car navigational devices, except that it does not use all of the available speed annotations in the digital map.

The control predictor only uses the speeds that are attached to road segments adjacent to the sensors. For example, if a route consists of road segments A-B-C-D-E-F-G and the sensors in the road are located at segments A, D and G, then it uses only the speeds attached to segments A, D, and G. Although we could probably improve the control method's performance by including more road segments (such as B, D, E, and F in the above example), this also holds for the learning method and would only weaken the comparison.

This scheme has a number of advantages as a control condition. The method is conceptually simple and it differs from the learning methods only in ways that are important for the comparison (the incorporation of learning). Moreover, it is sufficiently similar to current trip duration predictors to permit meaningful and relevant comparisons.

Experimental Evaluation

In order to compare our trip duration estimators experimentally, we needed some way to measure their performance. Since the estimators would be used by humans to plan trips, we desired some measure of their usefulness to drivers. Of course, this will depend on both the driver and the trip, but lacking real drivers and real trips, we needed to make some simplifying assumptions. In particular, we assumed that the cost of errors was independent of the driver and that overprediction and underprediction had equal costs. We also assumed that relative error (in relation to the actual trip time) was more appropriate than absolute error.

However, we expected that drivers would care not only about the average error, but also about its variation. To this end, we decided to use a single measure that bounds the relative error from above 84.1% of the time. That is, we defined our performance measure to be $|\mu| + \sigma$, where μ is the mean relative error and σ is the standard deviation. Hereafter, we refer to this performance metric as the *mean+sigma* or $\mu+\sigma$ bound.

We estimated the $\mu+\sigma$ bounds of the predictors by ten-fold cross validation on the 6,600 trips described earlier. For the cross validation to be useful, we needed to guarantee that each trip in the training set did not overlap (in either space or time) any trip in the test set and vice versa. We implemented this constraint using a greedy algorithm, which rejected about 25% of the examples normally used during cross validation. We replaced each discarded trip with another one, to keep the same number of trips in each partition.

The mean percentage error for the control predictor, calculated over the ten cross-validation runs, was 22.8 and the standard deviation was 19.5. This translates to a $\mu+\sigma$ bound of $22.8+19.5 = 42.3$, which means that, on 84.1% of novel trips, the control method will have a relative error of less than 42.3%. To facilitate comparisons, we used the same ten test sets for the wrapper-extended nearest neighbour method. This produced a mean percentage error of only 2.2, with a standard deviation of 4.8. This results in a $\mu+\sigma$ bound of $2.2 + 4.8 = 7.0$, which means that, on 84.1% of novel trips, the learned predictor will be off by less than 7.0 percent. In this domain, nearest neighbour gives much tighter bounds on travel time than speeds available from digital maps.

However, examination of the learning method's outputs revealed some unexpected behavior. In seven of the ten folds, the wrapper chose not to normalize the predictor variables, it incorporated day of the week in only two folds, and it never elected to use information about the time of day. In contrast, the system decided to include current sensor information about traffic conditions in all ten folds of the study. Although we ex-

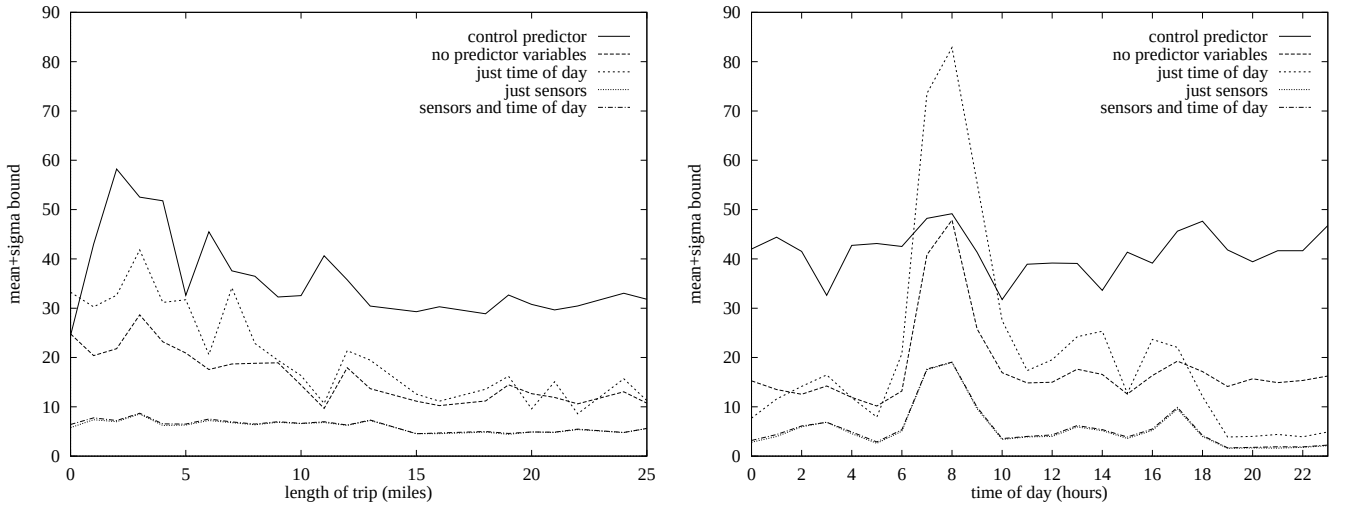

Figure 1: Errors for the control predictor and for four representative sets of model parameters, broken down by (a) the length of the trip and (b) the time of day at which the trip started.

pected online traffic data to prove useful, we assumed that information about the time (e.g., whether it was rush hour) and day (whether it was a weekend) would also have predictive power. These findings suggested the need for further analysis.

Analysis of the Experiments

To better understand the factors that led to the above results, we reexamined the predictive accuracies at a finer level of granularity. Also, instead of using the wrapper scheme, we selected manually some parameter settings of interest, then used ten-fold cross validation on the 6,600 examples to estimate the $\mu+\sigma$ bound for each one. In addition to the control method, we examined nearest neighbour using only readings from the traffic sensors to describe instances, only the time of day, and both sensor and time information. We also included a variant of nearest neighbour that used *none* of the predictor variables; since this viewed all stored cases as equidistant from the test case, it used the average trip duration (for each route) as its prediction.

One plausible hypothesis was that the attribute for time of day might prove more useful on longer trips. Figure 1 (a) shows the behavior of the five prediction methods, in terms of their $\mu+\sigma$ bounds, broken down by the trip length. All of the predictors, including the control method, fare better on longer trips, but sensor information about traffic still dominates even on trips of 25 miles. Moreover, including the time of day provides no predictive ability above that available from the sensor readings alone, independent of trip length.

Figure 1 (b) presents similar results, except that it partitions behavior by the time of day. For this particular stretch of freeway (southbound on Interstate 5 to San Diego), the primary rush hour occurs in the morning. Surprisingly, the control method's performance is nearly constant: its $\mu+\sigma$ bound hovers between 30 and 50 throughout the day, with no obvious pattern. Because the control predictor's estimates are independent of time, we would expect it to be inaccurate at rush hour, but the learned predictors are much more affected. One likely explanation is that the control predictor consistently over-estimates trip times, improving its accuracy at rush hour at the expense of other times.

In fact, the learned predictor that uses just time of day does spectacularly badly during rush hour, with a $\mu+\sigma$ bound of 70 to 90 between 7 and 9 AM. The other three learned predictors consistently have $\mu+\sigma$ bounds of 5 to 20 outside rush hour and 20 to 50 during rush hour. One reason that time of day fares so poorly may be that speed during rush hour is inherently unpredictable. In our 6,600-trip data set, the mean duration is about 500 ± 400 seconds, but the mean during rush hour (7 to 9 AM) is about 650 ± 550 seconds. Since the duration for rush-hour trips has a higher variance than for others, a prediction based on these trips will also have higher variance. Using traffic sensors sidesteps this problem by basing predictions on trips with similar driving conditions, rather than similar times of day.

Related Work on Predicting Trip Times

There exists considerable literature in the area of traffic management and intelligent highways, some of which pertains to predicting the duration of trips. We focus on the most closely related work here, although work in other areas of data mining, such as regression for financial prediction, also bears on our approach.

Oda (1990) reports one effort, which we discovered after obtaining our results, that predicts travel times using an approach similar to our time-series formulation. He compared his method's predictions to observed travel times on a single stretch of freeway, achieving slightly lower mean error than the best of our predictors, but he did not report the variance of his method. In later work, Oda, Takeuchi, and Niikura (1996) report results in which they predicted trip times using

'sensor' information uploaded from cars driving along the route. In this case, the prediction errors had both a low mean and a low variance, suggesting that current speed information from other cars is highly useful.

Three other efforts also have similar research goals. Hoffmann and Janko (1990) describe a system that learns the average speed per road segment in Berlin for one of four time periods (such as morning rush hour), but they did not report their predictor's performance. Taylor and Meldrum (1995) used learning in multi-layer neural networks to predict the traffic volume at an individual sensor; their work is similar to our own except for its focus on volume rather than travel time. Finally, Fu and Rilett (1995) used learning in neural networks to improve prediction of trip times in an artificial environment; their formulation was very similar to ours, using features like the origin, the destination, and the start time as the basis for predictions.

Concluding Remarks

Although we have made clear progress toward better predictors for trip duration, there remain many avenues for improvement. Recall that we estimated our travel-time data from readings of highway sensors, rather than from measured durations of individual trips. In order to test our approach on actual trip times, we are collecting data from automobiles in Silicon Valley that are equipped with a global positioning system, which lets us compute their time on each segment of a route. This will also let us predict trip durations for a given driver, which should be more accurate than predictions for an average driver. The availability of detailed trace data will also let us incorporate additional features from digital maps, such as the presence of intersections and road topology, in the learned predictors.

Other directions for future research focus on different formulations of the problem. Our breakdown of prediction errors by time of day showed that all the learning methods, even when using sensor readings, did worse during rush hour than during other times. The fact that traffic speed is less predictable during rush hour suggests that we also try predicting traffic volume. This variable may be less erratic during rush hour, and it can also play a role in route planning, since drivers typically prefer to avoid routes with high congestion.

Also, to date we have assumed that the prediction task occurs just before the user intends to start a trip, since one major use for trip duration estimates is within automated route planners. However, making predictions further in the future can also useful. For example, a driver may want to know in advance how long a trip to the airport will take, so he can plan when to pack. Changing the time of the prediction task should reduce the predictive utility of traffic sensors. The worst-case scenario is given by the 'just time of day' curve in Figure 1 (b), which shows that performance without sensors is two to five times worse than when using sensors. A model of the degradation in predictive accuracy with increasingly delayed start times would be useful.

Another limitation is that our approach to learning duration predictors does not handle long-term trends. Suppose the same trip at 9 AM Monday morning on consecutive weeks takes ever increasing amounts of time; our current use of nearest neighbour will miss this trend, since it will average the durations across different weeks. One response would be to formulate the problem differently, so as to incorporate information about trips on previous weeks; this scheme bears a resemblance to methods used in the financial prediction community.

In summary, we have explored the use of induction methods to predict the duration of trips in an automobile. In the process, we built a database of trips and their durations from actual traffic speeds on San Diego freeways, and we tested our learned predictors against a control method that uses speeds encoded in digital maps. Our experiments revealed that the learned predictors were generally better than the control predictor, in that they place tighter bounds on the prediction errors. We also noted that the most useful features involved traffic speeds available from sensors along for the route; surprisingly, the time of day and day of week were much less useful. Finally, we found that longer trips support more accurate predictions and that rush hour is less predictable than other times of day. Although there remains room for improvement, the existing results show the promise of machine learning for predicting the duration of automobile trips.

Acknowledgements

We thank Jerome Friedman, David Moriarty, and Seth Rogers for useful discussions that helped us formulate the approach we have reported in this paper.

References

Fu, L., & Rilett, L. R. (1995). Dynamic O-D travel time estimation using an artificial neural network. *Proceedings of the Vehicle Navigation & Information Systems Conference* (pp. 236–242). Seattle: IEEE Press.

Hoffmann, G., & Janko, J. (1990). Travel times as a basic part of the LISB guidance strategy. *Proceedings of the International Conference on Road Traffic Control* (pp. 6–10). London: IEE.

Langley, P., & Simon, H. A. (1995). Applications of machine learning and rule induction. *Communications of the ACM, 38*, 55–64.

Oda, T. (1990). An algorithm for prediction of travel time using vehicle sensor data. *Proceedings of the International Conference on Road Traffic Control* (pp. 40–44). London: IEE.

Oda, T., Takeuchi, K., & Niikura, S. (1996). Travel time measurement using infrared vehicle detectors. *Proceedings of the International Conference on Road Traffic Monitoring & Control* (pp. 178–182). London: IEE.

Pindyck, R. S., & Rubinfeld, D. L. (1991). *Econometric models and economic forecasts* (3rd edition). New York: McGraw-Hill.

Taylor, C., & Meldrum, D. (1995). Freeway traffic data prediction using neural networks. *Proceedings of the Vehicle Navigation & Information Systems Conference* (pp. 225–230). Seattle: IEEE Press.