Modeling Highway Traffic Volumes

Tomáš Šingliar and Miloš Hauskrecht

Computer Science Dept, University of Pittsburgh, Pittsburgh, PA 15260
{tomas,milos}@cs.pitt.edu

Abstract. Most traffic management and optimization tasks, such as accident detection or optimal vehicle routing, require an ability to adequately model, reason about and predict irregular and stochastic behavior. Our goal is to create a probabilistic model of traffic flows on highway networks that is realistic from the point of applications and at the same time supports efficient learning and inference. We study several multivariate probabilistic models and analyze their respective strengths. To balance accuracy and efficiency, we propose a novel learning model, mixture of Gaussian trees, and show its advantages in learning and inference. All models are evaluated on real-world traffic flow data from highways of the Pittsburgh area.

1 Introduction

The importance of road transportation systems to our daily lives can hardly be overstated. To facilitate monitoring and management of their complexities, sensors collecting traffic information (such as traffic volumes and speeds) are installed on many roads. Today, coverage is good for major highways in many metropolitan areas and traffic sensor deployment is rapidly increasing worldwide.

Road networks exhibit strong interaction patterns among traffic variables and traffic is subject to stochastic fluctuations. The ability to adequately model, reason about and predict stochastic behavior is crucial to computational support of many traffic management tasks, such as traffic routing, congestion analysis and accident detection. The objective of our work is to develop models of large multivariate continuous probability distributions describing vehicular traffic. We require that the models be compactly parameterized and admit efficient inference and learning algorithms.

The quantities of primary interest in this paper are traffic flows. Traffic flow is defined as the number of vehicles passing a point on a highway in a unit of time. Traffic flows in the highway network are typically modeled with Gaussian densities [1]. An assumption very often made in stochastic analysis of a network system is that the components of the system behave independently. The multivariate probabilistic model of the network then factorizes to a product of univariate distributions that are easy to learn and reason with. Unfortunately, the assumption of full independence is unrealistic and ignores strong interaction patterns between traffic variables observable in real data. On the other hand, the full-covariance model is too complex to learn reliably from limited data.

Consequently, we seek models between the two extremes that provide the right balance between model complexity and accuracy. The ideal model captures the important dependencies and at the same time remains tractable.

Intuitively, vehicles on distant roadways do not interact and the dependency patterns must be "local" and closely tied to the topology of the physical road network. In this work we study two models that attempt to capture only the key local interactions: the *conditional autoregressive (CAR) model* [2], and our novel approach with *mixture of Gaussian trees.*

We analyze and compare all models on real-world traffic data collected at 75 sensor locations on major highways in the Pittsburgh metropolitan region. We demonstrate that the new model represents the sought middle ground for Pittsburgh traffic network data.

2   Gaussian Models

The Gaussian probability distribution appears to be particularly suitable for modeling traffic volumes [1]. The number of cars passing during a given interval can be thought of as a result of a stochastic process in which drivers choose to take the particular segment with a probability, resulting in a binomial distribution of the number of observed cars. The binomial distribution, for large N, is well approximated by the computationally favorable Gaussian. The multivariate Gaussian model is characterized by its mean $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$. The parameters are usually learned from observed traffic data using maximum-likelihood estimates. The estimate quality depends on the number N of data-points available. The number of free parameters in the covariance matrix grows quadratically with the number of sensors, adversely impacting the variance of the parameter estimates.

The model-complexity problem of the full multivariate Gaussian model is often avoided by assuming that all variables are independent, i.e. the covariance matrix is restricted to be diagonal. As a result, the learning problem decomposes to D univariate learning problems, where D is the dimensionality of the data. The advantage of the approach is that the number of free parameters is linear in the number of sensors. The drawback is that ignoring all interactions is unrealistic for traffic networks that exhibit strong correlation between readings of neighboring sensors.

3   The Conditional Autoregressive Model

Traffic flows at places not directly adjacent will not influence each other except via the situation on a road segment(s) connecting them. In other words, the Markov property [3] holds. The popular conditional autoregressive (CAR) model [2] embodies this intuition about locality. The model assumes that the volume y observed at location s obeys:

$$y(s) = \epsilon_s + \sum_{r \in N(s)} \theta_r^s y(r), \tag{1}$$

where $N(s)$ denotes the neighborhood of s and $\epsilon_s \sim \mathcal{N}(0, \sigma_s^2)$ is additive noise. We fit the model parameters $\boldsymbol{\theta}$ using a ridge-regression procedure [4].

The limitation of the CAR model is that the conditional probabilities need not define a proper probabilistic model [5]. Even if it exists, the joint distribution may be intractable and the most natural way to obtain proper probabilities from the CAR model is Gibbs sampling [6].

4   Mixture of Gaussian Trees

Bayesian networks [7] are an elegant formalism for capturing probabilistic dependencies. A Bayesian network consists of a directed *acyclic* graph and a probability specification. In the directed graph, each node corresponds to a random variable, while edges define the decomposition of the represented joint probability distribution:

$$p(X) = \prod_{i=1}^{D} p(x_i | pa(x_i)),$$

where $pa(X_i)$ are the parents of X_i in the graph. The probability specification is the set of conditional distributions $p(X_i | pa(X_i))$, $i = 1, \ldots, D$. Intuitively (but not exactly), an edge in a Bayesian network represents a causal influence of the parent on the child. However, traffic congestions are subject to cyclic interaction patterns (e.g., gridlocking) that cannot be directly modeled with a Bayesian network.

One way to address the problem of cyclic interactions is to approximate the underlying distribution with a simpler dependence structure that permits both efficient learning and inference. Having the maximum number of edges without introducing cycles, tree structures are a natural choice. By committing to a single tree, we capture the maximum number of dependencies without introducing a cycle; but inevitably, some dependencies will be ignored. In the *mixture of trees* model [8], the missed dependencies may be accounted for by the remaining mixture components.

Meilă developed the model in a discrete variable setting. In this work we propose the *mixture of Gaussian trees (MGT)* model where Gaussian instead of discrete variables are used. The mixture of Gaussian trees consists of:

- a collection of m trees with identical vertex sets $T_1 = (X, E_1), \ldots, T_m = (X, E_m)$, where each node $v \in V_k$ with parent x_u has a conditional probability function such that $x_v \sim \mathcal{N}(\mu + c_v x_u, \sigma_v)$. Note that it is always possible to orient a tree so that each node has at most one parent.
- mixture weights $\boldsymbol{\lambda} = (\lambda_1, \ldots, \lambda_m)$ such that $\sum_{k=1}^{m} \lambda_i = 1$.

Let $T_k(\mathbf{x})$ denote the probability of $\mathbf{x}$ under the distribution implied by the tree Bayesian network T_k. The joint probability for the mixture model is then:

$$p(x) = \sum_{k=1}^{m} \lambda_k T_k(x). \tag{2}$$

4.1   Inference in the MGT Model

Any probabilistic query in the form $p(\mathbf{y}|\mathbf{e})$ can be answered from the joint distribution (Equation 2) by taking:

$$p(\mathbf{y}|\mathbf{e}) = \frac{p(\mathbf{y}, \mathbf{e})}{p(\mathbf{e})} = \frac{\sum_i \lambda_i T_i(\mathbf{y}, \mathbf{e})}{\sum_i \lambda_i T_i(\mathbf{e})} \tag{3}$$

Both the numerator and denominator represent m instances of inference in tree Gaussian networks, which is a linear-complexity problem [9].

4.2   Learning in the MGT Model

The maximum likelihood parameters for the MGT model can be obtained by the EM algorithm. Three quantities must be estimated in each M-step: (1) the structure of trees that constitute the mixture components, (2) their parameterization and (3) the mixture proportions. Let $\gamma_k(n)$ denote the posterior mixture proportion:

$$\gamma_k(n) = \frac{\lambda_k T_k(\mathbf{x}^n)}{\sum_i \lambda_i T_i(\mathbf{x}^n)}.$$

The $\gamma_k(n)$s can be interpreted as "responsibility" of tree k for the n-th datapoint. Computing $\gamma_k(n)$s constitutes the E-step. The quantity $\Gamma_k = \sum_{n=1}^{N} \gamma_k(n)$ takes on the meaning of the expected count of datapoints that T_k is responsible for generating. Let us also define the distribution P_k associated with T_k over the set of datapoints by $P_k(x^n) = \frac{\gamma_k(n)}{\Gamma_k}$.

In the M-step, we need to update three quantities: the tree structures, their parameters and the mixture proportions.

The **tree structures** are selected using a variant of the Chow-Liu procedure [10]. The Chow-Liu procedure selects a tree model T such that the KL-divergence (or equivalently, the cross-entropy) between the responsibilities computed in the E-step and the tree distribution is minimized:

$$T_k^{new} = \underset{T_k}{\mathrm{argmax}} \sum_{i=1}^{N} P_k(x^i) \log T_k(x^i). \tag{4}$$

This is accomplished by finding a Maximum Weight Spanning Tree (MWST), where the edges are weighted by the mutual information between variables they connect. The structure update for the tree T_k requires that we compute the mutual information between all variables $x_u, x_v \in X$. In the continuous case, this is computationally infeasible without making a distributional assumption. We therefore treat $P_k(x^i)$ as a sample from a Gaussian and compute the mutual information in closed form:

$$I_k(x_u, x_v) = -\frac{1}{2} \log(|\hat{\Sigma}_k|/(\sigma_u^2 \sigma_v^2)), \tag{5}$$

where $\hat{\Sigma}_k$ is the maximum likelihood estimate of the 2×2 covariance matrix and σ_u^2 and σ_v^2 are its diagonal elements. After we have determined the optimal

structure, we orient the tree by picking a vertex at random and directing all the edges away from it. In this manner we achieve that every vertex has at most one parent. Mutual information is symmetrical, which means that any orientation of edges yields an optimal spanning tree.

Parameter learning. It is unsurprising to derive that the M-step update for λ is to match the expected empirical marginal: $\lambda_k = \frac{\Gamma_k}{N}$.

Consider an arc $u \rightarrow v$ and recall that $x_v \sim N(\mu_v + c_v x_u, \sigma_v)$. We have data in the form $D_{uv} = \{(x_u^{(i)}, x_v^{(i)}, w^{(i)})\}_{i=1}^N$, where the weight $w^{(i)}$ corresponds to $P_k(x^i)$ computed in the E-step. We can update the parameters of v, denoted $\theta_v = \{\mu_v, c_v, \sigma_v\}$, by maximizing the likelihood of the data $P(D_{uv}|\theta_v)$. We obtain that the update of μ_v and c_v is the solution of the following linear system:

$$
\left(
\begin{array}{cc|c}
\sum\limits_{n=1}^{N} w_v^{(i)} & \sum\limits_{i=1}^{N} x_u^{(i)} w_v^{(i)} & \sum\limits_{i=1}^{N} x_v^{(i)} w_v^{(i)} \\
\sum\limits_{i=1}^{N} x_u^{(i)} w_v^{(i)} & \sum\limits_{i=1}^{N} x_u^{(i)} x_u^{(i)} w_v^{(i)} & \sum\limits_{i=1}^{N} x_v^{(i)} x_u^{(i)} w_v^{(i)}
\end{array}
\right).
\tag{6}
$$

Knowing μ_v and c_v we can estimate σ^2:

$$
\sigma_v^2 = \left(\sum_{n=1}^{N} w_v^{(i)} \right)^{-1} \sum_{i=1}^{N} (x_v^{(i)} - \mu_v - c_v x_u^{(i)})^2 w_v^{(i)}
\tag{7}
$$

E- and M- step are alternated until the expected complete log-likelihood stabilizes.

Model selection. The parameter that remains to be chosen is the number of mixture components (trees). We propose that the search be performed by learning the model with increasing number of components until the Bayesian Information Criterion (BIC) no longer decreases. The BIC is defined as an approximation to the integrated likelihood [11]:

$$
BIC(k) = -2 \ln p(\mathcal{D}|k, \hat{\theta}_k) + \psi_k \ln N
\tag{8}
$$

where $\hat{\theta}_k$ is the ML estimate of parameters and ψ_k is the number of free parameters in model with k components.

5   Experimental Evaluation

The data was collected by 75 sensors monitoring Pittsburgh highways. Each datapoint is thus a vector consisting of the numbers of vehicles passing the respective sensors during a five-minute interval. The dataset contains all measurements at a fixed time of all workdays throughout one year. The correlations between sensors are high and we expect that this will be a challenging dataset for the structure search algorithms, causing them to capture spurious correlations.

5.1   Evaluation Metrics

In order to assess the quality of distribution modeling, we use three metrics: a log-likelihood score, relative error and coefficient of determination. The complexity of the model is accounted for by also reporting the BIC score obtained in training of the model. The data is divided into the training and testing set. After the model is trained on the training set, some variables in each datapoint of the testing set are chosen to be hidden; they will be denoted by $h^{(n)}$, while the remaining – observed variables will be denoted by $e^{(n)}$. Denote the set of hidden variables by H.

We compute the log-likelihood score

$$LLS(H|\theta_M) = \sum_{n=1}^{N} \log p(h^{(n)}|e^{(n)}, \theta_M) \tag{9}$$

This score reflects how well the model predicts the set of chosen values, given the remaining observations. As this measure is computed on a subset of the unseen test set and the sample space of observables in all models is the same, it is not skewed by the different complexity of the models.

The coefficient of determination is a classical measure of predictor quality and can be interpreted as the proportion of the data variance explained by the model. It is obtained as $1 - RSS/TSS$. Denoting the actually observed value of the hidden variable h by $x(h)$ and the model's prediction by $y(h)$, the residual sum of squares is $RSS = \sum_{h \in H}(x(h) - y(h))^2$. The prediction given by model M is defined to be the mean of $p(h^{(n)}|e^{(n)}, M, \theta_M)$. The total sum of squares is defined as $TSS = \sum_{h \in H}(y(h) - E(y(h)))^2$.

The relative error is defined naturally as $e_{rel} = |x(h) - y(h)|/x(h)$.

We argue that multivariate metrics such as the LLS, which considers all hidden values in a particular datapoint jointly, reflect the model prediction quality better and should be given more weight than the univariate scores such as the coefficient of determination, which look at each missing value in isolation.

5.2   Experiment Setup and Parameters

The product of univariate Gaussians is learned using the ML estimate of mean and variance for each dimension separately. Conditioning is trivial for the model: $p(h^{(n)}|e^{(n)}, \mu, \sigma) = p(h^{(n)}|\mu, \sigma)$.

The full covariance Gaussians are parameterized from the data by the maximum likelihood estimates. Conditionals are obtained as $p(h^{(n)}|e^{(n)} = f, \mu, \Sigma) = \mathcal{N}(\overline{\mu}, \overline{\Sigma})$, where $\overline{\mu} = \mu_h - \Sigma_{he}\Sigma_{ee}^{-1}(\mu_e - f)$, $\overline{\Sigma} = \Sigma_{hh} - \Sigma_{he}\Sigma_{ee}^{-1}\Sigma_{eh}$ and $\Sigma_{..}$ are the respective block submatrices of Σ.

Since the CAR model may not define a proper distribution, we obtain a large number (10^6) of samples with the Gibbs sampler and fit a multivariate Gaussian to compute the likelihood.

Table 1. The likelihood scores (larger is better), and BIC scores (smaller is better), relative errors (smaller is better) and coefficients of determination (larger is better). The parenthesized numbers are the standard deviations across test splits.

Method	# prms	LLS	BIC	Relat err	Coef of det
$N(\mu, \Sigma)$	2,925	$-8,448(664.2)$	186,910 (6,751)	0.039(0.0022)	0.889(0.012)
$N(\mu, \mathrm{diag}(\sigma))$	150	$-13,314(1,036.3)$	197,910 (7,361)	0.073(0.0050)	0.638(0.019)
CAR	1,277	$-8,162(598.3)$	203,126 (5,970)	0.037(0.0023)	0.868(0.016)
SingleTree	224	$-8,282(616.6)$	13,667 (784.1)	0.057(0.0056)	0.765(0.050)
MixTrees(2)	449	$-8,176(638.6)$	17,159 (4,299)	0.053(0.0050)	0.766(0.052)
MixTrees(3)	674	$-8,158(632.0)$	24,562 (12,995)	0.055(0.0141)	0.704(0.176)
MixTrees(5)	1,124	$-8,226(624.2)$	67,787 (32,787)	0.197(0.4567)	0.305(0.341)

We learn the Mixture of Gaussian Trees (MGT) model using the EM algorithm described in Section 4, using 1,2,3 and 5 mixture components. The LL score is computed by conditional inference as described in Section 4.1.

In the reported experiment, 20% of the values are omitted at random from the testing set; the values omitted are fixed across the methods so that each method encounters the same set of missing data. This ensures comparability of the quality metrics across the methods. The statistics from 20 train/test splits are shown in Table 1.

5.3   Results

Evaluation results show that the mixture-of-trees model performed the best. The 3-component MGT yielded the best score, closely followed by the conditional autoregressive model. However, the MT model achieves this performance with much fewer parameters. The difference is reflected in the BIC complexity penalty. The BIC suggests that even a single-tree model might be appropriate, although the likelihood is lower for mixtures of 2 and 3 trees. Further in favor of the mixture model, the MGT model also has an embedded structure-learning component, while the CAR model operates with informed pre-defined neighborhoods. Therefore MGT achieves this performance with less prior information. MGT is very good at modeling the training data, yielding low BIC scores. This can lead to some amount of overfitting: the 5-component MGT shows signs of testing performance deterioration.

The relative error results confirm our original intuition that the MGT stands between the full and independent Gaussian in performance and complexity.

We note the disproportionately high BIC scores of the full-Gaussian, independent Gaussian and CAR models. In the independent Gaussian case, this is caused by poor modeling of the dependencies in data. On the other hand, the full Gaussian and CAR models suffer a high complexity penalty.

This cautions us that normally we do not have the data to fit a full covariance Gaussian. A variety of factors is observable in traffic networks, of which the most obvious is the variability with the time of day. The correct way to deal with

observable factors is to condition on them, which cuts down the training data severely. The full covariance Gaussian will likely meet scaling-up problems in such context.

6   Conclusions

We developed and presented a new mixture of Gaussian trees model that provides a middle ground in between the data-hungry full covariance and the unrealistic all-independent Gaussian models. We have explored several other methods for modeling traffic density and used a predictive likelihood measure to compare their performance. If data are plentiful, the full-covariance Gaussian can be estimated with a high accuracy. However, in the more realistic case when data is scarce, our mixture-of-trees model, with a number of parameters that increases linearly with the dimension of the dataset, offers itself as the method of choice.

Many interesting research issues remain open. For example, when learning from small sample-size datasets, it would be advantageous to generalize the Bayesian version of the MT learning algorithm [12] to handle the distributions in the exponential family (including Gaussians that we used here). Automatic model complexity selection and greater resistance to overfit are among the expected benefits of applying the Bayesian framework.

References

1. Belomestny, D., Jentsch, V., Schreckenberg, M.: Completion and continuation of nonlinear traffic time series: a probabilistic approach. Journal of Physics A: Math. Gen. 36, 11369–11383 (2003)
2. Besag, J., York, J., Mollie, A.: Bayesian Image Restoration With Two Applications In Spatial Statistics. Annals of the Institute of Statistical Mathematics 43(1), 1–59 (1991)
3. Lauritzen, S.L.: Graphical Models. Oxford University Press (1996)
4. Hastie, T., Tibshirani, R., Friedman, J.: Elements of Statistical Learning. Springer, Heidelberg (2001)
5. Chellappa, R., Jain, A. (eds.): Markov Random Fields - Theory and Applications. Academic Press, London (1993)
6. Doucet, A., de Freitas, N., Gordon, N.: Sequential Monte Carlo Methods in Practice. Springer, New York (2001)
7. Jensen, F.V.: An Introduction to Bayesian Networks. Springer, New York (1996)
8. Meilă-Predoviciu, M.: Learning with mixtures of trees. PhD thesis, MIT (1999)
9. Shachter, R., Kenley, R.: Gaussian influence diagrams. Management Science 35(5), 527–550 (1989)
10. Chow, C.J.K., Liu, C.N.: Approximating discrete probability distributions with dependence trees. IEEE Trans. on Inf. Theory 14(3), 462–467 (1968)
11. Schwarz, G.: Estimating the dimension of a model. Annals of Statistics 6, 461–464 (1978)
12. Meilă, M., Jaakkola, T.: Tractable Bayesian learning of tree belief networks. Technical Report CMU–RI–TR–00–15, Carnegie Mellon University Robotics Institute (2000)