Anomaly Detection on Spatiotemporal Graphs of Sparse Traffic Data

Harris Hardiman-Mostow, Tufts University

Supervised by Dr. James Murphy, Tufts University Department of Mathematics

With Marshall Mueller (Tufts University) and Opemipo Esan (Pennsylvania State University)

Abstract

Our challenge was to perform anomaly detection on sparsely sampled traffic data. The data, provided by the NSF, was collected hourly over a two-year period at 500 traffic sensors in an unnamed city. Using the provided information – including traffic flow, location, hour, weekday, month, and year – we must determine when traffic "anomalies" occur, which the NSF defines as a traffic flow observation three or more standard deviations from the mean flow for a particular station, day, and hour. However, the data we would like to analyze, is sparse – it is missing most of the observations of traffic flow. Hence, to perform anomaly detection, we must first estimate these missing data points.

To obtain our estimation, we use a novel algorithm which leverages spatiotemporal data, smoothing functions[2], as well as our natural intuition about traffic patterns. We also use a relatively recent algorithm, K-SVD[3], combined with the data science technique of bootstrap aggregating, or "bagging", to refine our estimation. Once we have a complete estimation of every data point, we can run the anomaly detection program that is provided by the NSF. We achieved .701 F1 Score – an evaluative metric for anomaly detection routines – on our training data.

Introduction

Anomaly detection[1] is a subfield of machine learning that is focused on identifying when data deviates from normality, where "normal" is defined in a precise mathematical sense. In our case, a traffic anomaly may occur due to a traffic jam or holiday, for example. Applications are wide-ranging, including in medicine, cybersecurity, and banking. In this project, we investigate anomaly detections methods with multiple sets of traffic data collected and distributed by the NSF. Our challenge was to identify when a traffic "anomaly" occurs within the data.

The main challenge presented by this NSF data was that it was purposefully incomplete – traffic sensors recorded hourly traffic flow data for two years, but only one to twenty percent of the data was given to us by the NSF. This was to simulate a common issue that occurs in data science: sparse data. Hence, our goal of detecting anomalies will be predicated on estimating these missing data points.

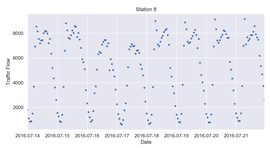

Figure 1: A complete data set example

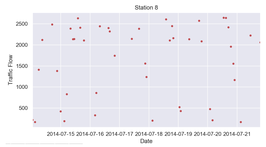

Figure 2: The sparse data we would like to analyze

A traffic anomaly is defined in the following way by the NSF: "For a given sensor s, hour h, and weekday w, the n^{th} observation of traffic flow $d^{(n)}_{(shw)}$ is anomalous if

$$\left| d^{(n)}_{shw} - \mu_{shw} \right| \geq 3\sigma_{shw}$$

Put into words, a traffic observation is anomalous if it is three or more standard deviations from the mean for that station, hour, and day.

Data

We are provided a "training" data set, called "City 1", which has all available data and classification as an anomaly or not:

	Timestamp	ID	TotalFlow	Year	Month	Day	Hour	Weekday	Latitude	Longitude	Anomaly	Fraction_Observed	Observed
0	2016-01-01 00:00:00	0	1058.0	2016	1	1	0	Friday	0.175793	0.167569	True	0.05	False
1	2016-01-01 01:00:00	0	853.0	2016	1	1	1	Friday	0.175793	0.167569	True	0.05	True
2	2016-01-01 02:00:00	0	631.0	2016	1	1	2	Friday	0.175793	0.167569	False	0.05	False
3	2016-01-01 03:00:00	0	502.0	2016	1	1	3	Friday	0.175793	0.167569	False	0.05	False
4	2016-01-01 04:00:00	0	353.0	2016	1	1	4	Friday	0.175793	0.167569	False	0.05	False
...	...	...	...	...	...	...	...	...	...	...	...	...	...
8770995	2017-12-31 19:00:00	782	469.0	2017	12	31	19	Sunday	0.604337	0.891310	False	0.10	False
8770996	2017-12-31 20:00:00	782	429.0	2017	12	31	20	Sunday	0.604337	0.891310	False	0.10	False
8770997	2017-12-31 21:00:00	782	303.0	2017	12	31	21	Sunday	0.604337	0.891310	False	0.10	False
8770998	2017-12-31 22:00:00	782	214.0	2017	12	31	22	Sunday	0.604337	0.891310	False	0.10	False
8770999	2017-12-31 23:00:00	782	139.0	2017	12	31	23	Sunday	0.604337	0.891310	False	0.10	False

We would like to classify anomalies on two "testing" data sets, "City 2" and "City 3" which is missing most data points:

	Timestamp	ID	TotalFlow	Year	Month	Day	Hour	Weekday	Latitude	Longitude	Fraction_Observed	Observed
20	2014-01-01 20:00:00	4	1640.0	2014	1	1	20	Wednesday	0.170318	0.563544	0.01	True
72	2014-01-04 00:00:00	4	964.0	2014	1	4	0	Saturday	0.170318	0.563544	0.01	True
168	2014-01-08 00:00:00	4	608.0	2014	1	8	0	Wednesday	0.170318	0.563544	0.01	True
199	2014-01-09 07:00:00	4	2394.0	2014	1	9	7	Thursday	0.170318	0.563544	0.01	True
212	2014-01-09 20:00:00	4	2868.0	2014	1	9	20	Thursday	0.170318	0.563544	0.01	True
...	...	...	...	...	...	...	...	...	...	...	...	...
8752484	2015-12-31 08:00:00	1509	3210.0	2015	12	31	8	Thursday	0.618732	0.241567	0.10	True
8752486	2015-12-31 10:00:00	1509	2707.0	2015	12	31	10	Thursday	0.618732	0.241567	0.10	True
8752495	2015-12-31 19:00:00	1509	2717.0	2015	12	31	19	Thursday	0.618732	0.241567	0.10	True
8752496	2015-12-31 20:00:00	1509	2139.0	2015	12	31	20	Thursday	0.618732	0.241567	0.10	True
8752497	2015-12-31 21:00:00	1509	1907.0	2015	12	31	21	Thursday	0.618732	0.241567	0.10	True

Methodology and Results

We want to estimate the missing data points:

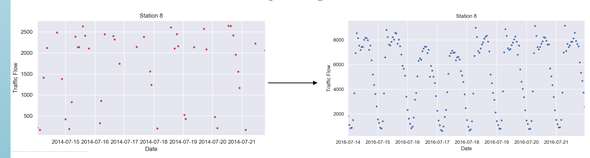

Figure 3: An illustration of what we are given, and where we would like to go!

From here, we can construct the following image, which allows us to explicitly classify anomalies:

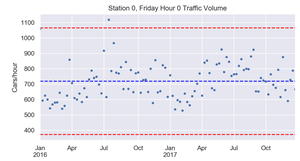

Figure 4: Anomaly detection plot. Note the anomaly on July 4.

In Figure 3, the blue line is the mean for this particular station, day, and hour, while the red signifies the three standard deviation boundary.

To estimate these missing points, we incorporated several methods. Foremost among them was a "Gaussian Kernel", which allows us to estimate missing points using a weighted average of nearby, known data points. The weight assigned to a traffic flow observation x in order to predict a traffic flow observation y is described by:

$$k(x,y) = \exp\left(-\frac{\|x-y\|^2}{2\sigma^2}\right)$$

Methodology and Results (continued)

We also applied this same idea across weekly and daily periodicity. To illustrate this, suppose we are trying to estimate a traffic flow observation on a Tuesday at 8 a.m. Then observations at 7 or 9 a.m. the same day, the next or previous day at 8 a.m., or next Tuesday at 8 a.m. would get this most weight – if we know those points. These ideas contributed to our initial algorithm.

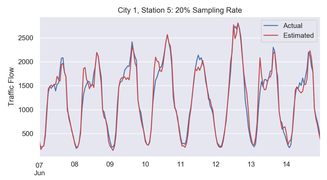

Figure 5: Actual Traffic Flow vs. Estimated Flow via initial algorithm.

Our evaluative metric is the F1 score, which is calculated as follows; *tp, fp,* and *fn* mean True Positive, False Positive, and False Negative, respectively:

$$F_1 = \frac{\mathrm{t}p}{\mathrm{t}p + \frac{1}{2}(\mathrm{f}p + \mathrm{f}n)}$$

The initial algorithm yielded an F1 score of .662, which was an improvement over the .437 baseline algorithm provided by the NSF.

Next, we wanted to refine our algorithm using a robust method called K-SVD[3]. K-SVD can detect underlining patterns even when data is sparse, such as in this project. In fact, the algorithm generated the following "dictionary" of traffic flow signals that could be used to reconstruct the traffic flow from the sparse data we are given.

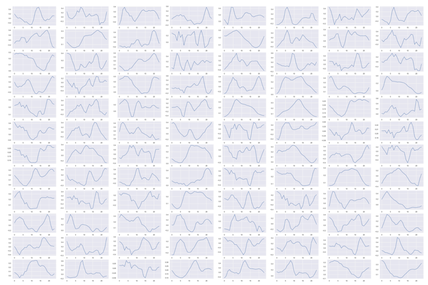

Figure 6: K-SVD Dictionary of Traffic Flow Signals

Using this Dictionary, we reconstructed the flow at each of the 500 stations using aggregated averages of weekly flow patterns. From here, we could classify anomalies in terms of their absolute or relative distance from the K-SVD's reconstructed average. Then, these classifications were compared to the original algorithm's, and the three methods (K-SVD absolute, relative, and original algorithm) "voted" on whether each individual point was an anomaly or not for the final output. This is associated with a common data science practice called "bootstrap aggregating." The following is the output table submitted to the NSF for evaluation of our performance on City 1.

Fraction_Observed	True Positive	False Positive	True Negative	False Negative	Precision	Recall	F1	Accuracy	Support
overall	5421	2018	656850	2611	0.728727	0.674925	0.700795	0.993059	8032
0.01	99	94	17279	128	0.512953	0.436123	0.471429	0.987386	227
0.02	205	140	34521	234	0.594203	0.466970	0.522959	0.989345	439
0.05	545	347	86375	533	0.610987	0.505566	0.553299	0.989977	1078
0.1	1406	606	172653	835	0.698807	0.627398	0.661180	0.991789	2241
0.2	3166	831	346022	881	0.792094	0.782308	0.787171	0.995121	4047

Conclusion

Using a spatiotemporal-based algorithm, a carefully selected kernel weighting scheme, KSVD dictionary learning, and bootstrap aggregating, we were able to estimate missing data in a large set of sparse traffic data. From these estimations, we could perform the NSF-provided anomaly detection routine to classify data points as anomalous or not anomalous. Our model, trained on the City 1 dataset, returned .701 F1 score - a metric which reflects the algorithm's ability to detect anomalies – compared to .437 F1 score of the baseline detector provided by the NSF. The NSF has yet to provide the complete labeled data for our testing data, so we do not know the exact metrics of our performance in this case. However, we do know that we finished 4$^{\text{th}}$ overall nationally among other research teams.

The results of this project demonstrate several techniques which are applicable to any setting where sparse data is present and anomalies must be identified. For example, medical imaging is an area with active anomaly detection research, which has been leveraged to detect tumors with image processing algorithms of images of patients' organs captured by powerful cameras. However, these are expensive and difficult to collect in large datasets – hence, analyzing traffic data provides an easy way to experiment and draw conclusions about anomaly detection methods.

References

1. Chandola, V., A. Banerjee, and V. Kumar. "Anomaly Detection: A Survey." *ACM Computing Surveys.* 2009.
2. Bretherton, C. "Interpolation and Smoothing." 2015.
3. Aheron, M., M. Elad, and A. Bruckstein. "K-SVD: An Algorithm for Designing Overcomplete Dictionaries for Sparse Representation." *IEEE Transactions On Signal Processing*, Vol. 54, No. 11. 2006.
4. Mairal, J., M. Elad, and G. Sapiro. "Sparse representation for color image restoration." *IEEE Transactions on Image Processing* 17, no. 1 (2007): 53-69.
5. Zelnik-Manor, K., and P. Perona. "Self-Tuning Spectral Clustering." *NIPS Proceedings.* 2005.
6. Olshausen, B., and K.J. Millman. "Learning Sparse Codes with a Mixture-of-Gaussians Prior." 1999.

Acknowledgements

I would like to thank Professor James Murphy, for his mentorship and guidance. I would also like to thank Marshall and Ope, for being intelligent and hard-working collaborators, and to Dr. Anne Moore, for organizing the Summer Scholars programming (even in the middle of pandemic!).

Finally, thank you to the Summer Scholars Program and thank you to the T-Tripods Institute (NSF HDR Grant 1934553) for partial support of this work. The traffic data collection and challenge was funded by NSF DMS 1924513.