Bayesian Outlier Detection with Inlier Rate Estimation

Enmao Diao

Introduction

We introduce a Bayesian inference mechanism for outlier detection using Inlier Rate Estimation proposed by [1]. Outliers are detected by forming a point estimate from samples of candidate transformation (models) T. We formulate and implement our method into Expectation-Maximization (EM) framework. Outlier detection methods like K-nearest-Neighbors (KNN), support vector machine (SVM) are widely used in machine learning society. Correlation-based techniques are widely used in high dimensional data. In Bayesian framework, people typically treat outlier detection problem as a mixture model. Bayesian outlier detection usually have a hidden variable $\boldsymbol{z}$ labeling each data sample. Fully Bayesian framework usually results in a Dirichlet Process Model (DPM) and people extend this model to different types of stochastic process models [2][3]. In a pseudo-Bayesian framework, people tend to use EM to optimize their parametric models along with the hidden variables $\boldsymbol{z}$. Typically, at E step, we sample weight $\boldsymbol{\alpha_k}$ from a Dirichlet distribution and $p(\boldsymbol{z_k}|\boldsymbol{y}, \boldsymbol{\theta_k}, \boldsymbol{\alpha_k})$ follows a Dirichlet-Categorical distribution. If we have N data and k number of mixtures then we would have a $N \times k$ weight matrix with each row sum to 1. Note that, typically, each row is sampled from the same prior. In outlier detection settings, typically we will only have $\mathrm{k} = 2$ and thus $\boldsymbol{\alpha_k}$ follows a beta distribution. At M step, we will condition on the weight matrix and optimize our parameters $\boldsymbol{\theta_k}$. After iteratively conduct these two steps, EM is guaranteed to converge to a local optimal. In fact, EM is a special case of Majorize-Minimization (MM) algorithm. This iterative optimization method exploits the convexity of a function in order to find their maxima or minima. The conditional expectation that we optimize in M step is a convex function and we use it to majorize the global object function where $\boldsymbol{z}$ are involved. Our idea is that, instead of sampling $\boldsymbol{z}$ and optimize θ afterward, why not try to optimize $\boldsymbol{z}$ based on samples from θ?

Background

Before we going to the details of our method, we want to first introduce the paper that inspires me. Their main focus is to perform outlier detection after interest point detection and matching. First, they introduce a grid based sampling method to sample candidate transformations between two images. Typically, this method cut an unbounded continuous data space into a bounded gridded data space. Then they generate candidate transformations by enumerate 4-point matching inside the boundary. However, there is memory space issue, since the number possible of enumerations can be extremely large. In the end, they have to compile their code in C to alleviate this problem. After enumerating grid-based data, and generating candidate models, they construct an error matrix $\boldsymbol{E}$ by computing the loss of each sample against each model. Typically, they will have N data samples and 10^6 transformations. The error matrix $\boldsymbol{E}$ is therefore a $N \times 10^6$ matrix. Then they sort each column of $\boldsymbol{E}$. Note that the ordering of data is disrupted. The sorted matrix is used to estimate inlier rate following Algorithm 1. After IRE, they use the inlier rate to select the best model based on a discrete and combinatorial optimization method called Branch and Bound (BnB). BnB is guaranteed to converge to the global optimal and its convergence speed relies on estimating upper and lower bounds on the optimal solution. It is the most widely used discrete optimization method for NP-hard problem. Choosing BnB to select the best model is logical since they have to select from their discrete sampled transformations.

However, several issues hinder their method to generalize beyond estimating projectivity transformation. First, although they cut data space in to grid, in fact, they are grid-based Monte Carlo sampling from the parameter space. If we mesh-grid the nine parameters in H, we will have very similar results. The only problem is memory. Mesh-griding in a two-dimensional space only alleviates the issue but the method also fails to generalize to high dimensional data and the case where parameters cannot be properly estimated with a small number of data. Second, instead of enumerating transformations, we should be able to perform IRE with a small number of

transformations if those transformations are properly sampled. A natural solution is to use Monte Carlo Markov Chain (MCMC). With MCMC, we can sample continuous parametric models efficiently and effectively, since our samples will mostly clustered around the Maximum Likelihood Estimate (MLE) estimate. Finally, we also want to select the best model while we are performing outlier detection with IRE. A natural way of constructing this optimization problem is to use the EM algorithm. It is obvious that our approach is the inverted version of the classical method.

1. Construct a sorted error matrix E from MCMC samples
2. Calculate a column vector r_{min} by taking the minimal error from each row of E.
3. Calculate a column vector $v_\epsilon(p)$ by counting the number of entries in each row of E that are at most larger than their respective value in $r_{min} + \epsilon$.
4. Take $\hat{p}$ to be the relative location of the minimal value in $v_\epsilon(p)$.

Algorithm 1. Inlier Rate Estimation (IRE) [1]

Method

We formulate our ideas under Bayesian Linear regression model. With MCMC, we first draw σ^2, and then we draw $\boldsymbol{\mu}$ conditionally. After iteratively draw these samples, we can construct the sorted error matrix $\boldsymbol{E}$ based on those sample. Here you can view these samples as candidate transformations. Following Algorithm 1., we can perform IRE and outlier detection. To detect outliers, we select Maximum A Posteriori (MAP) as our best model. In this case, we use Posterior Mean as our best model. Then we simply test data against this model. Outliers are those least-fitted samples beyond the inlier rate $\hat{p}$. Then we go back to E step and condition on the data of which outliers are removed.

E-step

- MCMC Sample $\boldsymbol{\beta}$ according to following Bayesian linear regression model,

$$
\begin{aligned}
p(y|X,\beta,\sigma^2) &\propto (\sigma^2)^{-\frac{n}{2}} \exp(-\frac{1}{2\sigma^2}(y-X\beta)^T(y-X\beta)) \\
p(\beta,\sigma^2|y,X) &\propto p(\beta|\sigma^2,y,X)p(\sigma^2|,y,X) \\
\beta|\sigma^2,y,X &\sim \mathcal{N}((X^TX)^{-1}X^Ty,\sigma^2(X^TX)^{-1}) \\
\sigma^2|\beta,y,X &\sim \text{Inv-Gamma}(a_n,b_n) \\
a_n &= a_0+\frac{n}{2} \\
b_n &= b_0+\frac{(y-X\beta)^T(y-X\beta)}{2}
\end{aligned}
$$

Here we assuming a flat prior on $\boldsymbol{\beta}$ and an Inverse-Gamma prior on σ^2

M-step

- Construct error matrix $\boldsymbol{E}$ from MCMC samples
- Sort each column of $\boldsymbol{E}$
- Estimate $\hat{p}$ from Algorithm 1.
- Select Posterior Mean as our best model
- Test data samples against best model and extract inliers with $\hat{p}$
- Construct new design matrix from inliers and go to E-step

In the classical settings, recall that we have a $N \times k$ weight matrix. In practice, all data samples share the same prior distribution and thus they are always equally likely to be considered as outliers. Then the weight matrix becomes a $1 \times k$ weight vector. Obviously, in the outlier detection problem, $\mathrm{k} = 2$. And the only free parameter left is just the inlier rate or outlier rate. Therefore, $\hat{p}$ is essentially the same as $\frac{\boldsymbol{\alpha_0}}{\boldsymbol{\alpha_0+\alpha_1}}$ or $\frac{\boldsymbol{\alpha_1}}{\boldsymbol{\alpha_0+\alpha_1}}$ in the classical settings. However, to our best knowledge, there does exist any closed form analytical solution for this ratio. Based on the theoretical justification of IRE shown in [1], we may need to use probability theory to derive certain point estimator for this weight matrix.

Results

We test our algorithm on one toy example and one real data. The toy example is a line-fitting problem and we take the data from the image of the homework. Therefore, we do not know the ground truth data generating process. We compare our method to RANSAC to show that our method can have comparable result. Part of the results are listed in Figure 1. As iteration grows we gradually converge to the optimal. Note that EM algorithm only guarantees local optimal. Thus, it is possible to get stuck at certain point. However, we can gradually decrease ϵ in Algorithm 1. Here you can view ϵ as a radius of tolerance. Decreasing ϵ will push $\hat{p}$ toward the relative location of r_{min}. This is also a suggested procedure in [1].

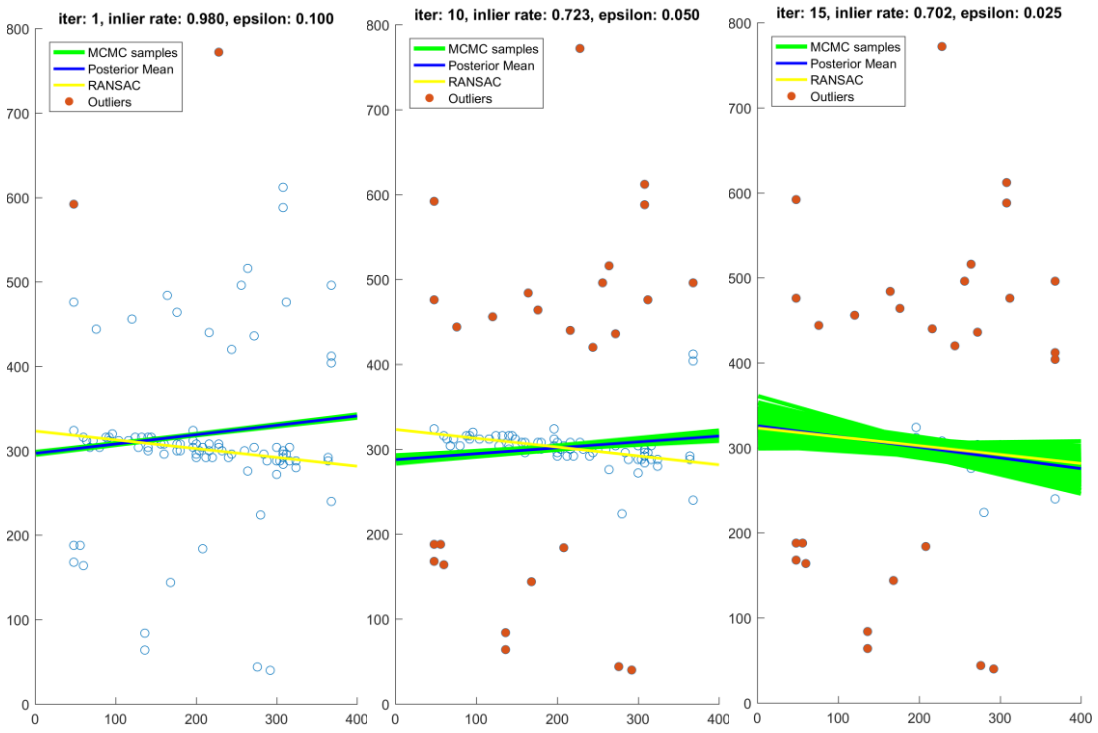

Figure 1. Fit a line with extreme outliers.

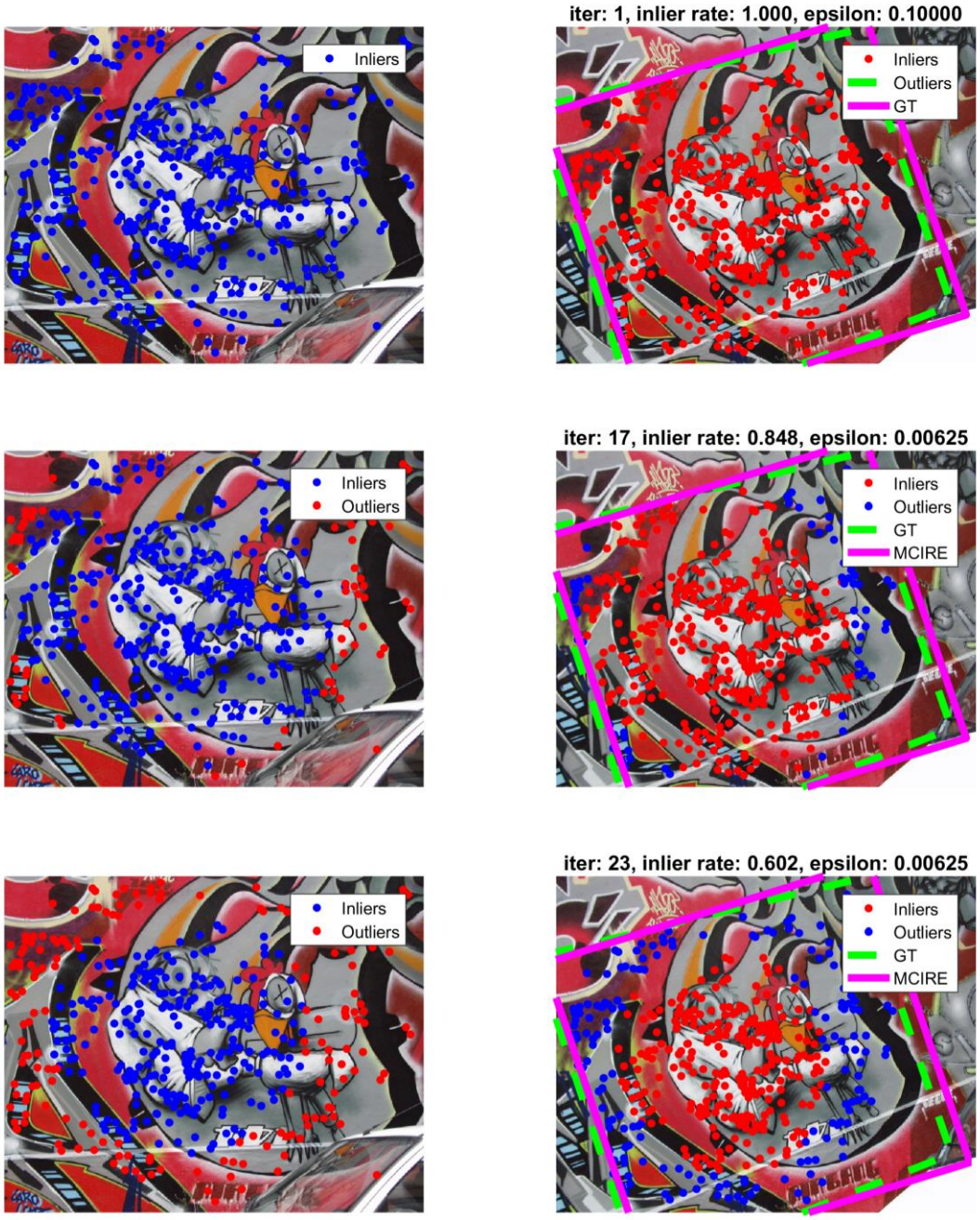

Figure 2. Results for image pair 1 and 2.

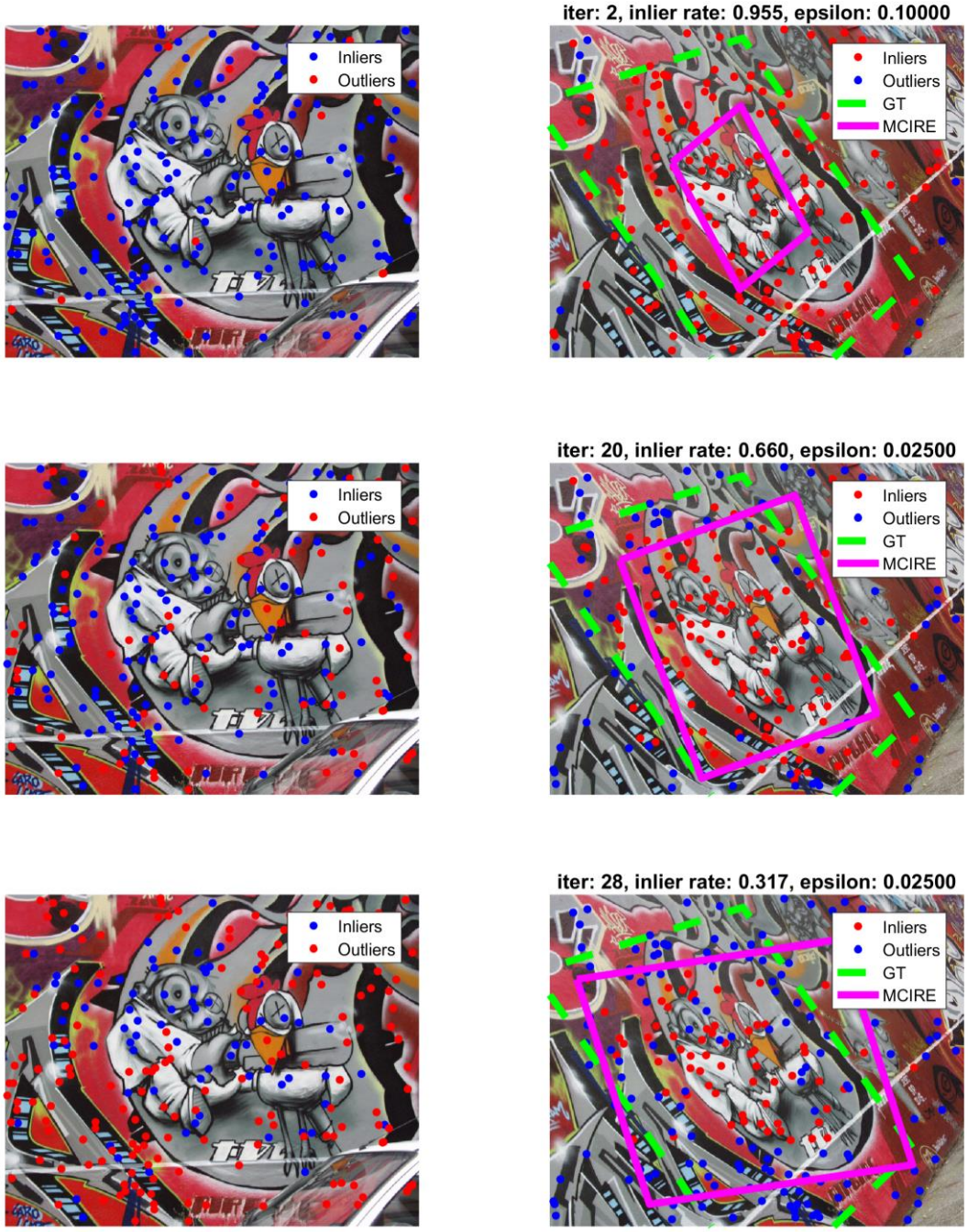

Figure 3. Results for image pair 1 and 4.

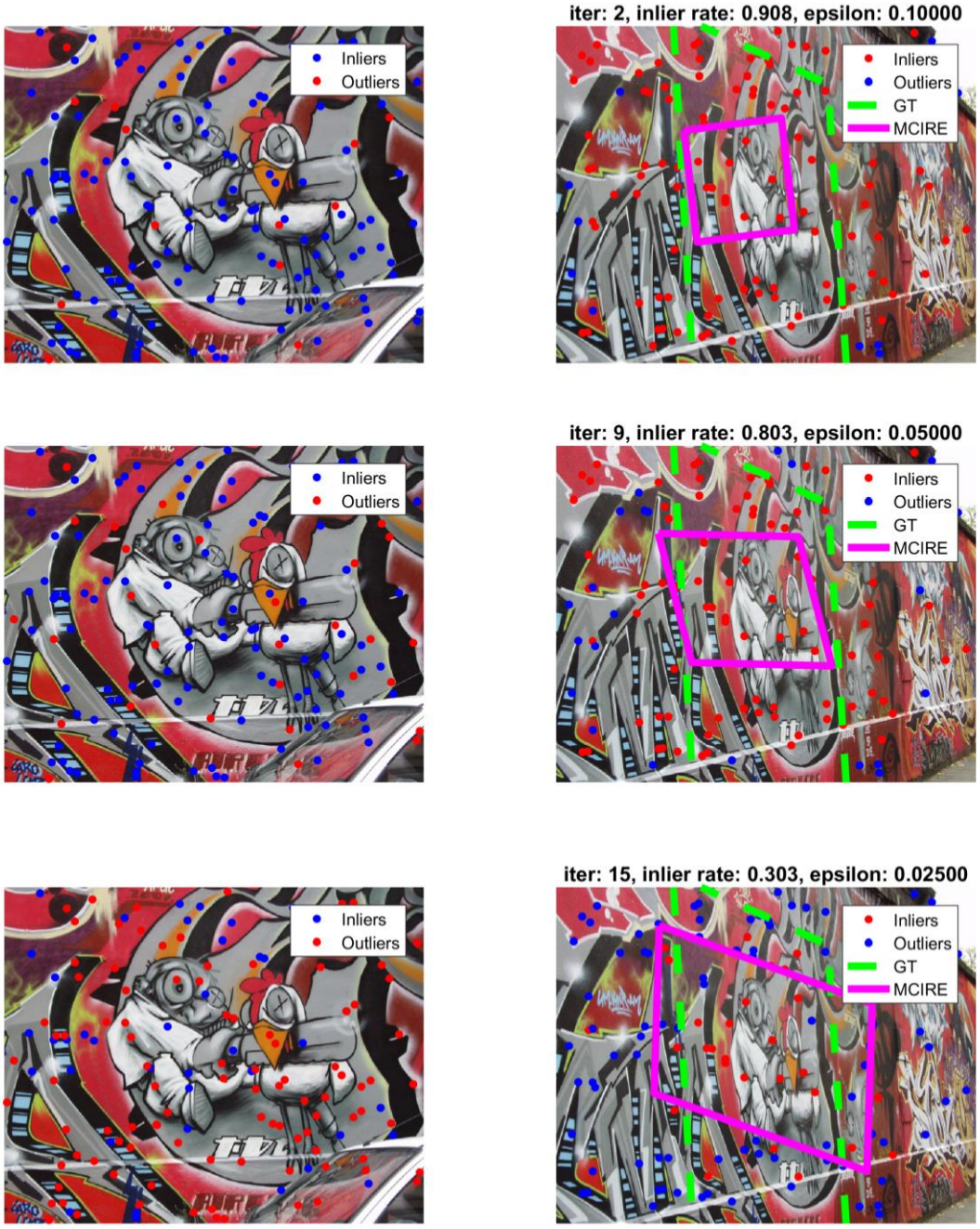

Figure 4. Results for image pair 1 and 6.

We also test our algorithm on real data presented by Mikolajczyk etal. [4]. They are originally constructed to benchmark interest point detectors and descriptors. We use the pairs 1-2, 1-4, 1-6, for which a ground-truth homography is provided. The results are shown in Figure 2-4. First, we use VL_Feat to extract Scale-invariant feature transform (SIFT) and matched interest points. Typically, we have less than 500 points and we

sample around 5000 candidate transformations.

As you can see from the results, there are several issues we fail to address. First, we do not specify a theoretically justified convergence condition. For now, we check if ϵ decreases below a threshold. However, the algorithm will fail to converge if ϵ is too small and many inliers are mislabeled as outliers. Since we do not have an analytical solution at M step, it becomes ambiguous to determine the convergence condition. Second, at each iteration in M step, the outliers are determined by hard-thresholding and are totally removed from the future iterations. This will accelerate convergence but we are also less likely to converge to the global optimum as the data sample size increases. One potential fix is that we can put different Dirichlet prior on different data samples. At M step, those that are labeled as outliers will add one more counts, meaning $\boldsymbol{a_1}$. Then at E step, we can sample from the updated prior and then sampled from Categorical distribution. In this way, $\frac{\boldsymbol{\alpha_0}}{\boldsymbol{\alpha_0+a_1}}$ of each data sample should converge to their inlier probability. By random sampling subset of data, we can create more candidate transformations conditioning on those samples. Actually, we are doing Bootstrapping and thus we can somewhat correct the bias of $\widehat{\boldsymbol{\beta}}$ and $\hat{p}$.

Discussion and Future work

The reason I am interested in this project is that I believe this approach can be generalized to a much broader framework. Furthermore, I really want to develop a light weight outlier detection algorithm. So far, most outlier detection algorithm acts like a prescreening stage and they are typically not feasible for big data. I am really interested in the idea that estimate inlier rate by sampling or generating candidate models. We are actually optimizing the model parameters and hidden parameters together. In the classical settings, imagine that you have 10^4 data, and doing an outlier detection problem. Typically, we will have 10^4 parameters to sample in order to treat each data sample independently. I believe it is not computationally feasible to sample those parameters and I do not want to retreat to the case where every data sample are treated

equally likely. Instead of interval estimate, we may be able to derive some point estimators of those parameters. So far, IRE can only be used for the case where every data is treated equally likely, but I think it is a good starting point to address this issue. Furthermore, since now we only need to sample $\boldsymbol{\theta}$, we may be able to alleviate the notorious degeneracy issue in mixture model.

Conclusion

We generalize from [1] and propose a novel Bayesian outlier detection algorithm under EM framework. Instead of sampling hidden variable $\boldsymbol{z}$, and then optimize model parameters $\boldsymbol{\theta}$, we propose first sample $\boldsymbol{\theta}$, and then estimate inlier rate $\hat{p}$. $\hat{p}$ here is essentially the same as $\frac{\boldsymbol{\alpha_0}}{\boldsymbol{\alpha_0+\alpha_1}}$ or $\frac{\boldsymbol{\alpha_1}}{\boldsymbol{\alpha_0+\alpha_1}}$. This novel approach inverts the classical mixture model method. Our simulation result shows its potential. But we are still lack of solid theoretical justification. Things like convergence condition and consistency of estimators are still not clear. We would like to further investigate into this problem and hopefully we can generalize it to a light weight outlier detection framework.

Reference

[1] Litman, Roee, et al. "Inverting RANSAC: Global model detection via inlier rate estimation." Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2015.

[2] Chaloner, Kathryn, and Rollin Brant. "A Bayesian approach to outlier detection and residual analysis." Biometrika 75.4 (1988): 651-659.

[3] Shotwell, Matthew S., and Elizabeth H. Slate. "Bayesian outlier detection with dirichlet process mixtures." Bayesian Analysis 6.4 (2011): 665-690.

[4] K. Mikolajczyk and C. Schmid. A performance evaluation of local descriptors. Pattern Analysis and Machine Intelligence, IEEE Transactions on, 27(10):1615–1630, 2005. 1, 7, 8