Opprentice: Towards Practical and Automatic Anomaly Detection Through Machine Learning

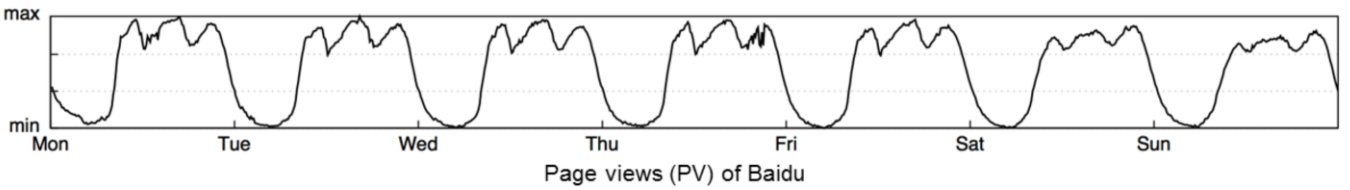
Dapeng Liu, Youjian Zhao, Haowen Xu, Yongqian Sun, Dan Pei, Jiao Luo, Xiaowei Jing, Mei Feng

Thank you for the introduction. Hello everyone, I'm Dapeng from Tsinghua University

Today, I' going to talk about Opprentice, a practical and automatic KPI anomaly detection framework based on machine learning

This is a joint work with collaborators from Tsinghua University, Baidu, and PetroChina

KPIs and Anomaly Detection



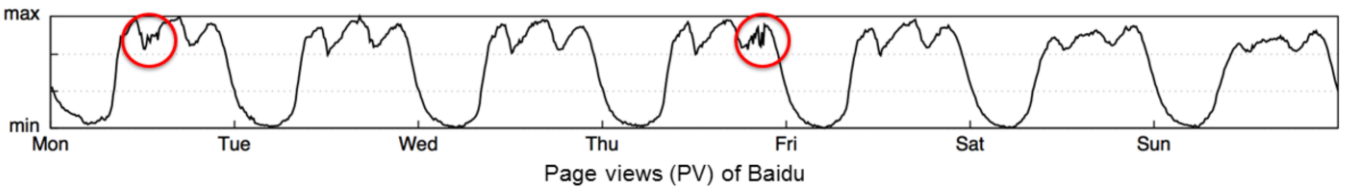
Page views (PV) of Baidu

KPIs (Key Performance Indicators): A set of performance measures that evaluate the service quality

To monitor the service quality, service providers often collect a set of performance measures, which are usually called key performance indicators or KPIs.

For example, this is one-week data of the page views of Baidu. It measures how many search queries are submitted from users to Baidu. This KPI is very important because it has a great influence on the advertising revenue.

KPIs and Anomaly Detection



Page views (PV) of Baidu

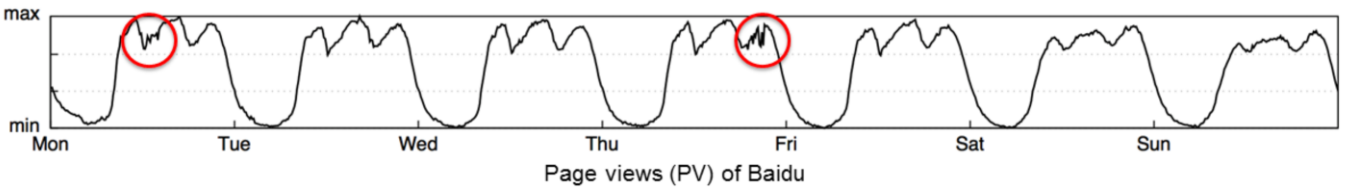
KPIs (Key Performance Indicators): A set of performance measures that evaluate the service quality

KPI anomalous (unexpected) behaviors → Potential failures, bugs, attacks...

We see that beyond those regular behaviors, the KPI can show some anomalous or unexpected behaviors.

These behaviors often indicate potential failures, bugs, attacks and so on

KPIs and Anomaly Detection



Page views (PV) of Baidu

KPIs (Key Performance Indicators): A set of performance measures that evaluate the service quality

KPI anomalous (unexpected) behaviors → Potential failures, bugs, attacks...

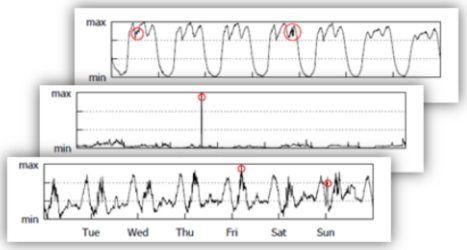
Anomaly detection matters: Find anomalous behaviors of the KPI curve
→ Diagnose and fix it
→ Avoid further influences and revenue losses

As a result, anomaly detection really matters for service providers. In this work, the anomaly detection means to find anomalous behaviors of the KPI curve.

** Click
Then we can diagnose whether these behaviors are really caused by a problem. If so, we should fix it to avoid further influences or revenue losses.

How to Build the Anomaly Detection System

Domain experts (Operators)
- Responsible for the KPIs
- Knowing the KPI behaviors well



Developers
- Building the detection system
- Knowing several anomaly detectors

Simple threshold

Historical Average

Wavelet

Holt-Winters

…

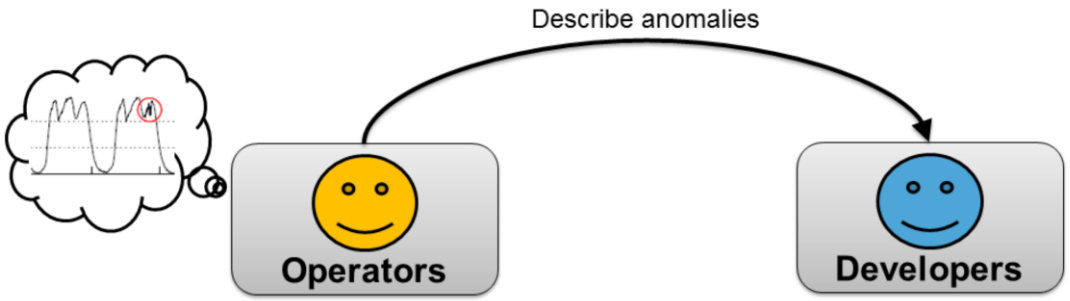
Now the question is how can we build an anomaly detection system?

**Click
Overall, there are two characters involved.
The first one is domain expert. #Explain…

**Click
The second one is developers. #Explain…

How to Build the Anomaly Detection System

In practice, it is more complex

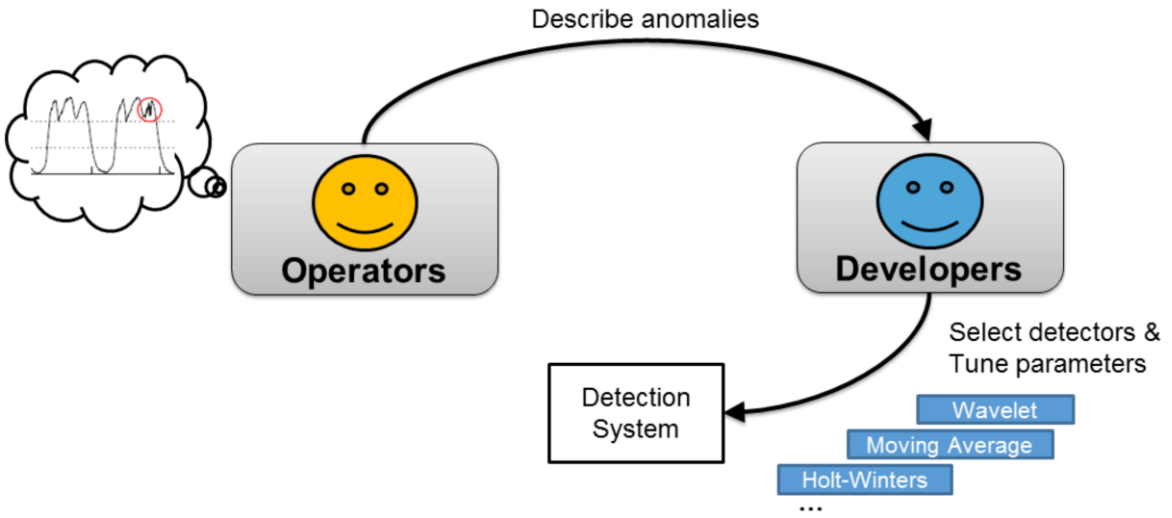


However, in practice, building an anomaly detection system is more complex.

** Click
First, operators are the users of the anomaly detection system, and they need to tell developers how anomalies look like? or what kinds of anomalies they care?

How to Build the Anomaly Detection System

In practice, it is more complex

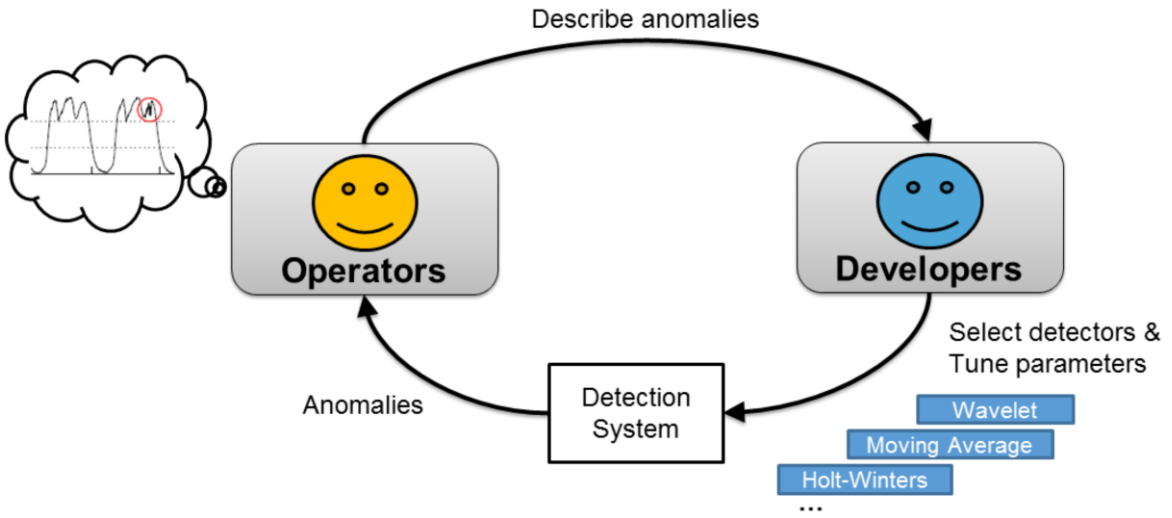


Then, to detect those anomalies, developers have to select suitable detectors and tune their internal parameters.

Sometimes, one detector is not sufficient enough, and they have to combine multiple detectors such as using majority-vote.

How to Build the Anomaly Detection System

In practice, it is more complex



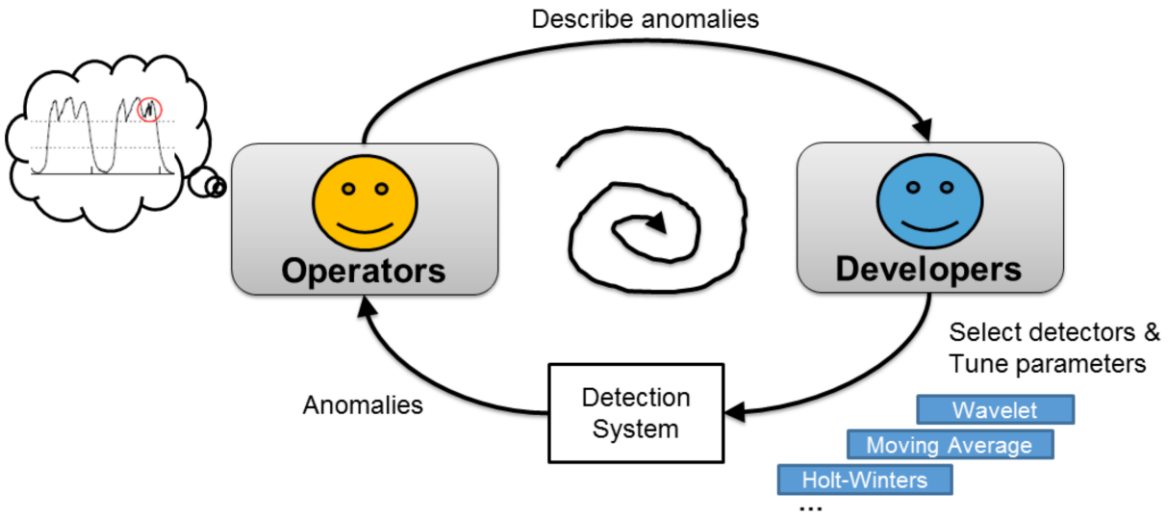
2015/12/15 Dapeng Liu (liudp10@mails.tsinghua.edu.cn) 7

The detection system starts to work and report anomalies to operators.

However, because operators may not describe the anomalies precisely, or developers cannot select or tune detectors well, the detection system is often inaccurate at the beginning

How to Build the Anomaly Detection System

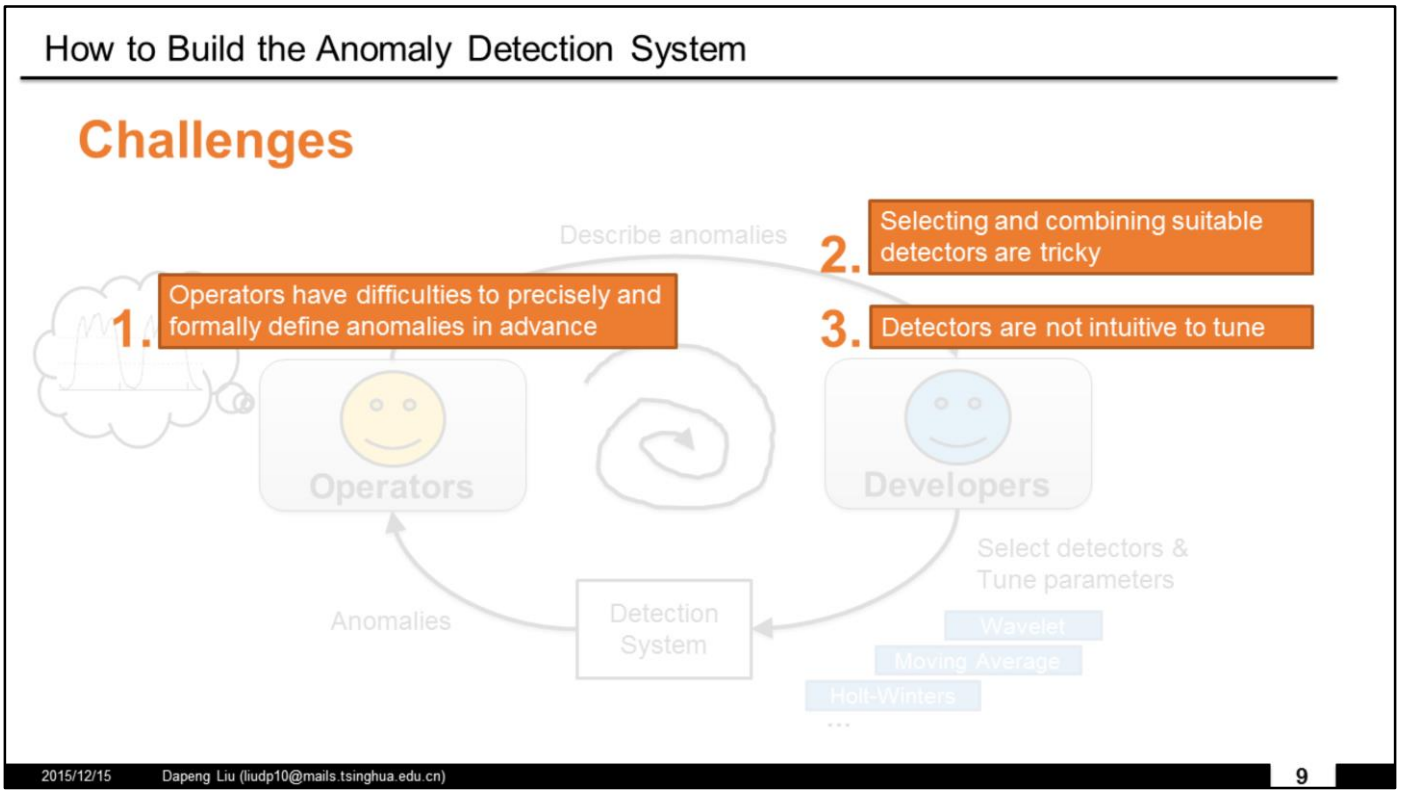
In practice, it is more complex



2015/12/15 Dapeng Liu (liudp10@mails.tsinghua.edu.cn) 8

So, it usually takes operators and developers a lot of efforts to iterate again and again, but at last the system may still not work, generating many false positives or false negatives, and operators are not willing to use it.

This is one important reason that many proposed detectors are not being used in practice.



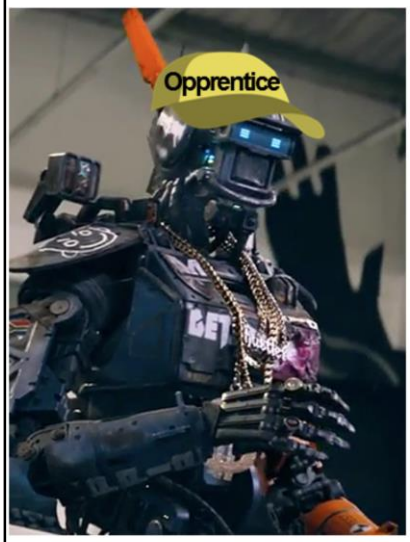
In summary, there are three main challenges in building an anomaly detection system.
First, …
Second…
Third…(Especially for those complex dectors)

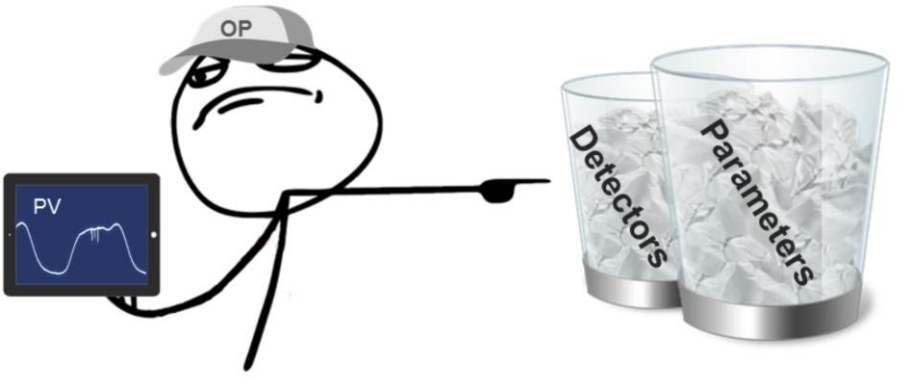
Because of these practical challenges, building an anomaly detection system requires a lot of manual efforts.

This is the problem we want to solve in this paper. We want to reduce the efforts of building an anomaly detection system. But how can we do that?

So we propose Opprentice, that is, operators' apprentice, a framework to learn anomaly detection from operators.

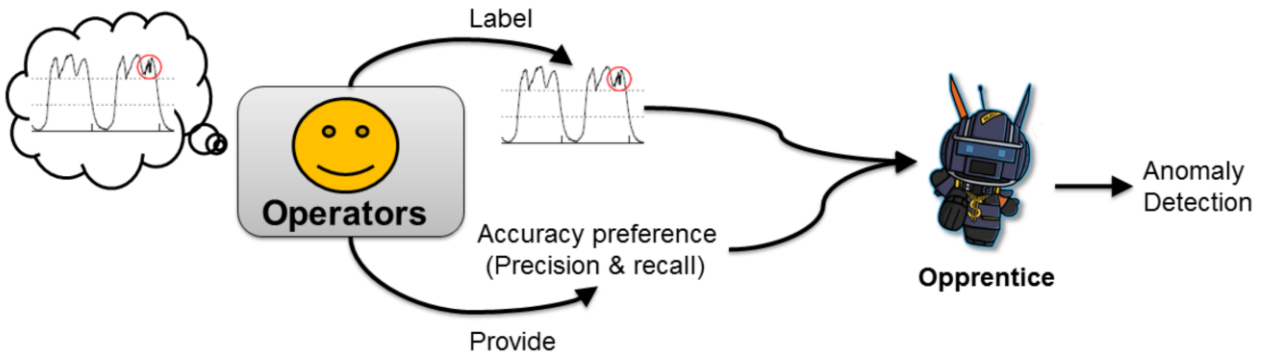
A More Natural Way





We know that given a KPI curve, it is more easy for operators to tell whether there exist anomalous behaviors on the curve. So, image this, operators only need to show Opprentice some anomaly cases and he will deal with the rest of things, such as selecting detectors and tuning parameters.

Design Goal

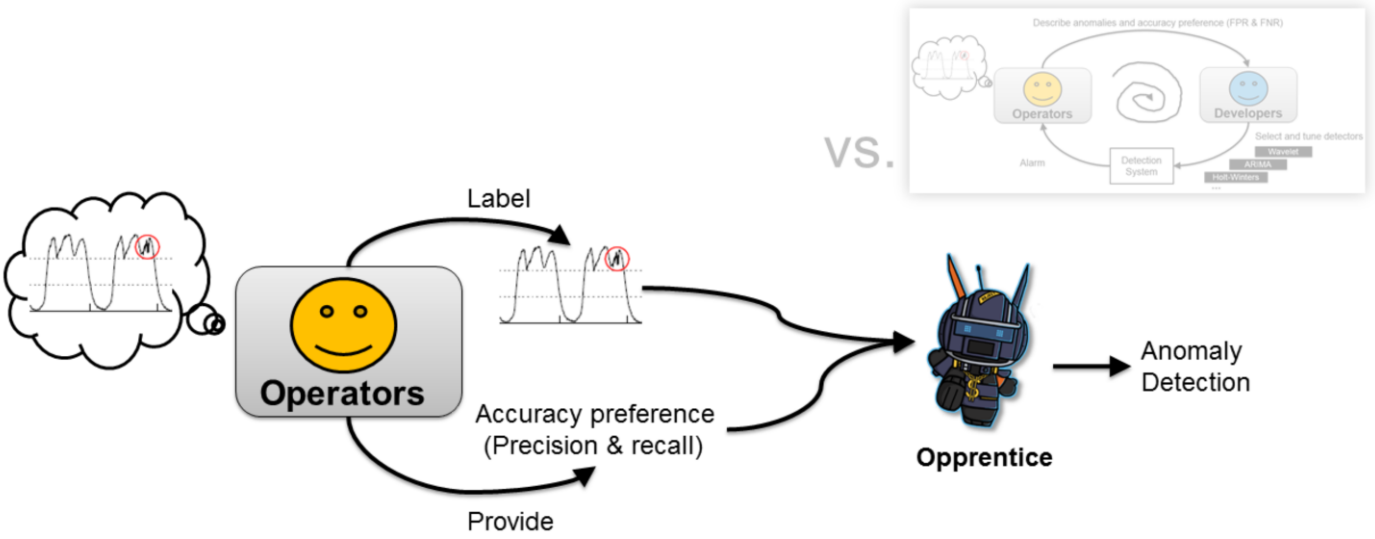


So we want to build a system like this.

Basically, we only require operators to do two things: label some anomaly cases in historical data, and provide their detection accuracy preference in the form of precision and recall.

The rest of things are all handled by Opprentice automatically.

Design Goal



We see that, when compared with the traditional way, operators do not need to formally define anomalies any more but only label some anomaly cases, which are much easier for them.

What's more, no developers are needed to select and tune many complex detectors.

Outline

- Background and Motivation
- **Key Ideas**
- Results
- Conclusion

Next, I will show you the key ideas of how we achieve this

Key Ideas

Detector model:

$$\text{data point} \xrightarrow{\text{a detector with parameters } \{p\}} \text{severity} \xrightarrow{\text{sThld}} \{1, 0\}$$

First, we use a uniform model to represent how different detectors work.

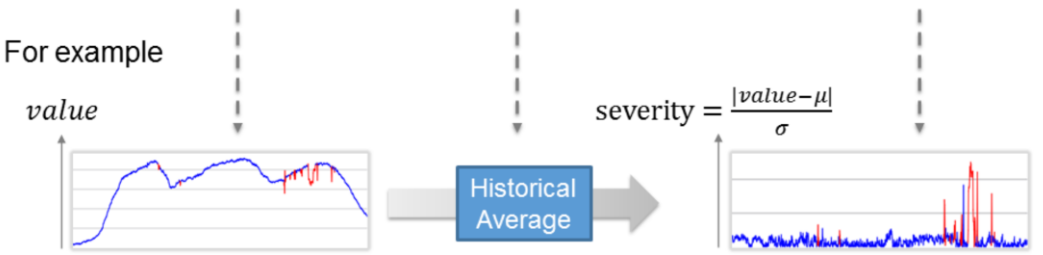
It is basically a two-step process.

Key Ideas

Detector model:

$$\text{data point} \xrightarrow{\text{a detector with parameters } \{p\}} \text{severity} \xrightarrow{\text{sThld}} \{1, 0\}$$

For example

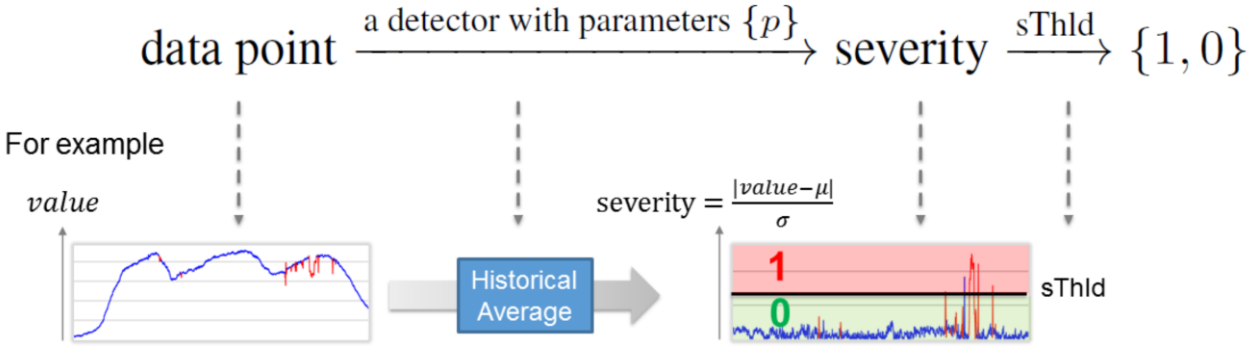


For example, this is our KPI data. The red parts are the anomalies labeled by the operators.

** Click
If we apply a detector, historical average for example, to this data, it will generate the anomaly severity for each data point. This detector measure the severity using how many times of the standard deviation each point is away from the average. The higher the severity is, the more anomalous the data point is.

Key Ideas

Detector model:



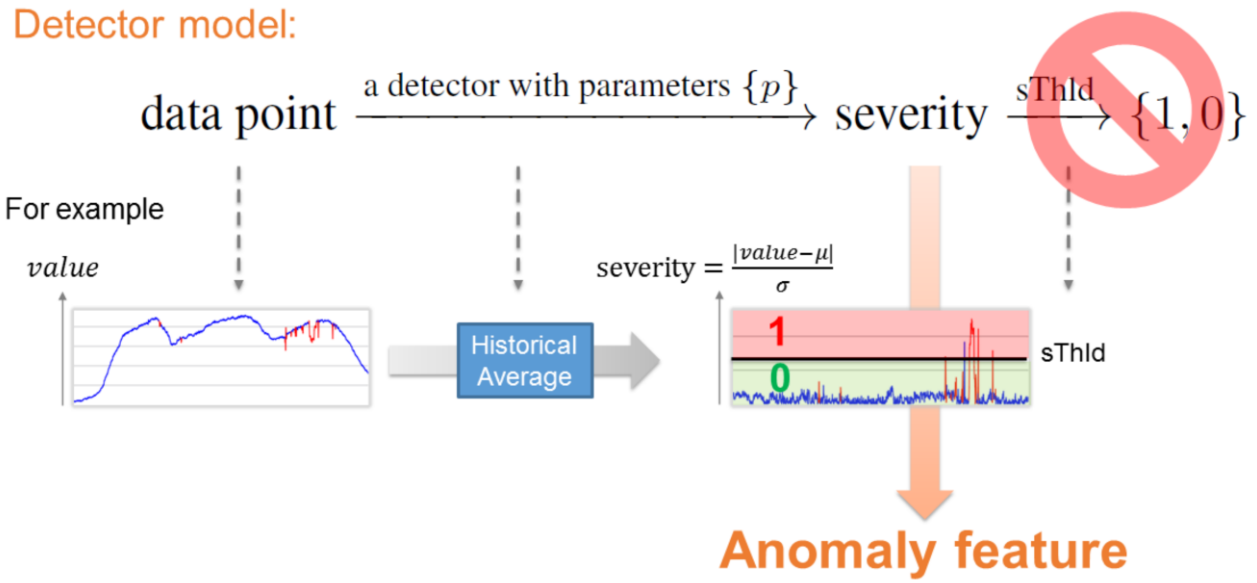
2015/12/15 Dapeng Liu (liudp10@mails.tsinghua.edu.cn) 17

Then, we need to set a severity threshold to decide anomalies.

So, this is a general detector model. Different detectors basically work in these two steps, except that they use different techniques or algorithms to measure the severities.

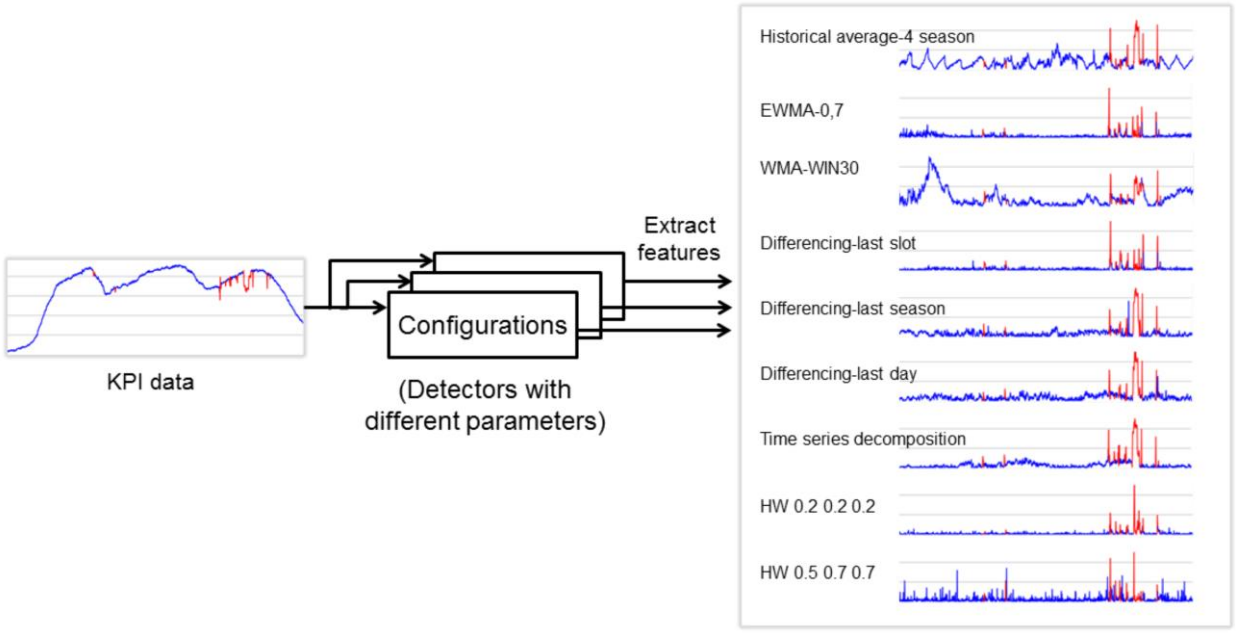
Key Ideas

Detector model:



In Opprentice, we do not let each detector determine the anomaly by itself. Instead, we only use the severity as the anomaly feature.

Key Ideas

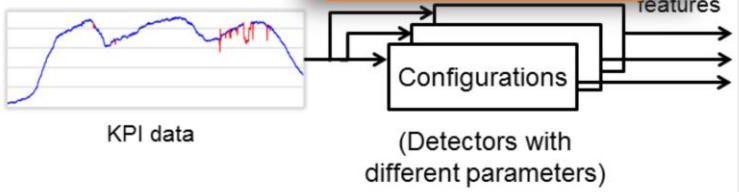


Multiple detectors with different parameters are used simultaneously to extract different anomaly features of the data.

We call a combination of a detector and its specific parameters a configuration.

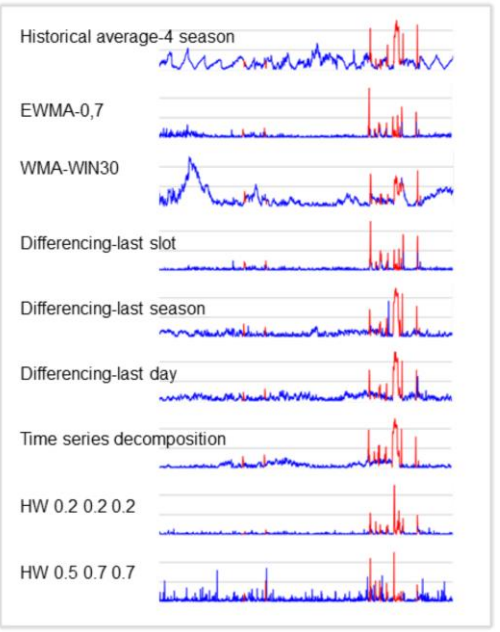
Key Ideas

Detector / # of configurations	Sampled parameters
Simple threshold [24] / 1	none
Diff / 3	last-slot, last-day, last-week
Simple MA [4] / 5	win = 10, 20, 30, 40, 50 points
Weighted MA [11] / 5	
MA of diff / 5	
EWMA [11] / 5	$\alpha = 0.1, 0.3, 0.5, 0.7, 0.9$
TSD [1] / 5	win = 1, 2, 3, 4, 5 week(s)
TSD MAD / 5	
Historical average [5] / 5	
Historical MAD / 5	
Holt-Winters [6] / $4^3 = 64$	$\alpha, \beta, \gamma = 0.2, 0.4, 0.6, 0.8$
SVD [7] / $5 \times 3 = 15$	row =10, 20, 30, 40, 50 points, column =3, 5, 7
Wavelet [12] / $3 \times 3 = 9$	win = 3, 5 ,7 days, freq = low, mid, high
ARIMA [10] / 1	Estimation from data
In total: 14 basic detectors / 133 configurations	



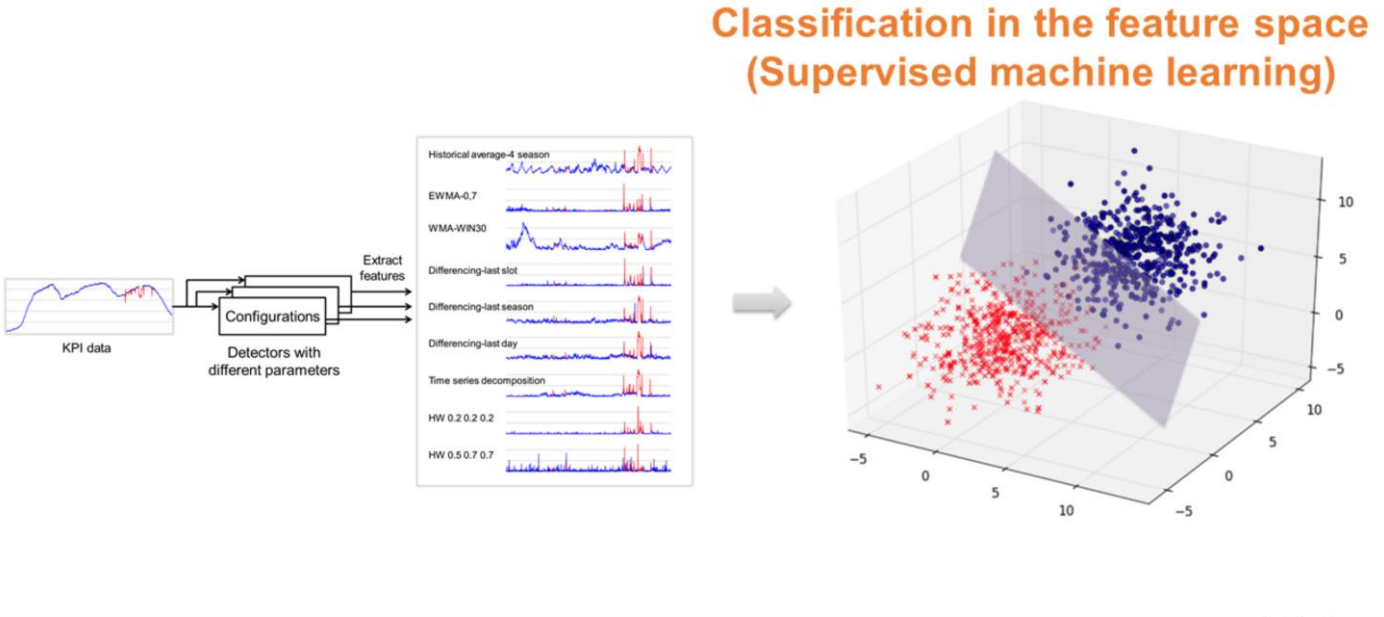
KPI data

(Detectors with different parameters)



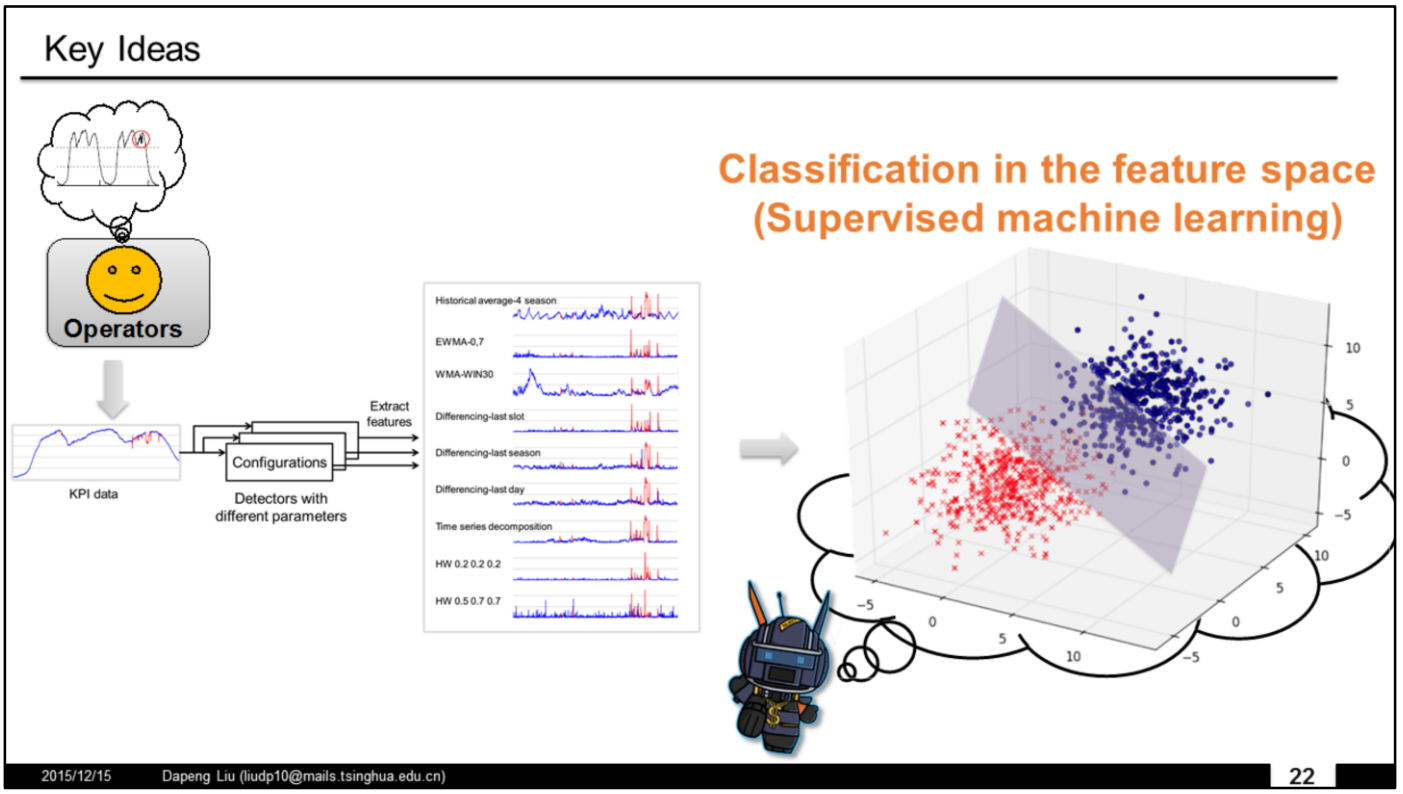
We broadly select 14 detectors and sample some parameters. Finally, we get 133 configurations. In other words, we have 133 anomaly features.

Key Ideas



If we consider each feature as one dimension, then each data point can be projected into a multi-dimensional space. Here is a 3D space example. The red points are corresponding to those anomalies labeled by operators.

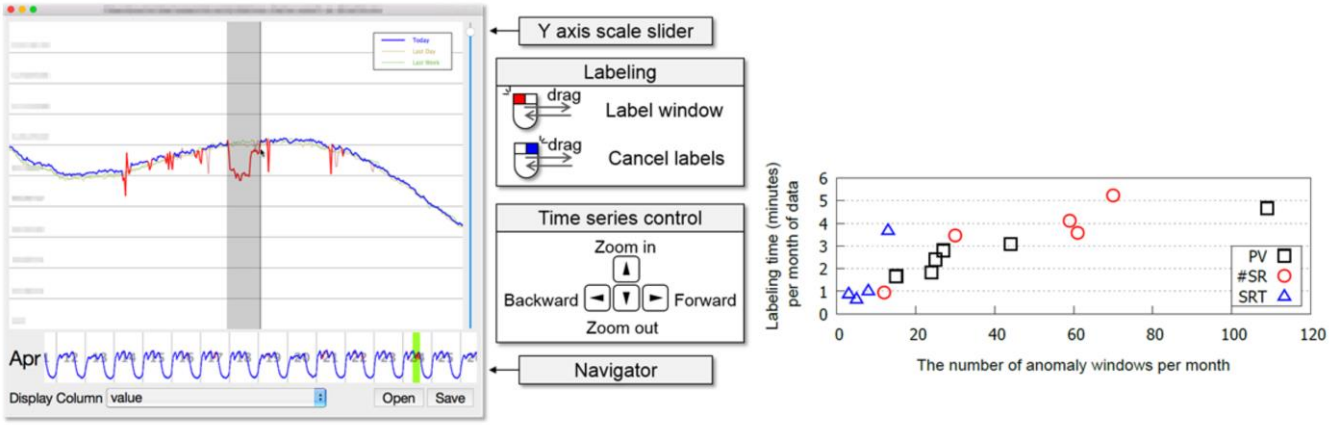
We can use supervised machine learning to do the classification in this multi-dimensional feature space.
And the classification model is used for future anomaly detection.



- So we see, the key idea behind this design is that, operators actually have the domain knowledge of anomalies in their mind, but it is difficult for them to formally define it. So, we use machine learning to model those anomaly concepts from historical cases.
- This space is just like the brain of Opprentice.
- In this process, machine learning will automatically find the classification boundaries in the feature space. In other words, it can select which detectors and parameters are suitable for detecting those anomalies cared by operators.

Address Challenges of Designing Opprentice

- Labeling overhead
  - Solution: an effective labeling tool



There are several challenges when designing Opprentice.

- The first one is how to solve the labeling overhead? Our solution is to develop an effective labeling tool. This is the user interface of our labeling tool.

- Labeling with this tool won't cost much time, actually less than 6 minutes for each month of data according to our use experience.

Address Challenges of Designing Opprentice

- Labeling overhead
  - Solution: an effective labeling tool
- Incomplete anomaly types in the historical data
  - Solution: incremental re-training with new data

- The second one is that the historical data may not contain all kinds of anomaly types. Our solution is to incrementally re-train Opprentice with the latest data

Address Challenges of Designing Opprentice

- Labeling overhead
  - Solution: an effective labeling tool
- Incomplete anomaly types in the historical data
  - Solution: incremental re-training with new data
- Class imbalance problem
  - Solution: adjusting classification threshold (cThld) based on the preference

- The third challenge is class imbalance problem. That is, anomalies are infrequent in the data, so they are often ignored by classification algorithms.
- To solve this challenge, we adjust the classification threshold based on the accuracy preference to give higher priority of the anomaly class

PC-Score: Adjust the Classification Threshold (cThld)

$$Precision = \frac{\#\ of\ true\ anomlies\ detected}{\#\ of\ anomlies\ detected}$$

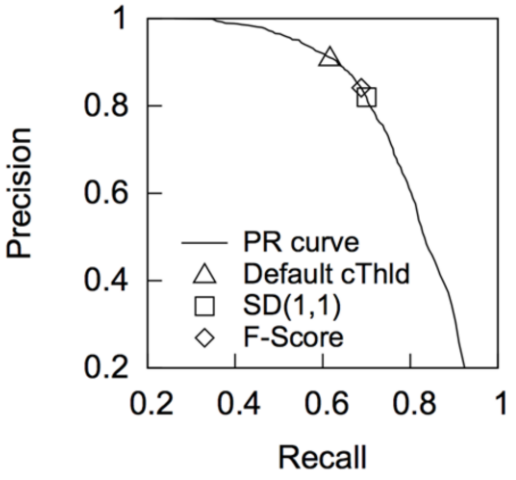
$$Recall = \frac{\#\ of\ true\ anomlies\ detected}{\#\ of\ true\ anomlies}$$

Precision and recall are two important metrics to evaluate the detection accuracy. Precision is how many anomalies detected are the true anomalies. Recall is how many true anomalies are detected. Empirically, precision and recall are often a trade-off.

PC-Score: Adjust the Classification Threshold (cThld)

$$Precision = \frac{\#\ of\ true\ anomlies\ detected}{\#\ of\ anomlies\ detected}$$

$$Recall = \frac{\#\ of\ true\ anomlies\ detected}{\#\ of\ true\ anomlies}$$



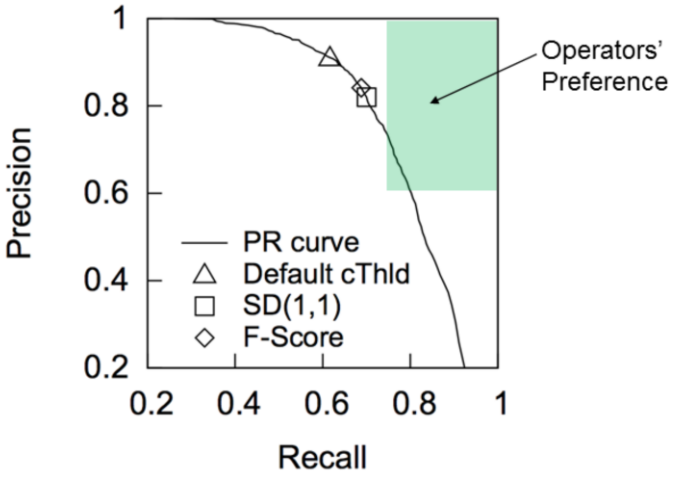
This curve is the precision-recall curve, or PR-curve of a random forest, each point on the curve is derived from a different classification threshold. To select a proper point on the curve, there are several methods, such as using the default cThld 0.5, selecting the point that has the shortest distance to the top-right corner, or the point that can maximize the F-Score.

However, these methods do not take operators' preference into considerations.

PC-Score: Adjust the Classification Threshold (cThld)

$$Precision = \frac{\#\ of\ true\ anomlies\ detected}{\#\ of\ anomlies\ detected}$$

$$Recall = \frac{\#\ of\ true\ anomlies\ detected}{\#\ of\ true\ anomlies}$$

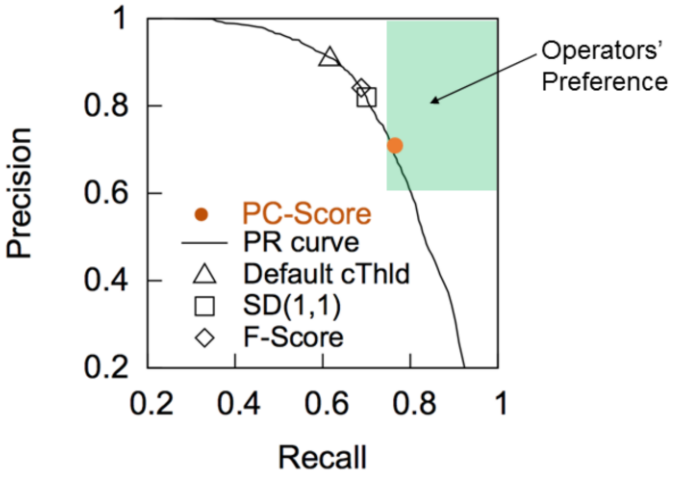


For example, suppose that this green region represents operators' accuracy preferences. We see the PR-curve in fact has points inside these regions, which means it can satisfy the preference. But, the traditional methods will ignore the preferences and won't select the correct points or the corresponding cThld.

PC-Score: Adjust the Classification Threshold (cThld)

$$Precision = \frac{\#\ of\ true\ anomlies\ detected}{\#\ of\ anomlies\ detected}$$

$$Recall = \frac{\#\ of\ true\ anomlies\ detected}{\#\ of\ true\ anomlies}$$



$$\boldsymbol{PC\text{-}Score} = \begin{cases} F\text{-}Score + 1 & ,if\ the\ point\ satisfies\ the\ preference \\ F\text{-}Score & ,otherwise \end{cases}$$

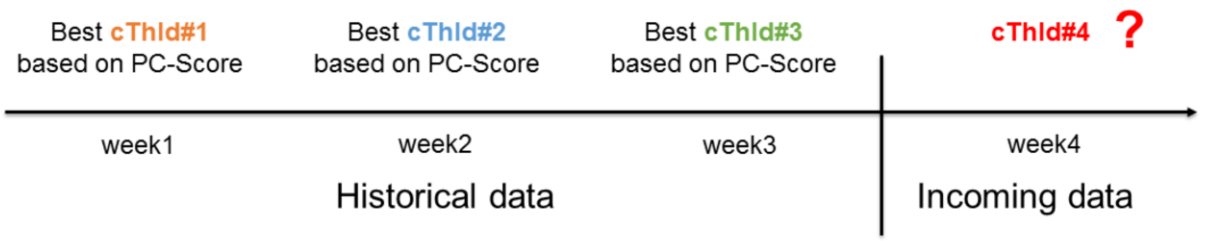
(Preference-centric Score)

2015/12/15 Dapeng Liu (liudp10@mails.tsinghua.edu.cn) 29

Therefore, we propose the Preference-centric score, or the PC-Score. PC-Score is based on F-Score, but it takes operators' preference into considerations. The main idea is intuitive. We give higher priority to those points that satisfy the preference by adding the F-Score by one.

So the cThld selected by the PC-Score can satisfy the preference if possible.

EWMA: Predict cThld based on PC-Score



Exponentially weighted moving average (EWMA)

$$\text{cThld}_i^p = \begin{cases} \alpha \cdot \text{cThld}_{i-1}^b + (1-\alpha) \cdot \text{cThld}_{i-1}^p & , i > 1 \\ \text{5-fold prediction} & , i = 1 \end{cases}$$

OK, now based on the PC-Score, we can determine the best cThld for each week if we have the data of that week.

**Click
Then, a question is that for online detection, how can we determine the best cThld for a future week, where we do not have the data yet.

**Click
To solve this problem, we here use **exponentially weighted MA** to predict the best cThld of next week based on those historical best cThld.

Address Challenges of Designing Opprentice

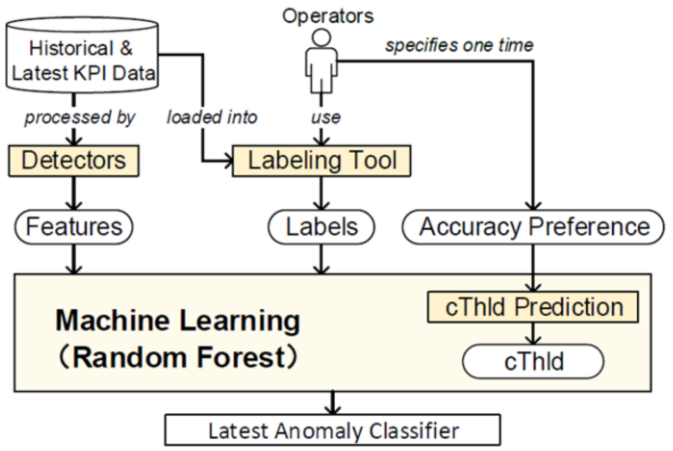
- Labeling overhead
  - Solution: an effective labeling tool
- Incomplete anomaly types in the historical data
  - Solution: incremental re-training with new data
- Class imbalance problem
  - Solution: adjusting classification threshold (cThld) based on the preference
- Irrelevant and redundant features
  - Solution: random forests

- And Last, because we want to save manual efforts, the detectors and their parameters are used without carefully evaluation, so many of the features provided by these detectors could be irrelevant or redundant.

- To solve this problem, we conducted pilot experiements on different machine learning algorithms, and find that the random forest, an ensemble based machine learning algorithm, turns out to be more accurate and robust to irrelevant and redundant features. The last speaker has also confirmed this in their paper.

Design Overview

Training a classifier



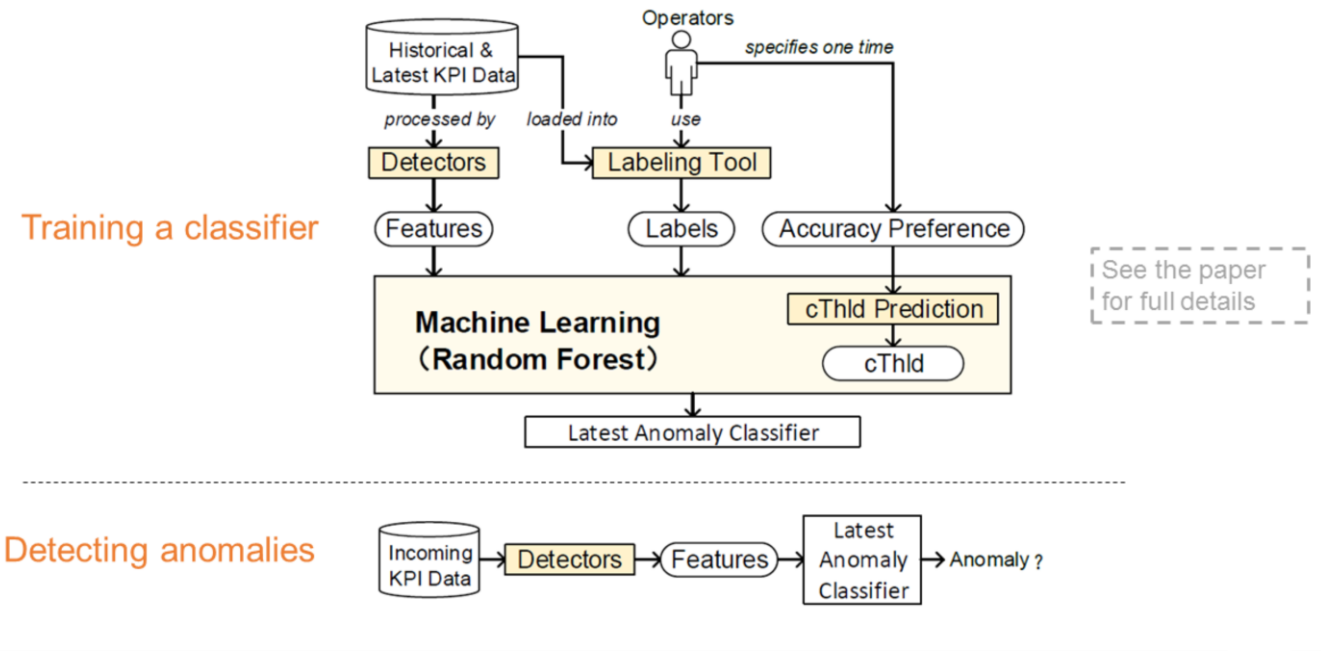
See the paper for full details

This is the design overview of Opprentice.

First we train the anomaly classifier. Give the KPI dataset, dozens of detectors extract the features, and operators provide the anomaly labels. The features and labels are used to train a random forest classifier. Besides, operators specify their accuracy preference in the form of precision and recall. It is used to adjust the classification threshold of the random forest.

The classifier is incrementally re-trained with the latest data. For example, we can do it once a week.

Design Overview



For the detecting process, the same detectors extract the features of the incoming data. Based on these features, the latest classifier will decide the class of each data point, anomaly or not.

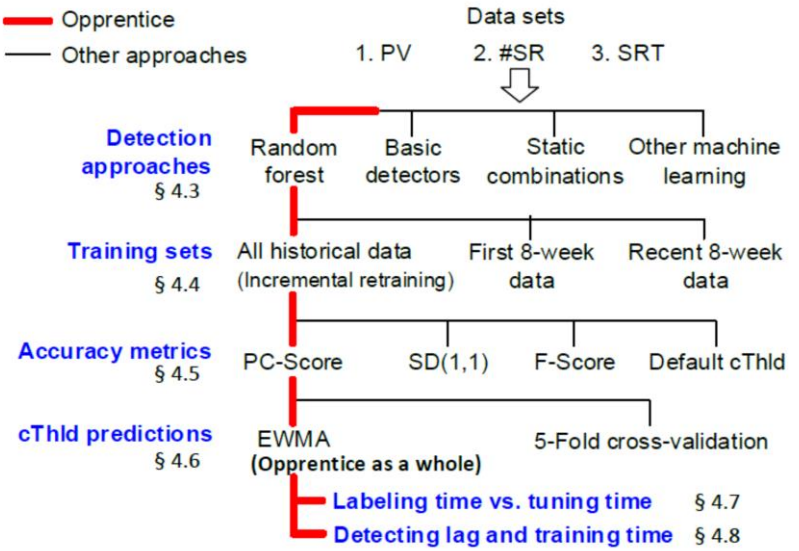
Please read our paper for the details of each component.

Outline

- Background and Motivation
- Key Ideas
- **Results**
- Conclusion

I will show you the evaluation results of Opprentice

Evaluation

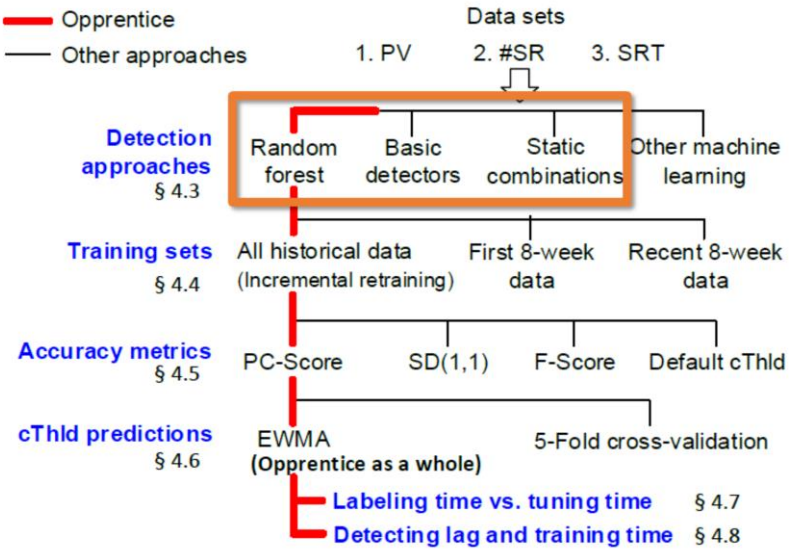


This is our evaluation flow. We use three representative KPIs from Baidu. They are labeled by the operators. We compare different components of Opprentice with other methods.

With all these four parts combined together, we get Opprentice as a whole here.

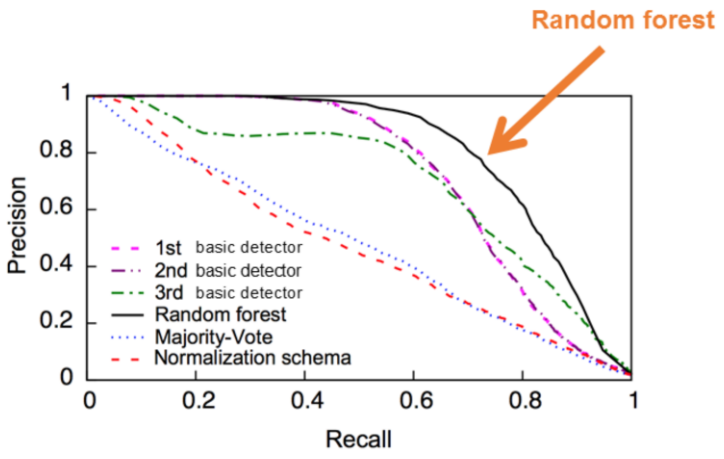
We also evaluate the labeling time, the online detecting lag and the offline training time of Opprentice.

Evaluation



Let's firs look at the comparison between the random forest and the basic detectors and two static methods for combining different basic detectors.

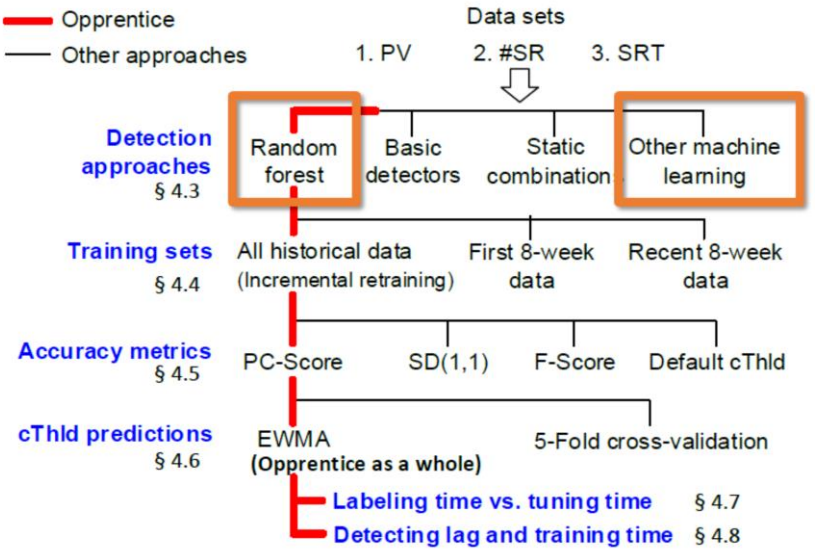
Random forests vs. Basic Detectors and Static Combinations



This figure shows the precision-recall curves, or PR curves on the PV dataset. The results of other two KPIs are similar.

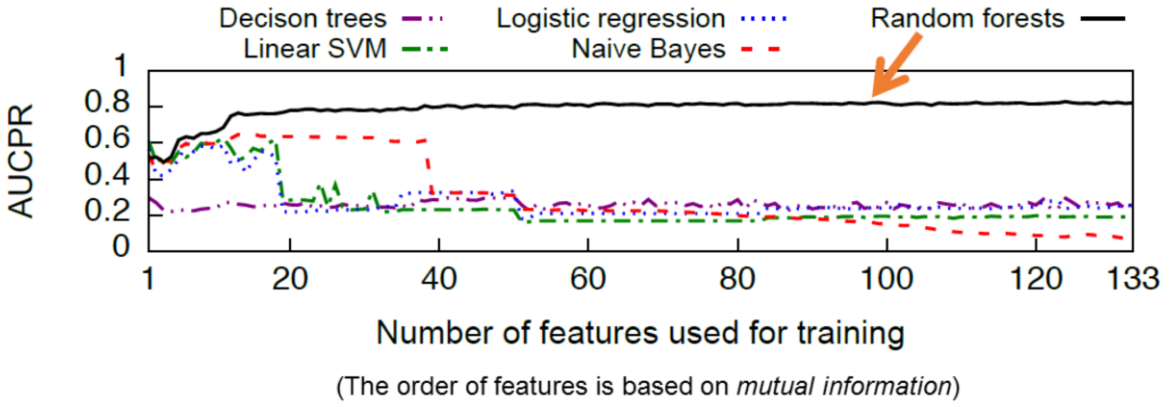
Empirically, the area under the PR curve is larger, the detection approach is better. We see that random forest outperforms other detection approaches, including the top-3 best basic detectors, and two static combination methods (majority-vote and normalization schema).

Evaluation



We also compare the random forest with other machine learning algorithms.

Random Forests vs. Other Learning Algorithms

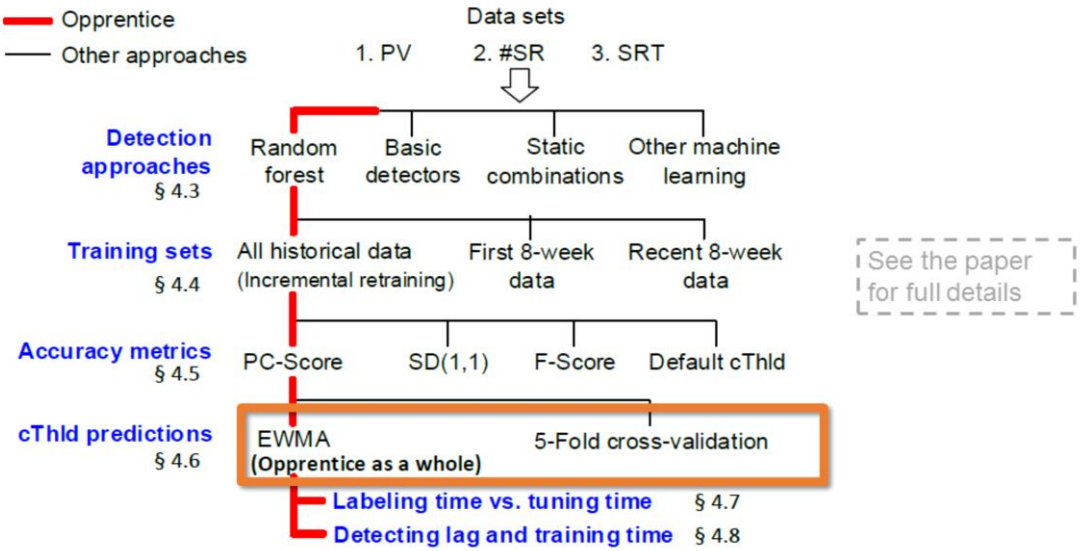


(The order of features is based on *mutual information*)

AUCPR means the area under the PR-curve, and it is the larger the better. In order to see how different machine learning algorithms perform when faced with irrelevant and redundant features, we rank the 133 features based on mutual information, a typical feature selection method, and add the features one by one.

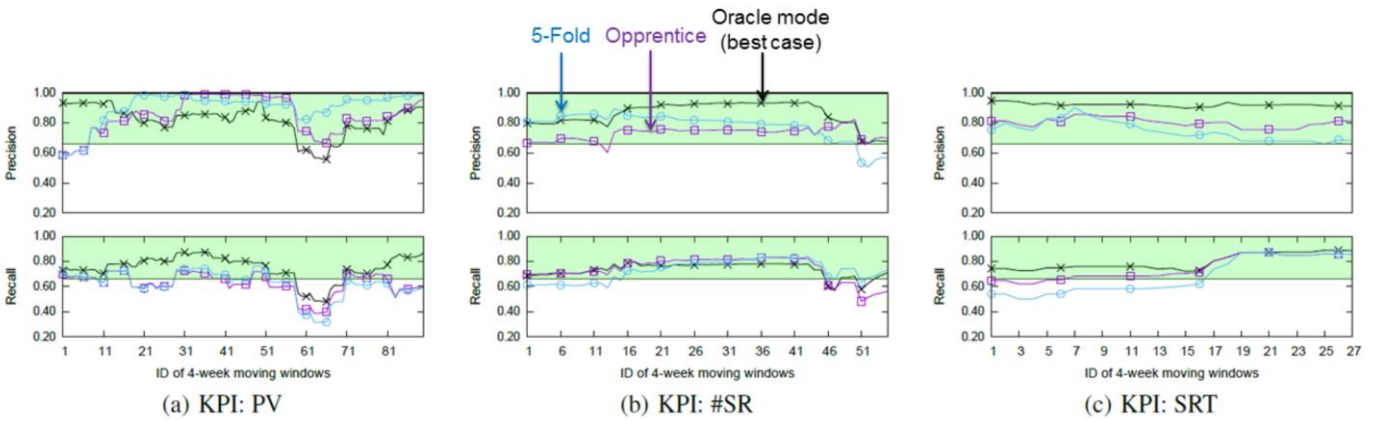
We see that the random forest is almost not affected when we use more features, on the other hand, the performance of other learning algorithms decreases a lot. The result shows that the random forest is more robust to irrelevant and redundant features

Evaluation



I will just skip other detailed results, and show the performance of Opprentice as a whole. In this step, we compare the EWMA used by Opprentice to predict the classification threshold with another method 5-fold cross-validation.

Opprentice as a whole



(a) KPI: PV (b) KPI: #SR (c) KPI: SRT

Opprentice achieves

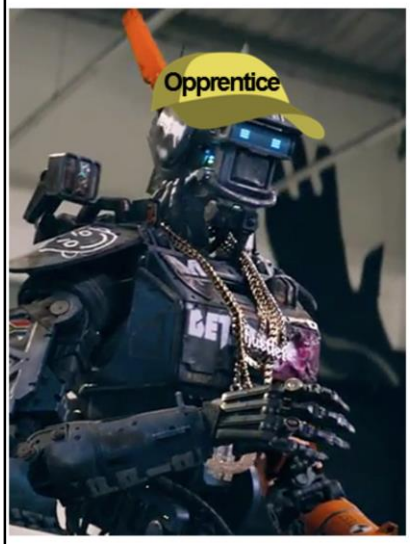
40% **23%** **110%**

more points inside the preference regions than 5-Fold cross-validation

The green regions represent the preference of precision and recall given by the operator. Overall, we see that for the three KPIs, Opprentice can satisfy or approximate the preference most of the time.

When compared with 5-fold cross-validation, Opprentice achieves more points in those preference regions.

Conclusion



- Opprentice is an **automatic** and **accurate** machine learning framework for KPI anomaly detection

~~Defining anomalies~~ ~~Selecting detectors~~ ~~Tuning detectors~~

- Opprentice **bridges the gap** in applying complex detectors in practice
- The idea of Opprentice
  i.e., **using machine learning to model the domain knowledge**

could be a very promising way to automate other service managements

In summary

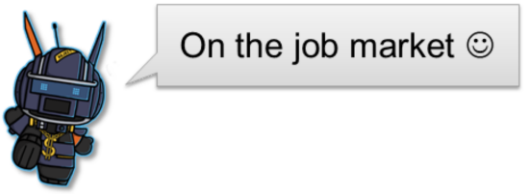
First, #Read#

Second, Opprentice bridges the gap in applying complex detectors from literatures in practice

Third, #Read# , Such as diagnosing root causes

Thank you

liudp10@mails.tsinghua.edu.cn



So, that's all for Opprentice, and I'm happy to take questions now

Thank you.

Backup

Questions

- What if anomalies are frequent?
  - First, based on our experience in Baidu, anomalies are not that many, or someone will be fired
  - If it is the case, we do not have to label all the anomalies, machine learning is good at dealing with noisy data
  - But, it is more promising to test the performance of Opprentice with different sizes of noisy data. We consider it as future work, after all, right now we do not find frequent anomalies as a common case
- What if there are many KPI curves?
  - Different KPIs are operated by different operators, and the KPIs per operators are not that many
  - The anomalous behaviors of different KPIs can be similar, such as the page views of different ISPs. So we can reuse the classifier for those similar KPIs (we need to do the normalization for feature extraction)
  - This could be a great extension for Opprentice. We can explore this direction in the future.

Questions

- If operators do not know anomalies? Or some anomalies may not be visually identified from the curve?
  - First, the anomaly we focus on in this work is the anomalous behavior on single KPI curve (Not troubleshooting)
  - Our goal is detect anomalies confirmed by operators. After all, operators are the users of the detection system. If the anomalies they do not agree with, they do not know how to start investigation and may just ignore it.
- Precision and Recall are not very high
  - Because in the evaluation, we use the performance of fixed window size, 4 weeks. But anomalies are rare (actually, less than 4% for some weeks). For example, missing just a few could decrease recall a lot
  - But, in this case, low recall does not mean we miss a lot of anomalies, the absolute number is small too, so the performance is not that bad

Questions

- Why random forests work better than others
  - There are a lot of irrelevant and redundant features
  - The two properties of random forests: ensemble and random (talk offline)
- What about network traffic data
  - Opprentice can deal with traffic data, it is similar with the PVs of Baidu (seasonality)
  - We apply Opprentice to a traffic data in our journal paper
- What do you mean by complex detectors
  - They take advantage of some more complicated techniques, such as wavelet analysis. It takes more time for operators to learn and understand such detectors
  - They have more parameters, or their parameters are not intuitive to tune
- Is Opprentice deployed?
  - We have build a prototype of Opprentice, and evaluate it with real data from Baidu
  - We are now working on deploying Opprentice in Baidu to detect PVs of different products