NTNU

Norwegian University of Science and Technology

Anomaly Detection Using Improved k-Means Clustering on Apache Flink

RECSI 2022

Slobodan Petrović and Aleksander Styrmoe

October 19, 2022

Contents

Outline

Introduction and background
- Introduction to cluster-based IDS
- The k-means algorithm
- Apache Flink, streaming and distributed computing
- Overall goal: Make a better IDS

Motivation and methodology
- Challenges and Possibilities
- How we will make a better IDS

Contributions, results and conclusions
- Theoretical contributions
- Experimental work
- Final words

Introduction to Intrusion Detection Systems (IDS)

- Detect and identify various attacks against hosts and networks in real-time.
- Misuse detection systems.
- Anomaly detection systems.

Machine learning and cluster analysis

- Supervised and unsupervised machine learning.
- Cluster-based anomaly detection.
- Labelling needs to be done.

k-means algorithm

- Originally proposed by Forgy (1965) and Lloyd (1982).
- Data points or vectors are grouped into k clusters.
- A centroid is the mean of a cluster.
- On-line k-means algorithms, like MacQueen (1967).
- Offline k-means in IDS: Batch k-means.

The algorithm

1. Initialize k centroids at random.
2. While the algorithm has not converged:
   2.1 Assign each vector to its currently closest centroid.
   2.2 Move each centroid to the mean of its currently-assigned vectors.

Improvements of k-means

Naïve k-means is sometimes too slow for application in IDS. Many unnecessary computations take time.

The triangle inequality

For any three points, a, b and c, where the distances between the points are $d(x, y), x, y \in \{a, b, c\}, x \neq y$, the following holds:
$d(a, b) \leq d(a, c) + d(b, c)$.

Some improvements

- Compare-means (Philips)
- Upper and lower bounds (Elkan)
- One lower bound (Hamerly)

Streaming platforms and distributed computing

- Streaming: A pipeline of multiple operators process events as they arrive.
- Distributed (cluster) computing: Split the work on multiple nodes.
- Parallelization like the MapReduce paradigm.
- Examples: Apache Hadoop and Apache Spark frameworks.

NTNU

Apache Flink and FlinkML

- A very new and active community.
- Supports stateful operations.
- Two modes of operation: Batch and streaming natively!

FlinkML

- Framework for ML.
- Stages and Models - fit and transform.
- Iteration paradigm.
- Supports some ML algorithms out of the box.
- Offline KMeans and Online KMeans are added.
- The Offline implementation converge after a fixed number of times.

Overall goal

Build a better k-means-clustering-based anomaly detection system that

1. makes use of proposed improvements of the k-means algorithm based on the triangle inequality and
2. is built in Apache Flink.

NTNU

Outline

Challenges and questions

1. Not much research on improvements of k-means put in IDS has been done.
   - What is the best improvement of k-means for application in IDS?
2. Apache Flink is a very young platform.
   - What needs to be done to make an k-means-cluster-based IDS in Apache Flink?
   - What needs to be done to make a functioning k-means implementation utilizing the triangle inequality in Apache Flink?

Possibilities and interesting potential

- Flink now supports new, very interesting features for application in k-means-based IDS.
- There are many different ways to operate k-means when it comes to online and offline k-means and variations/combinations.
- The fit and transform paradigm is an interesting approach to k-means. But, how accurate will it be? How can it be used in IDS?

How to make a better IDS

Anomaly detection is a trade-off between speed and accuracy.

- Ways to change the speed:
  Utilize the proposed k-means improvements.
- Ways to change the accuracy:
  Utilize different ways to operate the k-means algorithm.

In our article, we look at ways to increase the speed.

Outline

Introduction and background
- Introduction to cluster-based IDS
- The k-means algorithm
- Apache Flink, streaming and distributed computing
- Overall goal: Make a better IDS

Motivation and methodology
- Challenges and Possibilities
- How we will make a better IDS

Contributions, results and conclusions
- Theoretical contributions
- Experimental work
- Final words

NTNU

Introducing domains as a concept

- K-means cannot process non-numeric features.
- Z. Yu (2011) suggests splitting the dataset and performing k-means on each subset, which all share the same values on non-numeric attributes.
- Contribution: Introduce domains and a method to support domains in streaming platforms.

Domain
A domain is a group of points with the same value on all non-numeric attributes (i.e., attributes that cannot be included in the k-means algorithm). The k-means algorithm is performed on each domain separately.

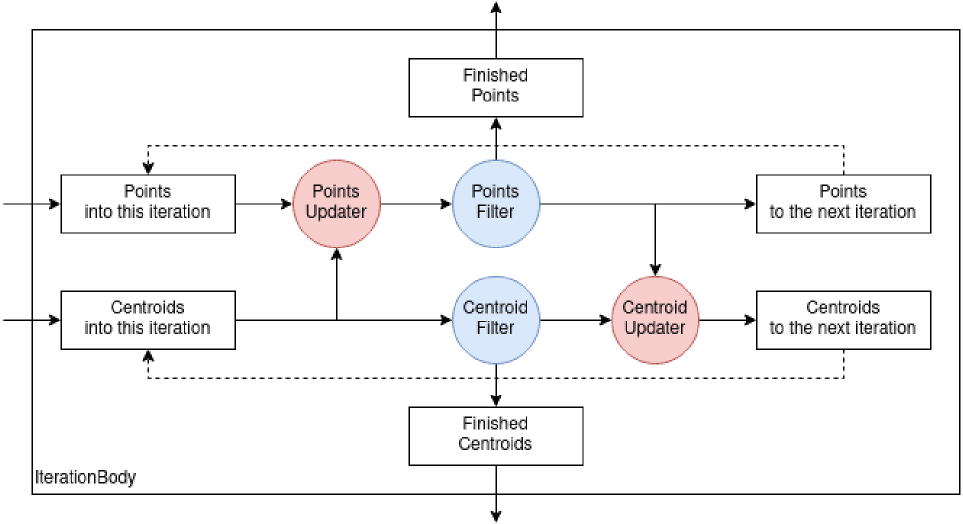
Consequences of introducing domains

- Domains may be just anomalies or nomalies. Therefore, we need support from signature-based IDS.
- Some domains may be so small k-means need to wait before running on that domain.
- In streaming, domains can converge at different times!

Contributions to Apache Flink

- Adjust KMeans in Apache Flink to support domains by having one stream for *Point* objects and one for arrays of *Centroid* objects.
- Adjust KMeans in Apache Flink to converge by storing movement of centroids in *Centroid* objects and setting *finished* flags in *Point* objects.
- Adjust KMeans in Apache Flink to support multiple improvements of the k-means algorithm by introducing *PointUpdater* and *CentroidUpdater* operators that call *update* functions in *Point* and *Centroid* objects respectively.
- Adjust the improvements to fit into Apache Flink.

Data flow of k-means inside an iteration in Apache Flink



Experimental work

- Implemented improvements have the same accuracy, measured in Positive Predictive Value (PPV).
  $PPV = \frac{BR*TPR}{BR*TPR+(1-BR)*FPR} \approx 0,7$
- Reduction of number of distance calculations compared to naïve k-means is used as a measure of efficiency.
- All tests was done using offline mode of operation and the NSL-KDD dataset.

Experimental work - results

- With no parallelization only Compare-Means gives a small gain in speed.
- Parallelization is important to gain a smaller execution time.

Version	% of dist.calc. skipped
Naïve k-means	$0,0\%$
Compare-Means	$39,0\%$
Elkan's improvements	$81,1\%$
Hamerly's improvement	$59,8\%$

NTNU

Some final words

- With parallelization and the proposed improvements of k-means, execution times can be reduced drastically.
- Elkan's improvement reduced the number of the costly distance calculations the most with $81,1\%$
- Multiple theoretical contributions have been made by introducing domains.
- Also, pseudo-code adjusting improvements of k-means to Flink has been made.