Entity Resolution as an Application- Team B

Data Science Institute
COLUMBIA UNIVERSITY

Names of Authors: Haikun Du, Xiaoxiao Yan, Yuanhao Lu, Yixiang Cai
Industry Mentors: André Carvalho Bittencourt, Michel Avila
Faculty Mentors: Andreas Mueller

Data Science Capstone Project with Neoway

Abstract

A great model alone does not guarantee a great product, we also need a robust infrastructure to well support the data flow under any circumstances. Therefore, as team B, we focus on the data platform side. We built up a scalable data pipeline to set up a system from data storage to data transfer and finally to the distributed system where the model can run on with both efficiency and accuracy.

Motivation

In data architecture aspect, we mainly faced following problems:

- **Too many data providers and consumers might have too many pipelines to maintain;**
- **The Shema of the data set might be vulnerable to some changes in the future;**
- **It can be very computational costly to compare each reference record with the input. Reduction of computation is needed.**

General Pipeline

According to problems stated above, we designed the pipeline as shown in Figure 1. It solves the problems in data transportation, data storage and computational cost.

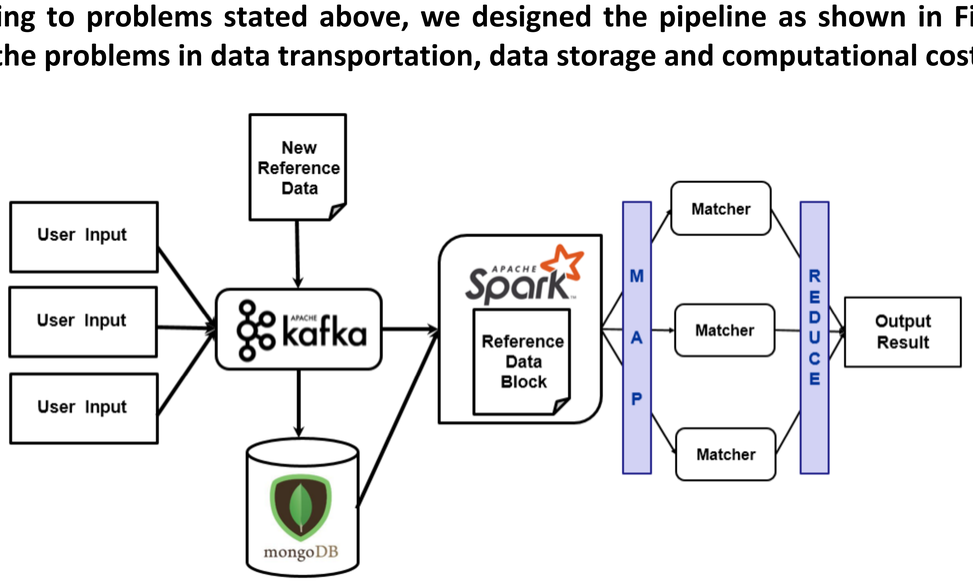

Figure 1. Architecture overview for the pipeline.

- **Kafka, which serves as a medium for data transformation, can 1) decouple the interaction between source and destination, which provides a uniform connector; 2) play a role as buffer serving data only when requested; 3)partition the data in order to be fault-tolerant.**
- **Due to the large-scale and schema-less nature of the real-world entity dataset, NoSQL databases will be more suitable than relational database as a main data storage in this scenario. With rich secondary indexes and great compatibility with Spark, MongoDB would be a good candidate among all the NoSQL databases. MongoDB store the reference data along with the blocks they belong to, and it is then connected to Spark to prepare the reference data for parallelized operations.**
- **By filtering with blocking keys and matching records in parallel, we can 1) greatly reduce the total computation, constrain the comparison only on the reference that are likely to match; 2) achieve a higher general computation speed in map-reduce styled computation.**

Blocking Strategy

As a preprocessing of reference data, blocking plays an important role in reducing computation and controlling general resolution. We dived into different strategies on different dataset. We also implement a general (domain-independent) blocking strategy with flexibility using supervised learning. (result shown in Fig. 2)

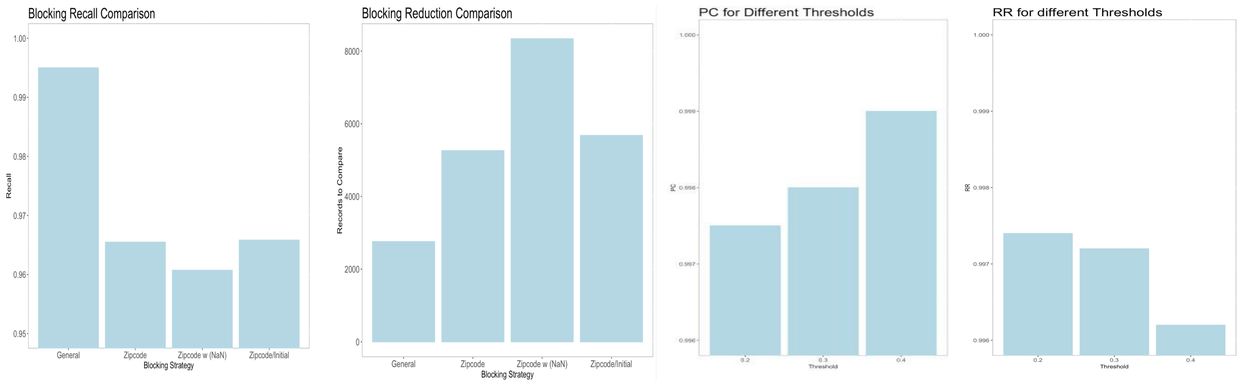

Figure 2. Blocking Result Comparison

- **Comparing intuitive blocking (according to domain) with general blocking on the same dataset (Figure 2), we can see higher recall and a much fewer pairs to compute in general blocking strategy, which indicates the big advantage of general blocking method;**
- **Efficiency and effectiveness are both important to the ER system. Set different thresholds to adjust the blocking keys for different situations. Fig.2 shows the pair completeness(PC) and reduction ratio(RR) for different thresholds on a subset of neoway company data.**

Conclusion

We constructed an application pipeline with robustness and consistency: Kafka supports data transportation with scalability and fault tolerance; MongoDB provides data storage with Schema flexibility; Blocking reduces total computation; Spark parallelizes the model application to fasten the process. This gives us a prototype of infrastructure for big data. an

Acknowledgments

We would like to thank Dr. Andreas Mueller for his support and feedback to the work of our team. Also thanks for the guide and help from the data engineers in Neoway, who give us strategy guidance and opportunity to dive into meaningful real-world topics.

References

[1] Matthew Michelson and Craig A Knoblock. Learning blocking schemes for record linkage. In AAAI, pages 440–445, 2006.