FIREFLY SPARK CLASSIFICATION OPTIMIZATION

Jacob Olinger
Simone Ludwig
North Dakota State University

Abstract: Classification is a problem at the forefront of computer science. However, models for classification remain extremely computationally expensive. Therefore, stochastic algorithms provide a more efficient manner of model training. Furthermore, the Apache Spark context provides parallel processing capabilities in order to further improve classifier efficiency
through increasing concurrency.

01. INTRODUCTION

In line with the old adage "more is better" computer scientists have been trying to develop parallel processes to speed up large-scale data processing for decades. In 2006 Apache released its first parallelization framework Hadoop, which enabled batch concurrent processing of large-scale datasets. The Hadoop package utilized the Map-Reduce framework of data processing from Google to effectively split and evaluate data. Hadoop also incorporated the Hadoop Distributed File System (HDFS), which is a fault-safe large-scale file storage system for concurrent cluster utilization based on splitting the data into 5 components designed for a master-worker cluster architecture. This architecture served as the basis for Apache's 2014 design of Spark which further improved HDFS with Resilient Distributed Data sets (RDD), increased computational efficiency, real-time data processing, and implemented in-memory file management, which resulted in an easier-to-use, low latency, general framework for at-scale computation. Classification is a classic machine learning problem grouping specific individuals into certain groups. Algorithms for classification tasks in machine learning have existed since the 1940s, however, with the increasing computational capability of the 21st-century classification has become an increasingly important part of modern machine learning. Since the classification of NP-hard problems, scientists have tried to deduce efficient deterministic algorithms. However, as these algorithms are currently mathematically improbable scientists instead turn towards less accurate, but much faster stochastic algorithms. These processes test only a small proportion of the sample space instead of searching its entirety leading to their non-deterministic nature. One popular area of exploration is bio-inspired algorithms that are based on the observation of wild animals. These algorithms provide a more extendable solution compared to classic gradient descent functions.

02 PROBLEM DESCIRPTION

Swarm Intelligence (SI) has become increasingly important in stochastic search algorithms. These algorithms are inspired by wild species' natural migration and movement patterns. The Firefly algorithm was authored by Xin-She Yang in his 2008 publication while working at Cambridge. The Firefly algorithm is a subset of bio-inspired algorithms based on the movement of fireflies. The Firefly algorithm randomly moves particles in the swarm toward the brightest or fittest direction for a defined number of iterations. Apache Spark is a relatively new parallel computational network based on the idea of a master-worker architecture of nodes executing similar algorithmic steps on different segments of the data. This structure benefits nature-inspired algorithms as each individual particle in the algorithm can be processed independently by a worker node. Spark will then enable us to concurrently process every particle in each iteration of the Firefly algorithm to determine the optimum centroids for data evaluation. As the Firefly algorithm is stochastic, it requires fewer evaluations to find global maxima. Several modern stochastic algorithms exist for classification, however, applying Wolpert's no-free-lunch theorem to classification averaged across all datasets all stochastic classification algorithms retain the same accuracy. Thereby, the implementation of any one algorithm will be equally applicable to any other. The use of a simplistic centroid-based vector Euclidean algorithm is thereby justified to predict target parameters and demonstrate the power of bio-inspired Spark-integrated algorithms.

03 RELATED RESEARCH

There is significant promise in using the Spark framework to parallelize many computationally intensive computer science problems. In fact, it has been proven to limit the curse of dimensionality problem that plagues popular swarm optimization algorithms. Many Swarm intelligence algorithms have been used in classification, the simplest of these is the Particle Swarm (PSO) algorithm. This algorithm is a more direct iteration of the Firefly algorithm that works on birds' migration and hunting patterns. PSO is the most commonly used stochastic algorithm for classification, however, PSO suffers from local convergence where the particles converge too quickly to local maxima instead of finding the global extreme. The Firefly algorithm converges based on the total fitness of surrounding individuals and a complex movement equation, which creates a less direct convergence pattern. The Firefly algorithm mitigates PSO's convergence problem through path and velocity vector optimizations that take into account the total fitness of the space related to distance instead of just the current global best. One popular optimization of the PSO algorithm is the use of more points in movement computation called local and global clustering to create a less deterministic movement of each particle. In fact, the higher topology or interaction between particles in computing the stochastic natural walk integral to nature algorithms the higher the probability of deducing a true maxima. The Firefly algorithm takes this principle to the extreme by exponentially increasing the interactions between particles and can be optimized through the parallel processing ability of Apache Spark. The low-latency capabilities of Spark have been well-documented in various NP-Hard scenarios promising speed-up observations for PSO and other machine learning models.

04 METHEDOLOGY

The implementation of the Firefly algorithm shall rely on three central parts fitness, brightness, and movement equations. These equations will serve as the mathematical core of the algorithm while Spark will enable these equations to process asynchronously in Java. There are alternative implementations to each of these equations that exist as optimizations to the Firefly algorithm, however, these are the most common iterations referenced in the original 2008 paper. The fitness equation for the function will be based on the Euclidean Distance function. the fitness of a particle shall be the sum of the distances between it and every other point p in the dataset with the same classification targetThe brightness at particle x of particle y based on the number of data records (n) is derived from the inverse square law. The movement between particles x and y given a random constant 0<r<1. While these equations form the mathematical foundation for the algorithm its scalability is found in loop concurrency when evaluating particle fitness and movement that enable Spark to scale at execution.

4.2 Psuedocode

```
Load dataset into spark RDD
initialize particle swarm
while i< Max Iterations do
    for x in dataset do
        for y in swarm do
            if x.target = y.target then
                                y.fitness+ = d(x, y)
            end if
        end for
    end for
    for j in swarm do
        for l in swarm do
            if i.fitness/n < b(j, l) then
                                j+ = m(j, l)
            end if
        end for
    end for
    i++
end while
```

06 CONCLUSIONS AND ACKNOWLEDGMENTS

The experimental implementation of a parallel Spark-enabled firefly Euclidean classification algorithm revealed that linear speed-up values more closely match increasing node numbers with larger datasets. Therefore, demonstrates the efficiency optimizations of Spark for live big data and the high concurrent exploitation available in nature-inspired algorithms like the Firefly algorithm. Future work would pursue further concurrent optimizations of the algorithm, improvements in the fitness function to promote more accurate clustering, and wider testing of the algorithm.

The authors acknowledge the support of the NDSU Department of Computer
Science sponsored by NSF Directorate for Engineering and EPSCoR Co-Funding through NSF grant EEC-2050175. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

05 RESULTS

The implemented algorithm was run in Java 8 on the North Dakota State University Spark Cluster. The Spark Cluster has 6 worker nodes, each with 1024MB of memory and 8 cores allowing it to physically run 6 executors and virtualize many more. The data was run on the publicly available binary EEG-Eye state data records, measuring the eyelid's response to different brain waves, with 15 thousand records, and a poker dataset, measuring the outcome of poker hands, with 1.2 million records to demonstrate the difference in speed-up based on the number of records and differences between binary and multi-classification targets. Each dataset was run with 1, 2, 6, and then 64 nodes in order to capture Spark's virtualization tendency. Each classification target had 100 particles initialized for it and the algorithm ran for 150 iterations. The EEG dataset has less proportionate speed up as the number of nodes increased while maintaining higher accuracy. The shifts in accuracy are caused by the random nature of the swarm initialization and as the Firefly algorithm is stochastic; deterministic results are not guaranteed resulting in changing accuracies. The speed-up values increased more proportionately with the number of nodes in the poker dataset demonstrating the difference between the binary targets of the EEG dataset and the ten multi-targets of the poker dataset as well as the eighty-factor difference in their sizes. In addition, the drop-off in speed-up after 6 nodes is due to the physical limitations of the cluster itself, which only has 6 worker nodes. Past this point, Spark is virtualizing nodes meaning that instead of adding physical computation Spark is job scheduling and switching on the same executor to simulate extra executors. Therefore, the speed-up values do not continue to linearly increase. In addition, the generally low accuracy of the poker dataset is from a lack of particles as both trials had only 100 particles per classification target, and as the dataset multiplied the algorithm failed to exploit the larger dataset for appropriate cluster groupings.