Big Data Analytics And Neural Networks In Industrial Manufacturing

Pleshan, Olena
sbs24043@student.cct.ie

***Abstract* – This paper looks at the role of Big Data technologies and neural networks in Industry 4.0 manufacturing systems, focusing on tools like Hadoop, Apache Spark, and cloud computing. It shows how these technologies enable real time data processing, predictive maintenance, and operational efficiency. Also, neural networks play a crucial role in tasks such as defect detection, process optimization and even manufacturing design. Despite the advantages, challenges around data integration, system interoperability, and ethical concerns, still remain. The paper concludes by discussing the emergence of Industry 5.0, which emphasizes human-centric collaboration with automation to create more resilient manufacturing systems.**

***Keywords* – Apache Hadoop, Apache Spark, neural networks, smart manufacturing, Industry 4.0 and 5.0**

I. INTRODUCTION

Manufacturing is one of the main areas of Big Data applications and it can be described as a 5M system consisting of materials, machines, methods, measurements and modeling (Lee et al., 2013). A lot of work is conducted in this field concerning, e.g., sensor applications in tool condition monitoring in machining, predictive maintenance of industrial robots and assessing the health of sensors using data historians (Chang, 2019). Industry 4.0 manufacturing systems refer to the integration of advanced digital technologies, such as the Internet of Things (IoT), artificial intelligence (AI), big data analytics, and automation into manufacturing processes. Multiple authors emphasized that the vision for Industry 4.0 manufacturing systems requires big data analytics and IoT to drive value creation. Big data analysis is conducted through sensors, processing, communication, and storage. The capabilities of big data analytics (BDA) in industrial manufacturing address challenges such as quality or process control, energy efficiency, diagnostics and maintenance, and risk analysis, with BDA techniques generating valuable outcomes (Rahul, 2023). Key technologies such as Apache Spark and Hadoop facilitate big data processing and analytics, while cloud computing provides scalable cloud infrastructure for storage and computing (Karau, 2015). Additionally, neural networks play a crucial role in advanced machine learning applications, enabling tasks like image and speech recognition. These tools are the key areas of focus in digitization of the modern manufacturing systems.

This paper aims to explore the use of Big Data technologies and Neural networks in advanced manufacturing and industry 4.0 contexts. The focus is on manufacturing, including applications of Spark, Hadoop, AWS, and neural networks.

II. RESEARCH OBJECTIVES

This paper aims to evaluate and analyze state-of-the-art technologies, such as Hadoop, Apache Spark, cloud computing, and deep learning algorithms, in the context of their applications in manufacturing. This includes exploring the role of various data sources in the application of these technologies, and in going through various case studies and prototype work to evaluate the applications of the manufacturing data and data analytics.

The text also aims to identify key challenges and opportunities presented by the integration of Big Data Analytics, Industrial IoT, and neural networks in the manufacturing industry, with a focus on how these technologies can transform the sector while addressing emerging concerns such as automation, ethics, and human involvement.

We also aim to describe an experimental setup that demonstrates a potential practical application of the use of big data in an industrial setting.

Finally, the paper aims to provide an assessment of opportunities and areas that need further research.

III. STATE OF THE ART TECHNOLOGIES OVERVIEW

A. Hadoop

Hadoop is an open-source framework that allows for the distributed processing of large data sets across clusters of computers using simple programming models. In manufacturing, it is used to store and analyze vast amounts of data generated from various sources, such as sensors and machines, enabling real-time insights, predictive maintenance, and improved decision-making to enhance operational efficiency and reduce costs (Lin, 2016).

B. Apache Spark

Multiple studies have been performed and proof-of-concept systems have been developed, tested and deployed in different manufacturing industries using Apache Stark in combination with downstream and upstream data processing / visualization technologies (Gupta, 2017).

Apache Spark is a high-speed cluster computing engine designed for handling large volumes of streaming data (Holden Karau, 2015). A study conducted by Su et al, 2024, demonstrated the use of Spark for effectively predicting machine faults and malfunctions in real time. The study demonstrated a successful combination of the main three pillars: data warehousing, big data processing and deep learning on Spark, showing minimal latency in data processing and highly accurate fault detection (Su, 2024).

Another study (Uğuroğlu, 2021) focused on the transformation of the automotive industry through Industry 4.0, emphasizing the digitization of the manufacturing value chain. It explores the challenges of collecting and analyzing real-time data from production processes, specifically examining the relationship between injection machine parameters and product quality at Farplas Automotive Company. The study aimed to develop machine learning models to identify defective products and implemented a streamed data pipeline using Kafka and Apache Spark to facilitate real-time analysis. This approach shows the potential of enhancing automation in production by reducing reliance on human intervention for defect detection, moving toward a more automated "dark factory" environment.

C. Cloud computing for Big Data Analytics

Cloud computing is described as a utility model that provides on-demand access to a shared pool of computing resources, facilitating the integration of manufacturing processes and enhancing inter-organizational relationships. It enables manufacturers to adopt innovative practices through cloud-based services such as Infrastructure-as-a-Service (IaaS), Platform-as-a-Service (PaaS), and Software-as-a-Service (SaaS), which streamline operations and reduce costs (Ooi, 2018).

This evolution, according to Li 2017, has led to the emergence of cloud manufacturing, which implies a smart, networked model that combines cloud computing with manufacturing to support product customization, global collaboration, and agile market responses.

Xu, 2012, explores the concept of "Design Anywhere, Manufacture Anywhere" (DAMA) in the automotive industry, highlighting the transformative role of cloud computing in facilitating this approach. It discusses two main forms of cloud adoption in manufacturing: traditional cloud computing and cloud manufacturing, the latter representing a more integrated and resource-efficient model. The study emphasizes the benefits of cloud computing, such as scalability, cost savings, and improved business processes, which enhance smart manufacturing practices. Additionally, it outlines the architecture of cloud manufacturing, including layers for manufacturing resources, virtual services, and application support, and examines the virtualization of resources and the need for effective integration in this context. Ultimately, the study illustrates how cloud technologies can revolutionize operational efficiency and collaboration in the manufacturing sector.

Hence, Cloud computing is used in manufacturing to enable flexible, scalable, and efficient operations, allowing companies to integrate various business processes and optimize resource management. It facilitates real-time data access and collaboration across locations, supporting the shift towards smart manufacturing and the DAMA philosophy (Xu, 2012)..

To summarize, cloud-based analytics in manufacturing enables real-time data processing and analysis, allowing for improved decision-making and operational efficiency. It also provides scalable storage and computational power, facilitating the integration of diverse data sources and advanced analytics tools to enhance productivity and reduce costs (Xu, 2020).

D. Deep Learning Algorithms

Artificial Neural Networks (ANNs) and Recurrent Neural Networks (RNNs) are both types of neural networks and both are used for manufacturing and various industrial applications, but there are key differences in them (Huang, 1994).

ANNs are typically feedforward networks, meaning data moves in one direction - from input nodes, through hidden layers, to output nodes. On the other hand, RNNs have connections that form cycles, allowing them to maintain a hidden state or memory. This structure enables RNNs to "remember" previous inputs, making them ideal for sequence-based data (Huang, 1994). These can be used to construct sequential controllers for manufacturing systems (Abdelhameed, 2002). CNNs are also heavily used in manufacturing automation. Different use cases are addressed, and in particular, deep learning has proved effective in failure prediction and tool ware (Xu, 2020). In combination with Cloud computing, neural network algorithms have evolved into very sophisticated technical setup in multiple factories across the globe (Xu, 2020). Another study (Franco-Sepúlveda, 2019) focuses on techniques used to optimize mining processes and on artificial neural networks (ANN), which are essential for predicting these processes. It highlights the applications and outcomes of these methods in various mining operations such as blasting, transportation, and mineral processing, areas where current models or techniques for prediction are not universally applicable across all mining complexes.

E. Keras on Apache Spark

Pumperla 2022, describes a breakthrough way of combining the power of distributed computing of Spark with Keras. Elephas is the API that introduces the SparkModel as its primary abstraction for integrating Keras models with Spark. To use it, one can define a Keras model, load the data, configure the SparkModel by specifying the desired training mode and update frequency, and then train the model on distributed data.

IV. RESEARCH METHODOLOGIES

The following key research methodologies have been used throughout the key papers that were evaluated for this research:

- Data Categorization and Evaluation: The review provides a consolidated overview and the categorization of data sources in manufacturing and evaluates the potential for analysis (e.g., ul Rehman, 2019; Sazu, 2022). This suggests the use of descriptive analysis and data categorization methodologies to organize and assess available data types.

- Case Studies: The Fiat study, as well as other industry examples, are used to showcase applications of big data and neural networks in real manufacturing contexts (e.g., Alexopoulos, 2022). This indicates the use of case study methodology.
- Experimental and Simulation-Based Analysis: Some studies employed neural networks for predictive analytics and defect detection using experimental setups, including CNN and RNN models, as well as the fabrication of synthetic data (e.g., Alexopoulos, 2022; Qi, 2019). This methodology includes simulation-based experiments and machine learning model experimentation (Elhoone, 2019).

V. LITERATURE REVIEW

A. Defining Big Data in Industrial Context

Intelligent manufacturing, driven by IoT, focuses on harnessing big data, characterized by its volume, variety, and velocity. However, challenges arise in effectively utilizing this data, which is often multi-source, multi-dimensional, noisy, imbalanced, and time series in nature. To address these issues, researchers propose frameworks and methodologies for integrating and analyzing manufacturing data, enhancing process monitoring, predictive maintenance, and decision-making capabilities (Wang, 2021).

B. Overview of Data Sources in Manufacturing

In order to understand the different applications of big data analytics in manufacturing systems, we first need to deeply understand the available data sources and evaluate the potential for their analysis. This can largely inform the opportunities in this field. In manufacturing and industry, Big Data Analytics (BDA) draws from a variety of data sources, which can be broadly categorized into the following types.

Sensors and IoT Devices, such as machine and equipment sensors. They collect data on equipment performance, temperature, pressure, vibration, and other physical conditions to monitor machinery health and operational efficiency. Additionally, Industrial IoT are smart devices embedded in production lines and logistics, generating real-time data on processes, asset tracking, and supply chain management. Although Industrial IoT and BDA have been widely studied separately, only a few studies have explored the convergence of the two domains (Lade, 2017). Big data production in the Internet of Things is evident due to large-scale deployment of sensing devices and systems in pervasive and ubiquitous industrial networks. Given that the concept of IoT systems is still evolving, complete integration and implementation of BDA processes in IoT systems are unavailable yet (ul Rehman, 2019).

Manufacturing Execution Systems (MES) are used to track and monitor production processes on the shop floor, collecting data related to work orders, production schedules, inventory levels, and quality control (Chang, 2019).

Computer-Aided Design (CAD) and Simulation Systems are another important source of data (Xu, 2020). CAD and digital simulation tools provide data on product designs, testing, and prototypes, allowing manufacturers to optimize designs and simulate potential production issues (Qi, 2019).

Data from supply chain management systems (SCM) covers everything from inventory levels and transportation routes to supplier performance and order tracking. This helps manufacturers optimize logistics and reduce bottlenecks. Additionally, data is being gathered on business operations, including procurement, finance, human resources, supply chain logistics, and order fulfillment (Li, 2021). This data helps with forecasting, resource allocation, and optimizing overall business processes.

Customer and Market Data are data from customer orders, preferences, behavior, and feedback. This can include sales data, e-commerce analytics, CRM systems, and market trends. Understanding this data helps tailor products and services to market demands (Sazu, 2022).

Quality Management Systems collect data on product quality, defects, and compliance with standards. Analytics helps identify process inefficiencies and areas where quality can be improved (Abdelhameed, 2002 and Alexopoulos, 2022).

Maintenance and Asset Management Systems: Data on equipment health and maintenance schedules, such as predictive maintenance systems, which analyze sensor data to predict failures and optimize repair schedules (Su, 2024).

Cameras and vision systems on production lines capture data for image analysis, which can be used for product quality checks, defect detection, and even process optimization through visual AI (Uğuroğlu, 2021).

Energy Management Systems: Data on energy usage in factories and production facilities. This helps track energy consumption patterns, enabling optimization of energy efficiency and cost reduction. (Li, 2021)

Importantly, there are also systems for monitoring air quality, emissions, and environmental factors to collect data to ensure compliance with safety and environmental regulations. These can be applied for health and safety as well as regulatory purposes and compliance (Rahul, 2023).

Finally, another source of data for manufacturing can be synthetic data. The fabrication of synthetic data that exhibits similar characteristics and similar distribution as the real data is a challenging task. The IBM Test Data Fabrication technology (TDF) was used for that purpose. TDF requires constraint rules that model the relationships and dependencies between the data and leverages a Constraint Satisfaction Problems (CSP) solver to fabricate data that satisfies these constraints. The rules for the production of synthetic data were set by CRF with the help of IBM. The correlation between the real parameters and the synthesized parameters was further refined after reiteration of the data analysis. (Alexopoulos, 2022)

C. Key Challenges in Big Data Analytics in Industry

Data integration, real-time analysis, storage issues, and actionable insights. Data collected from several sources can be disorganized and in different formats and data may not be exploited. While numerous authors highlight various challenges, a few key ones stand out, including big data imbalance, management, cleaning, analytics, aggregation, and accessibility, among others. (Rahul, 2023)

Another big challenge in applying deep learning for accurate analysis is the need for large volumes of labeled data, which is costly and time-consuming to obtain in many manufacturing contexts. For instance, identifying tool tip dynamics for a new tool requires hundreds of impact tests. Leveraging historical data to assist in training new models offers a promising solution (Xu, 2020).

Existing models and techniques are not universally applicable even within a specific industry, especially if the risks associated with incorrect predictions (monetary losses, safety risks) are very high (Franco-Sepúlveda, 2019). Hence, another challenge is that certain areas require completely novel, state of the art development of models and algorithms and sometimes it also requires acquisition of data that has not been collected before.

D. Applications of Big Data in Manufacturing and Industry

Predictive Maintenance: Big data analysis techniques can handle extensive data streams from multiple sources in real time, enabling PDM to maintain numerous devices, while reducing costs. These techniques continuously gather and scrutinize data using machine learning (ML) models to identify patterns for ongoing performance enhancement. The proposed platform includes the functions of automatic ML model selection and hyperparameter optimization, based on machine data and cross-validation results using random search (Su, 2024).

Quality Control and Defect Detection: Data analytics helps in improving product quality and minimizing defects. Combining Spark, Hadoop, and AWS with neural networks can achieve scalable results as demonstrated in numerous case studies (Uğuroğlu, 2021)

The study at the Fiat automobile manufacturing factory illustrates the architecture, which includes several key components. Data is transferred from CRF's internal server to the I-BinDaaS platform, where it is pre-processed, cleaned, and then used to train a complex neural network model implemented in PyTorch. The model outputs training metrics for visualization and a trained model for inference, with synthetic data used in early development phases to accelerate progress before real data is fully prepared. Based on the hypothesis that sensor data, thermal data, and process outcomes are correlated, an additional task is to classify combined image and sensor data to determine if the cast engine blocks are free of production defects. This data analysis is formally an M-ary supervised classification task. Given the dataset's image classification component, Deep Convolutional Neural Networks (CNNs) are employed for this task (Alexopoulos, 2022).

Smart Manufacturing (Industry 4.0): Smart manufacturing uses the Internet of Things (IoT) and big data to connect machines, sensors, and systems, enabling real-time monitoring and data-driven decision-making. IoT devices collect vast amounts of data from manufacturing processes, which are then analyzed using big data techniques to optimize efficiency, predict maintenance, and improve product quality (Lade, 2017). This integration allows for greater automation, flexibility, and responsiveness in production. It reduces downtime, enhances resource management, and increases overall productivity. The result is a more intelligent, adaptive, and efficient manufacturing process (ur Rehman, 2019).

E. Role of Neural Networks in Industrial Data Analytics

Neural networks for manufacturing have been used and gradually introduced since the early 1990s (Huang, 1994). According to Qi, 2019, an NN has strong evaluating skills for representing complex, highly nonlinear relationships between input and output features, and it has been shown that a network with only one hidden layer but sufficient neurons can express an arbitrary function.

Neural networks, including CNNs and RNNs, excel in modeling complex, non-linear data, with CNNs suited for visual inspections and RNNs handling time-series analysis and anomaly detection. Ensemble methods like boosting and bagging, particularly XGBoost, improve prediction accuracy in PDM, especially with imbalanced datasets. Integrating machine learning algorithms into industrial IoT systems enables real-time equipment monitoring, enhancing predictive maintenance efficiency. Advances like CNN-LSTM networks and AutoGluon further improve cost efficiency and machine reliability by predicting failures with greater accuracy. (Su, 2024)

Some of the key use cases for using neural networks in manufacturing can be summarized as: predictive analytics, anomaly detection, fault diagnosis, image recognition (for defects), and quality prediction (Li, 2021). Neural networks have been extensively used in manufacturing design (for example, in additive manufacturing Qi, 2019), in situ process monitoring (Qi, 2019), planning and scheduling, quality and health management, industrial big governance (Wang, 2021). For example, Elhoone, 2019, presented a study where a data-driven smart agent system was developed that dynamically identifies optimal additive manufacturing techniques for digital designs over the cyber network. Three sub-systems were developed which include the artificial neural network-based expert system, cyber-interface IoT simulator and dynamic machine identification system. Multiple other studies (Li, 2016; Ooi, 2018; Qi, 2019) also show-cased how a combination of big data, neural networks, and IoT concepts can drive an end-to-end manufacturing process and improve design, machine utilization, etc.

VI. CRITICAL EVALUATION

Vast experimental research and practical implementations exist for the use of Big Data in the manufacturing and industrial context. Nonetheless, the adoption of Big Data Analytics coupled with Industrial IoT presents numerous research opportunities, in addition to the use cases that have already been explored.

First, automation and AI will optimize industrial processes, making AI a core component for managing complex datasets. Future IIoT systems will handle vast amounts of data from both internal and external operations, with AI optimizing and analyzing it.

Secondly, human–machine interaction will evolve through augmented reality and wearable computing, resulting in more intuitive interfaces and autonomous systems. Additionally, cybersecurity, privacy, and ethics will become critical as BDA helps detect threats, unauthorized access, and data misuse in real-time.

However, universal standards are needed to define how data should be collected, secured, and shared across industries, ensuring ethical data practices. New protocols are required to ensure interoperability among industries, addressing data heterogeneity and computing challenges. Additionally, there's an opportunity to develop an end-to-end analytics pipeline to manage data from various sources, enhancing knowledge discovery. BDA will also drive precision manufacturing by using customer data to design more tailored products. However, integrating BDA into IIoT systems remains challenging, requiring real-time, interactive applications across industries. Concentric computing systems, which support data processing near its source, will improve efficiency and reduce operational costs. Emerging technologies like containerization and microservices will help manage the complexities of BDA in IIoT. Finally, technologies like fog computing and blockchain will enhance local data processing and secure storage, supporting real-time analytics and decentralizing data management in IIoT systems. (ur Rehman, 2019).

In addition, despite the multiple successful applications of big data and neural networks in the manufacturing industry, According to Huang, 1994, neural networks offer several advantages over knowledge-based expert systems, notably their learning ability and parallel structure, which attract researchers in AI. However, some expectations, like fully replacing conventional computers and eliminating programming, are unrealistic due to several disadvantages:

1. Neural networks lack explicit explanations for their results, making their user interfaces less user-friendly compared to expert systems.
2. Their knowledge representation is vague and hard to interpret.
3. Configuring a neural network is time-consuming, often requiring trial and error to identify the right architecture.
4. Current learning algorithms are inefficient and lack guaranteed convergence.
5. Optimal training set derivation remains an unresolved issue.

Experience shows that the ability to provide explanations is crucial for user acceptance of AI systems.

These concerns are repeatedly addressed in further research with limited real solutions proposed to date.

Given this cross-platform integration, IIoT systems need to ensure interoperability, virtualisation, decentralization, real-time capability, service orientation, modularity and security across all verticals. This integration demands a robust platform capable of optimizing the processes related to manufacturing, including predictive maintenance and quality assurance. The described existing processes often do not offer a holistic end-to-end approach, or if they do, they sometimes use outdated systems. Human resources required to develop and maintain these systems are often an issue.

One of the major ethical concerns we can foresee is the elimination of human labor and the loss of jobs in the manufacturing sector due to excessive automation. In particular, the concept of 'dark factory' eliminates human intelligence out of the manufacturing cycle. While this would decrease the manufacturing costs, this would also eliminate the well-being of the people who are supposed to be buying the manufactured goods, thus eliminating the consumer base.

According to our research, this concern is being addressed. Industry 5.0 builds on Industry 4.0 by integrating human intelligence and creativity into the automation and digitalization of manufacturing systems, creating more human-centric and resilient environments. A key concept, "human-in-the-loop", emphasizes collaboration between humans and automated systems to improve decision-making, problem-solving, and adaptability in smart manufacturing through real-time data analytics and collaborative automation (Su, 2024).

Cloud manufacturing allows for centralized management of distributed manufacturing resources, enabling multiple users to request services simultaneously and promoting sustainable production practices. However, various implementation architectures exist, such as public, private, community, and hybrid clouds, each with its own complexities and challenges that require further study and innovation.

VII. CONCLUSION

To summarize, integration of Big Data technologies and neural networks into Industry 4.0 manufacturing systems has been transforming the sector, offering good opportunities for optimization, predictive maintenance, and automation. Technologies like Hadoop, Apache Spark, and cloud computing provide the infrastructure for handling vast amounts of data, and then neural networks offer the sophisticated tools capable of driving real-time decision-making and improving operational efficiency.

However, challenges such as data integration, system interoperability, and ethical concerns around automation and job displacement are becoming more prominent. As the industry moves toward Industry 5.0, a more human-centric approach combining human creativity with advanced automation is being proposed, ensuring that technology and human intelligence work hand-in-hand.

We would propose further research to address the complexities of cloud architecture to support data analytics in real time and in production systems, ethical and privacy concerns, and real-time analytics in the evolving landscape of smart manufacturing.

VIII. APPENDIX

The table below provides a comprehensive high level overview of big data analytics in the manufacturing industry, focusing on the specified technologies. This help create a mental framework for working with data analysis in the manufacturing context.

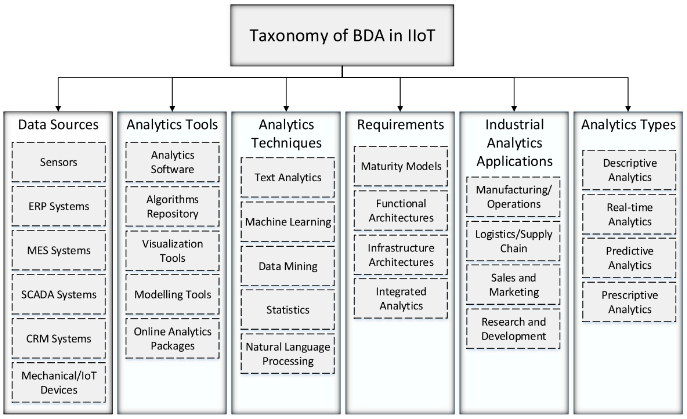


Fig. 1. Data sources of Big Data in Manufacturing and Industry (ur Rehman et al, 2019)

IX. EXPERIMENTAL SETUP

The below describes the experimental part of this review, where an attempt was made to use Apache Spark, Amazon EMR (with Hadoop), and Keras to simulate a scenario of predicting impurities of iron ore being mined.

A. Setup

Hardware & Software

- MacBook Pro
- 2.3 GHz Dual-Core Intel Core i5
- 8 GB 2133 MHz LPDDR3
- Apache Spark version 3.5.2 (prerequisite - Java 8)
- Jupyter
- Keras version 2.14.0
- Amazon EMR (Hadoop and Spark images were used) running 2xlarge default node instance types

Dataset:

Mining Process Flotation Plant Database under the C0: Public Domain License

Code for the Experiment:

https://github.com/sbs24043/assignments/tree/main/ca3

https://github.com/CCT-Dublin/integrated-ca1-sem-2-sbs24043.git

B. Keras Neural Network experimentation

We have created an Experiment class in order to be able to run different experiments with different hyperparameters, tuning one parameter at a time and capturing the same set of measurements and graphs. Additionally an experiment was performed on ANN vs RNN to compare and contrast the results and evaluate the difficulty of hyperparameter tuning. ANNs are general-purpose, and work with a variety of data types. CNNs are best suited for spatial data (e.g., images) and hence in this experiment CNNs were skipped. RNNs are designed for sequential data (e.g., time series, text), however, we would like to apply those to the experiment as well and compare them to ANN on the very tabular dataset. For this particular experiment we use LSTM which are a type of RNN, and they expose a rich set of parameters which can be fine tuned.

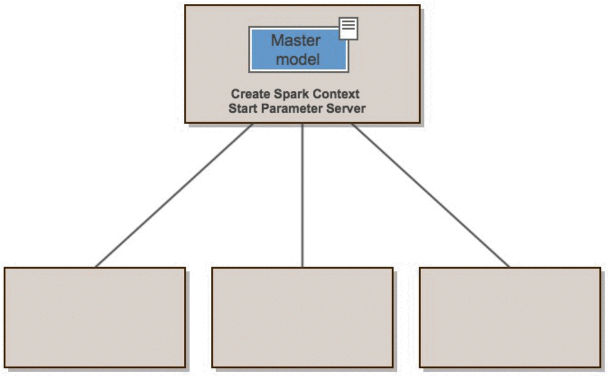


Fig. 2. Keras on Apache Spark (Pumperla, 2015)

Below are the initial results for the first round of experimentation with sigmoid activation function and adagrad optimizer.

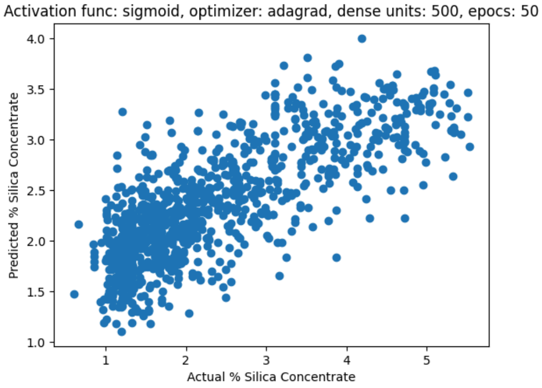


Fig. 3. ANN predicted vs actual labels scatterplot.

A residual plot is a graphical tool used to assess the performance and fit of a regression model by plotting the residuals (errors) on the y-axis against the predicted values or another variable on the x-axis:

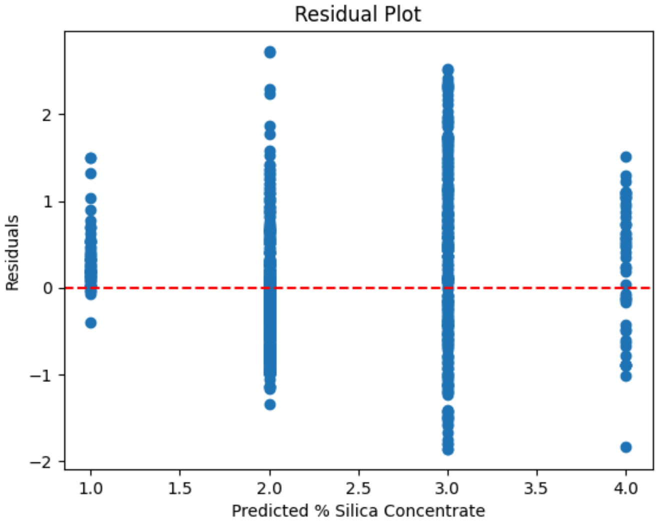


Fig. 4. ANN predicted vs actual labels residual plot.

When it comes to model evaluation, a commonly used evaluation metric for regression tasks is the mean squared error (MSE), which calculates the average of the squared differences between actual and predicted values. The MSE assesses how closely predictions match the true values, placing greater emphasis on larger errors. However, MSE is sensitive to outliers, meaning a few extreme values can disproportionately affect the result. In such cases, the mean absolute error may be a better option. MAE computes the average of the absolute differences between actual and predicted values, treating all errors equally. While more resistant to outliers, MAE is less sensitive to smaller errors. Hence, below we are plotting MAE and RMSE. Accuracy is consistently at 0, because this is a *regression problem* and it is not being computed, instead MAE and RMSE are to be used:

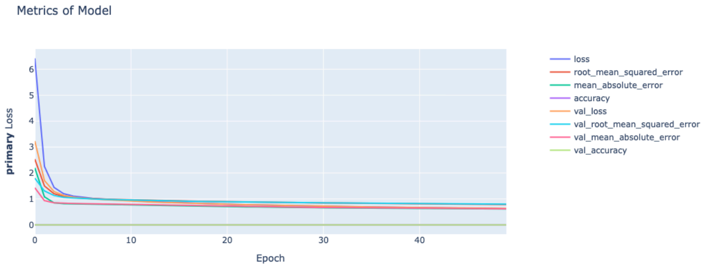


Fig. 5. ANN training metrics

Over multiple iterations we have been able to select the set of hyperparameters that have the best accuracy and showed the least loss. For our particular dataset, relu activation function with adam optimizer showed the best results. Increasing the number of training epochs and the number of nodes helped increase the accuracy and decrease the loss, however, the graph below also show that after some number of epochs the loss oscillates around particular values and does not decrease any further.

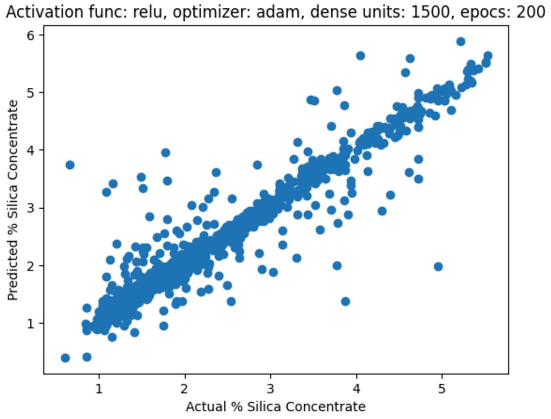


Fig. 6. Post fine-tuning: ANN predicted vs actual labels scatterplot.

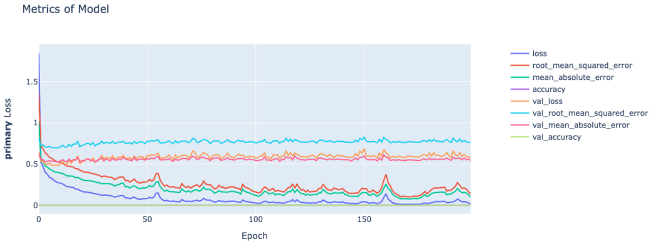


Fig. 7. Post fine-tuning: ANN training metrics

After selecting the best activation function and the next optimizer, we can keep fine-tuning the layer activation functions and especially the optimizer which exposes a lot of configurable parameters:

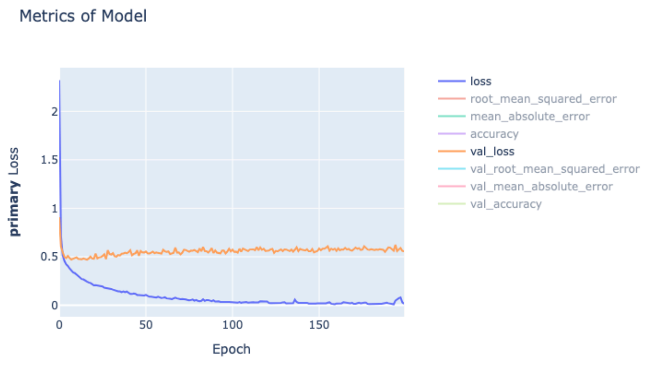


Fig. 8. Keras optimizer's learning rate adjustment smoothes the loss graph

An interesting observation while using the Keras library is that it has split the dataset into training and validation under the hood, hence this step can be avoided during the data preparation.

Below are the loss and validation loss for our best ANN candidate:

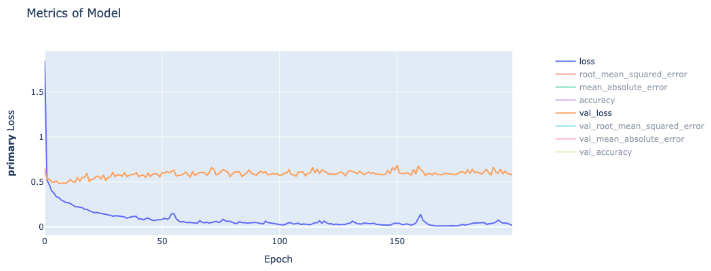


Fig. 9. Fine tuned ANN Loss vs Value Loss

Finally, below is the RNN (LSTM) after hyper parameter tuning, which show results comparable to ANN:

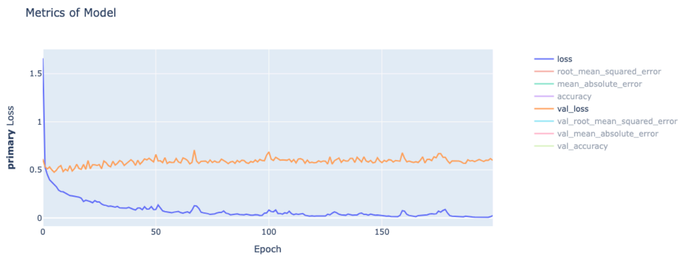


Fig. 10. Fine tuned RNN Loss vs Value Loss

Hence, we can see that our ANN performed well for our regression problem and the use of an RNN would have been excessive.

We observe that our model is prone to overfitting, as the validation data has a high loss of over 0.5 as compared to 0.1 on the training data. Hence, to remediate the problem we have also applied drop outs and regularization. Another step that could be taken to fix the model is normalizing the input data.

Using sigmoid activation function for the input layer and keeping adagrad as an optimizer seemed to show the next results:

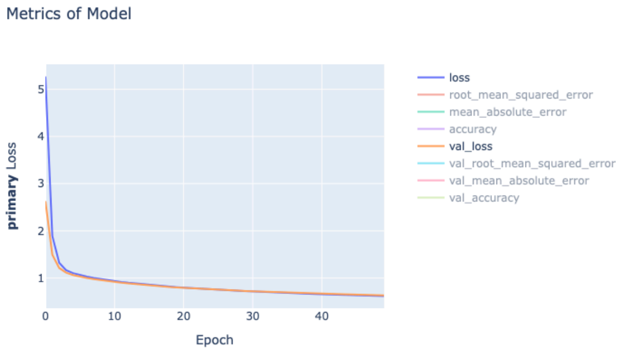
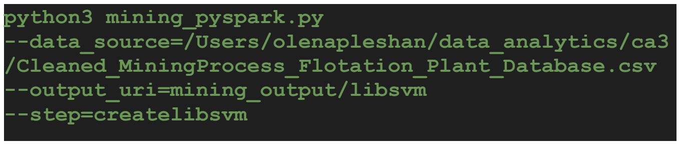


Fig. 11. ANN with sigmoid input layer activation and adagrad optimizer.

C. Neural Network Using SparkML & Multilayer Perceptron

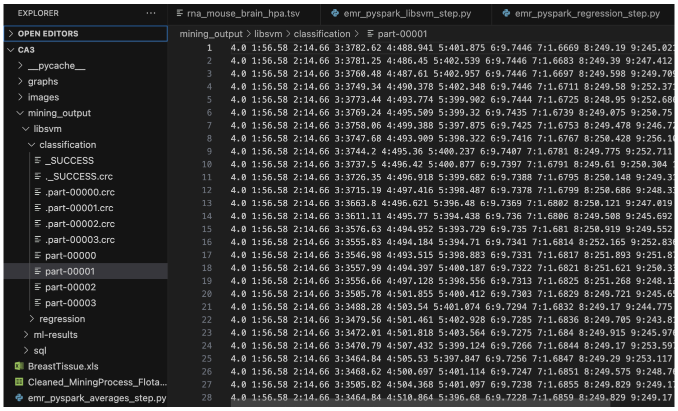
The native way of creating neural networks on Apache Spark is using MultiLayer Perceptron. Classifier trainer based on the Multilayer Perceptron. Each layer has a sigmoid activation function, the output layer has softmax. Number of inputs has to be equal to the size of feature vectors. Number of outputs has to be equal to the total number of labels (spark.apache.org, n.d.). Spark's MLLib is suitable when you're doing relatively simple ML on a large data set. However, in order to perform a successful experiment, data had to also be converted into LibSVM format, properly scaled, ect before it was ready to consume by SparkML, hence a lot of preparatory work was required. Fine tuning seems to be more limited and requires further exploration, and the Spark implementation only allows for classification problems, and not regressions.

Preparing the data: In Spark, the LibSVM format is a popular file format used for representing labeled sparse data, typically for machine learning tasks such as classification and regression. This format is supported by the MLlib library in Apache Spark, and it's widely used because of its simplicity and efficiency in handling sparse datasets. To create libSVM files:



```
python3 mining_pyspark.py
--data_source=/Users/olenapleshan/data_analytics/ca3
/Cleaned_MiningProcess_Flotation_Plant_Database.csv
--output_uri=mining_output/libsvm
--step=createlibsvm
```

Here is the example of the output received for that preprocessing:



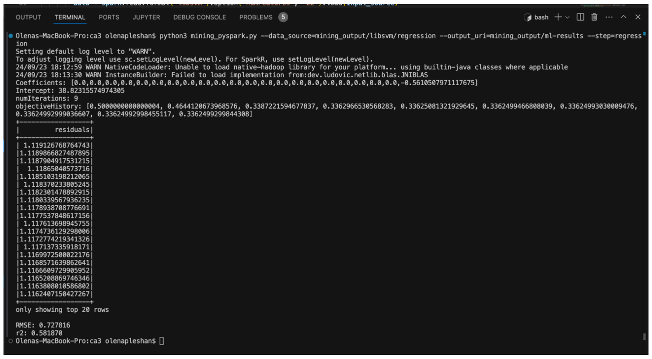
```
4.0 1:56.58 2:14.66 3:3782.62 4:488.941 5:401.875 6:9.7446 7:1.6669 8:249.19 9:245.02
4.0 1:56.58 2:14.66 3:3781.25 4:486.45 5:402.539 6:9.7446 7:1.6683 8:249.39 9:247.412
4.0 1:56.58 2:14.66 3:3760.48 4:487.61 5:402.957 6:9.7446 7:1.6697 8:249.598 9:249.70
4.0 1:56.58 2:14.66 3:3749.34 4:490.378 5:402.348 6:9.7446 7:1.6711 8:249.58 9:252.37
4.0 1:56.58 2:14.66 3:3773.44 4:493.774 5:399.902 6:9.7444 7:1.6725 8:248.95 9:252.68
4.0 1:56.58 2:14.66 3:3769.24 4:495.509 5:399.32 6:9.7435 7:1.6739 8:249.075 9:250.75
4.0 1:56.58 2:14.66 3:3758.06 4:499.388 5:397.875 6:9.7425 7:1.6753 8:249.478 9:246.72
4.0 1:56.58 2:14.66 3:3747.68 4:493.909 5:398.322 6:9.7416 7:1.6767 8:250.428 9:256.10
4.0 1:56.58 2:14.66 3:3744.2 4:495.36 5:400.237 6:9.7407 7:1.6781 8:249.775 9:252.711
4.0 1:56.58 2:14.66 3:3737.5 4:496.42 5:400.877 6:9.7397 7:1.6791 8:249.61 9:250.304
4.0 1:56.58 2:14.66 3:3726.35 4:496.918 5:399.682 6:9.7388 7:1.6795 8:250.148 9:249.31
4.0 1:56.58 2:14.66 3:3715.19 4:497.416 5:398.487 6:9.7378 7:1.6799 8:250.686 9:248.33
4.0 1:56.58 2:14.66 3:3663.8 4:496.621 5:396.48 6:9.7369 7:1.6802 8:250.121 9:247.019
4.0 1:56.58 2:14.66 3:3611.11 4:495.77 5:394.438 6:9.736 7:1.6806 8:249.508 9:245.692
4.0 1:56.58 2:14.66 3:3576.63 4:494.952 5:393.729 6:9.735 7:1.681 8:250.919 9:249.552
4.0 1:56.58 2:14.66 3:3555.83 4:494.184 5:394.71 6:9.7341 7:1.6814 8:252.165 9:252.83
4.0 1:56.58 2:14.66 3:3546.98 4:493.515 5:398.883 6:9.7331 7:1.6817 8:251.893 9:251.87
4.0 1:56.58 2:14.66 3:3557.99 4:494.397 5:400.187 6:9.7322 7:1.6821 8:251.621 9:250.33
4.0 1:56.58 2:14.66 3:3556.66 4:497.128 5:398.556 6:9.7313 7:1.6825 8:251.268 9:248.13
4.0 1:56.58 2:14.66 3:3505.78 4:501.855 5:400.412 6:9.7303 7:1.6829 8:249.721 9:245.65
4.0 1:56.58 2:14.66 3:3488.28 4:503.54 5:401.074 6:9.7294 7:1.6832 8:249.17 9:244.775
4.0 1:56.58 2:14.66 3:3479.56 4:501.461 5:402.928 6:9.7285 7:1.6836 8:249.705 9:243.81
4.0 1:56.58 2:14.66 3:3472.01 4:501.818 5:403.564 6:9.7275 7:1.684 8:249.915 9:245.97
4.0 1:56.58 2:14.66 3:3470.79 4:507.432 5:399.124 6:9.7266 7:1.6844 8:249.17 9:253.59
4.0 1:56.58 2:14.66 3:3464.84 4:505.53 5:397.847 6:9.7256 7:1.6847 8:249.29 9:253.117
4.0 1:56.58 2:14.66 3:3468.62 4:500.697 5:401.114 6:9.7247 7:1.6851 8:249.575 9:248.76
4.0 1:56.58 2:14.66 3:3505.82 4:504.368 5:401.097 6:9.7238 7:1.6855 8:249.829 9:249.17
4.0 1:56.58 2:14.66 3:3464.84 4:510.864 5:396.68 6:9.7228 7:1.6859 8:249.829 9:249.17
```

Fig. 11. Using Apache Spark and SparkML requires a separate data preparation step that is different to Leras or SkicitLearn

Linear Regression: We are dealing with a regression problem here, but unfortunately, we cannot create a neural network in Spark for regression. Hence, below, are the results for a Linear Regression model using Apache Spark. To train Linear Regression model:

```
python3 mining_pyspark.py
--data_source=mining_output/libsvm/regression
--output_uri=mining_output/ml-results
--step=regression
```

The output can be seen in screenshot below:



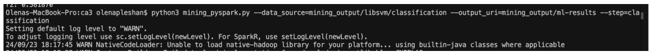
```
Olenas-MacBook-Pro:ca3 olenapleshan$ python3 mining_pyspark.py --data_source=mining_output/libsvm/regression --output_uri=mining_output/ml-results --step=regress
ion
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use setLogLevel(newLevel).
24/09/23 18:12:59 WARN NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
24/09/23 18:13:30 WARN InstanceBuilder: Failed to load implementation from:dev.ludovic.netlib.blas.JNIBLAS
Coefficients: [0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,-0.5610507971117675]
Intercept: 38.82315574974305
numIterations: 9
objectiveHistory: [0.5000000000000004, 0.4644120673968576, 0.3387221594677837, 0.3362966530568283, 0.33625081321929645, 0.3362499466808039, 0.33624993030009476,
0.33624992999036607, 0.33624992998455117, 0.3362499299844308]
+------------------+
|         residuals|
+------------------+
| 1.119126768764743|
|1.1189866827487895|
|1.1187904917531215|
|  1.11865040573716|
|1.1185103198212065|
| 1.118370233805245|
|1.1182301478892915|
|1.1180339567936235|
|1.1178938708776691|
|1.1177537848617156|
| 1.117613698945755|
|1.1174736129298006|
|1.1172774219341326|
| 1.117137335918171|
|1.1169972500022176|
|1.1168571639862641|
|1.1166609729905952|
|1.1165208869746346|
|1.1163808010586802|
|1.1162407150427267|
+------------------+
only showing top 20 rows

RMSE: 0.727816
r2: 0.581870
Olenas-MacBook-Pro:ca3 olenapleshan$ 
```

Fig. 12. Using Apache Spark to train a linear regression model for iron purity mining problem

Multilayer Perceptron: Next, we have to turn the regression problem into a classification problem. Since our dataset deals with iron production impurities (defects), we have broken down the problem into a classification based on the level of impurity. To train Multilayer Perceptron:

```
python3 mining_pyspark.py
--data_source=mining_output/libsvm/classification
--output_uri=mining_output/ml-results
--step=classification
```

Below are the outputs of the classification:



```
r2: 0.581870
Olenas-MacBook-Pro:ca3 olenapleshan$ python3 mining_pyspark.py --data_source=mining_output/libsvm/classification --output_uri=mining_output/ml-results --step=cla
ssification
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use setLogLevel(newLevel).
24/09/23 18:17:45 WARN NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
```

Fig. 13. Apache Spark data processing in stages. Running Classification.

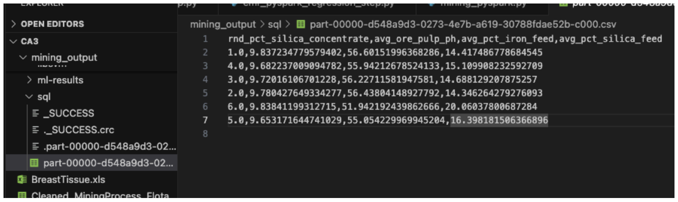
Since we are dealing with a big dataset, it is useful to save the results of the pre-trained model for future use for inferences. This can be achieved using the `MLUtils.saveAsLibSVMFile` function. This will save the results of the determined model weights and other metadata into a file. Then, in a production setup, this file can be re-instantiated into a model instead of re-training it from scratch.

D. Using Spark SQL

Using SQL instead of creating a machine learning (ML) model can be more useful in situations where the task is relatively straightforward, relies heavily on predefined rules, or requires real-time querying and reporting. So, we can also run aggregations and use SQL :

```
python3 mining_pyspark.py
--data_source=/Users/olenapleshan/data_analytics/ca3
/Cleaned_MiningProcess_Flotation_Plant_Database.csv
--output_uri=mining_output/sql --step=averages
```

and this produces aggregated data such as:


OPEN EDITORS
CA3
mining_output
ml-results
sql
_SUCCESS
._SUCCESS.crc
.part-00000-d548a9d3-02...
part-00000-d548a9d3-02...
BreastTissue.xls
Cleaned_MiningProcess_Flota
mining_output > sql > part-00000-d548a9d3-0273-4e7b-a619-30788fdae52b-c000.csv

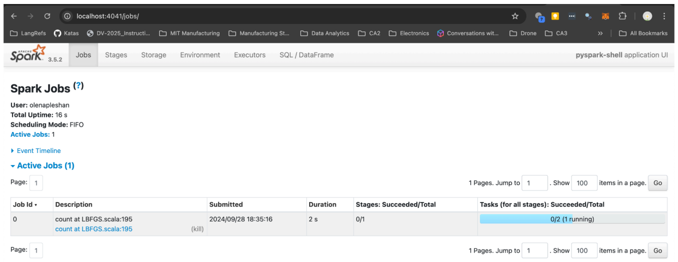
```
rnd_pct_silica_concentrate,avg_ore_pulp_ph,avg_pct_iron_feed,avg_pct_silica_feed
1.0,9.837234779579402,56.60151996368286,14.417486778684545
4.0,9.682237009094782,55.94212678524133,15.109908232592709
3.0,9.72016106701228,56.22711581947581,14.688129207875257
2.0,9.780427649334277,56.43804148927792,14.346264279276093
6.0,9.83841199312715,51.942192439862666,20.06037800687284
5.0,9.653171644741029,55.054229969945204,16.398181506366896
```

Fig. 14. Outputs of an SQL query on Apache Spark

This could be turned into a monitoring Dashboard or used for high level decision making.

E. Spark and Hadoop on Amazon EMR

Using Amazon S3 can be fairly expensive. Hence, before launching any jobs there, we first developed and tested a script locally. Local run of Spark creates the job on localhost, where longer-running or multi-step jobs can be inspected.



Spark Jobs (?)

User: olenapleshan
Total Uptime: 16 s
Scheduling Mode: FIFO
Active Jobs: 1

▸ Event Timeline

▾ Active Jobs (1)

Page: 1 — 1 Pages. Jump to 1 . Show 100 items in a page. Go

Job Id ▾	Description	Submitted	Duration	Stages: Succeeded/Total	Tasks (for all stages): Succeeded/Total
0	count at LBFGS.scala:195 count at LBFGS.scala:195 (kill)	2024/09/28 18:35:16	2 s	0/1	0/2 (1 running)

Page: 1 — 1 Pages. Jump to 1 . Show 100 items in a page. Go

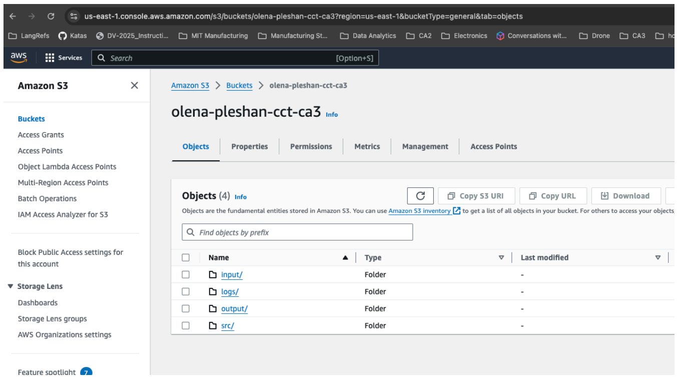
Fig. 15. Running Apache Spark job locally

The script created as part of this experiment, has functions with commands on how to execute them locally. Then the script was broken out into multiple scripts so it is easier to debug the jobs once submitted to EMR.

In order to set up the Python scripts developed locally and run them on EMR, we followed the official Amazon tutorial (docs.aws.amazon.com, n.d.). is a managed cluster platform that simplifies running big data frameworks, such as Apache Hadoop and Apache Spark, on AWS to process and analyze vast amounts of data.

1. Uploading the data into Amazon S3

In Amazon EMR, data files need to be uploaded into Amazon S3.


us-east-1.console.aws.amazon.com/s3/buckets/olena-pleshan-cct-ca3?region=us-east-1&bucketType=general&tab=objects

Amazon S3 > Buckets > olena-pleshan-cct-ca3

olena-pleshan-cct-ca3 Info

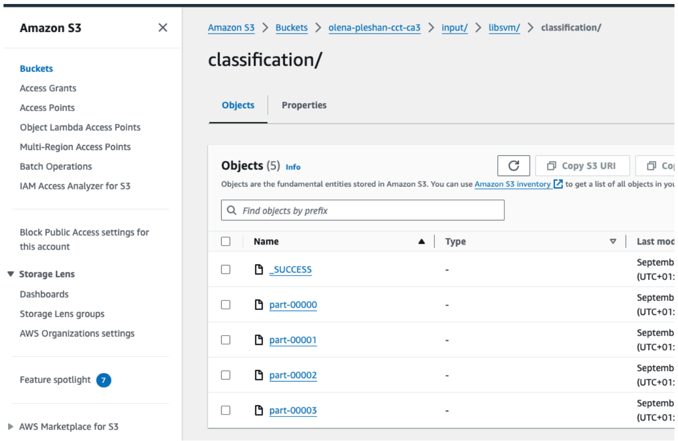
Objects (4) Info

Objects are the fundamental entities stored in Amazon S3. You can use Amazon S3 inventory to get a list of all objects in your bucket. For others to access your objects

Name	Type	Last modified
input/	Folder	-
logs/	Folder	-
output/	Folder	-
src/	Folder	-

Fig. 16. Creating data input, output, logs and src (for application code files) in the s3 bucket.

Amazon EMR and Hadoop provide a variety of file systems that you can use when processing cluster steps. You specify which file system to use by the prefix of the URI used to access the data. For example, `s3://amzn-s3-demo-bucket1/path` references an Amazon S3 bucket using EMRFS.


Amazon S3 > Buckets > olena-pleshan-cct-ca3 > input/ > libsvm/ > classification/

classification/

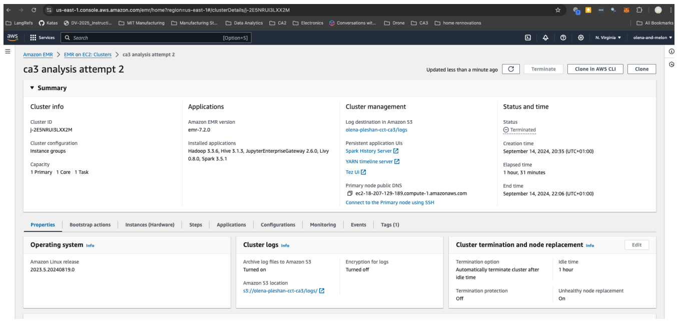
Objects (5) Info

Objects are the fundamental entities stored in Amazon S3. You can use Amazon S3 inventory to get a list of all objects in you

Name	Type	Last mod
_SUCCESS	-	Septemb (UTC+01:
part-00000	-	Septemb (UTC+01:
part-00001	-	Septemb (UTC+01:
part-00002	-	Septemb (UTC+01:
part-00003	-	Septemb (UTC+01:

Fig. 17. Input data gets processed by Spark into LibSVM format ready for consumption by SparkML

2. Creating an EMR cluster


us-east-1.console.aws.amazon.com/emr/home?region=us-east-1#/clusterDetails/j-2E5NRUI3LXX2M

Amazon EMR > EMR on EC2: Clusters > ca3 analysis attempt 2

ca3 analysis attempt 2

Updated less than a minute ago

Summary

Cluster info
Cluster ID
j-2E5NRUI3LXX2M
Cluster configuration
Instance groups
Capacity
1 Primary | 1 Core | 1 Task

Applications
Amazon EMR version
emr-7.2.0
Installed applications
Hadoop 3.3.6, Hive 3.1.3, JupyterEnterpriseGateway 2.6.0, Livy 0.8.0, Spark 3.5.1

Cluster management
Log destination in Amazon S3
olena-pleshan-cct-ca3/logs
Persistent application UIs
Spark History Server
YARN timeline server
Tez UI
Primary node public DNS
ec2-18-207-129-189.compute-1.amazonaws.com
Connect to the Primary node using SSH

Status and time
Status
Terminated
Creation time
September 14, 2024, 20:35 (UTC+01:00)
Elapsed time
1 hour, 31 minutes
End time
September 14, 2024, 22:06 (UTC+01:00)

Operating system Info
Amazon Linux release
2023.5.20240819.0

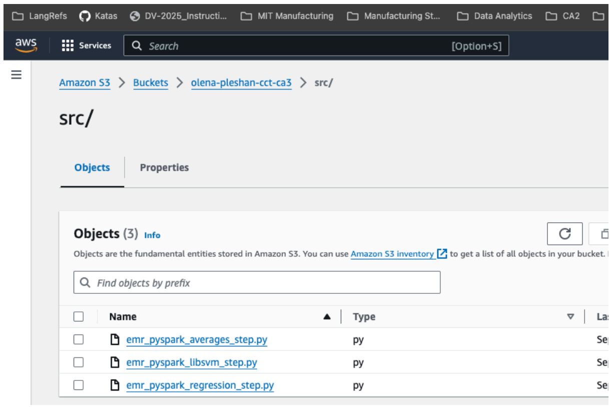
Cluster logs Info
Archive log files to Amazon S3
Turned on
Amazon S3 location
s3://olena-pleshan-cct-ca3/logs/
Encryption for logs
Turned off

Cluster termination and node replacement Info
Termination option
Automatically terminate cluster after idle time
Idle time
1 hour
Termination protection
Off
Unhealthy node replacement
On

Fig. 18. Created EMR cluster pointing to the scripts S3 folder and giving the right permissions to the bucket

3. Defining and running steps (jobs)

In S3 bucket, we need to put the Spark scripts (Apps), which can be written in different languages, but Python in our case. There will be three steps which will be performed on the prepared EMR cluster: some data aggregations using SQL, then preparing data in ML lib, and finally doing a simple regression. Classification model was run locally and regression is for EMR demo purposes only.


Amazon S3 > Buckets > olena-pleshan-cct-ca3 > src/

src/

Objects (3) Info

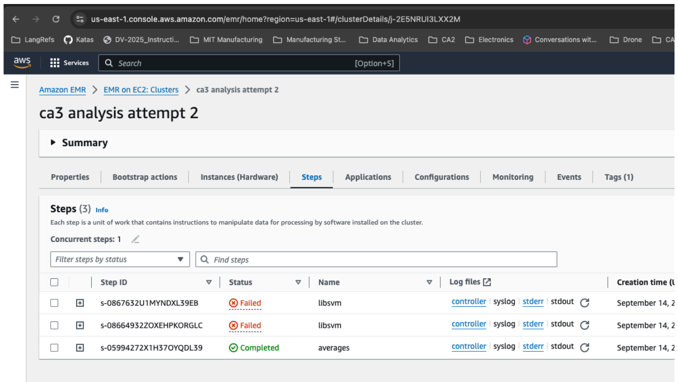
Objects are the fundamental entities stored in Amazon S3. You can use Amazon S3 inventory to get a list of all objects in your bucket.

Name	Type	La
emr_pyspark_averages_step.py	py	Se
emr_pyspark_libsvm_step.py	py	Se
emr_pyspark_regression_step.py	py	Se

Fig. 19. Scripts for each EMR step

4. Running the scripts, inspect the logs and outputs

After launching a cluster, we can submit work to the running cluster for processing and analyzing data. A step is a unit of work consisting of one or more actions. For example, in our case we defined steps by pointing at the scripts in the S3 buckets. Those steps, before running them on the cluster, were first tested locally, using a local instance of Apache Spark.



ca3 analysis attempt 2

▸ Summary

Steps (3) Info

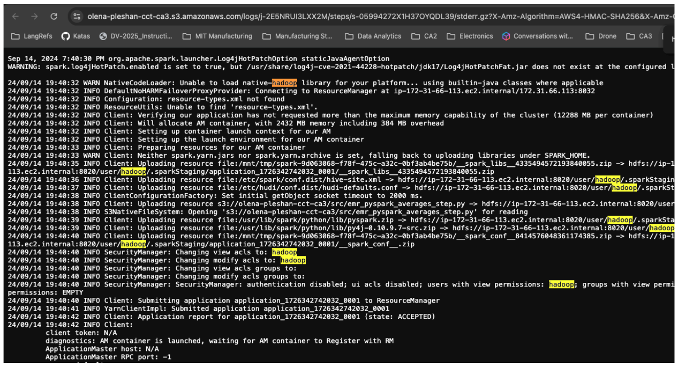
Each step is a unit of work that contains instructions to manipulate data for processing by software installed on the cluster.

Concurrent steps: 1

□		Step ID	Status	Name	Log files	Creation time (U
□	⊞	s-0867632U1MYNDXL39EB	⊗ Failed	libsvm	controller | syslog | stderr | stdout	September 14, 2
□	⊞	s-08664932ZOXEHPKORGLC	⊗ Failed	libsvm	controller | syslog | stderr | stdout	September 14, 2
□	⊞	s-05994272X1H37OYQDL39	✓ Completed	averages	controller | syslog | stderr | stdout	September 14, 2

Fig. 20. Break out the Spark job into steps. Since our algorithm has steps that depend on each other (e.g., creating libSVM data before proceeding to ML), these step can be executed and retried independently and sequentially.

In the job logs, we can observe this being run on Hadoop and HDFS.


olena-pleshan-cct-ca3.s3.amazonaws.com/logs/j-2E5NRUI3LXX2M/steps/s-05994272X1H37OYQDL39/stderr.gz?X-Amz-Algorithm=AWS4-HMAC-SHA256&X-Amz-
LangRefs
Katas
DV-2025_Instructi...
MIT Manufacturing
Manufacturing St...
Data Analytics
CA2
Electronics
Conversations wit...
Drone
CA3

```
Sep 14, 2024 7:40:30 PM org.apache.spark.launcher.Log4jHotPatchOption staticJavaAgentOption
WARNING: spark.log4jHotPatch.enabled is set to true, but /usr/share/log4j-cve-2021-44228-hotpatch/jdk17/Log4jHotPatchFat.jar does not exist at the configured l

24/09/14 19:40:32 WARN NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
24/09/14 19:40:32 INFO DefaultNoHARMFailoverProxyProvider: Connecting to ResourceManager at ip-172-31-66-113.ec2.internal/172.31.66.113:8032
24/09/14 19:40:32 INFO Configuration: resource-types.xml not found
24/09/14 19:40:32 INFO ResourceUtils: Unable to find 'resource-types.xml'.
24/09/14 19:40:32 INFO Client: Verifying our application has not requested more than the maximum memory capability of the cluster (12288 MB per container)
24/09/14 19:40:32 INFO Client: Will allocate AM container, with 2432 MB memory including 384 MB overhead
24/09/14 19:40:32 INFO Client: Setting up container launch context for our AM
24/09/14 19:40:32 INFO Client: Setting up the launch environment for our AM container
24/09/14 19:40:33 INFO Client: Preparing resources for our AM container
24/09/14 19:40:33 WARN Client: Neither spark.yarn.jars nor spark.yarn.archive is set, falling back to uploading libraries under SPARK_HOME.
24/09/14 19:40:35 INFO Client: Uploading resource file:/mnt/tmp/spark-9d063068-f78f-475c-a32c-0bf3ab4be75b/__spark_libs__4335494572193840055.zip -> hdfs://ip-1
113.ec2.internal:8020/user/hadoop/.sparkStaging/application_1726342742032_0001/__spark_libs__4335494572193840055.zip
24/09/14 19:40:36 INFO Client: Uploading resource file:/etc/spark/conf.dist/hive-site.xml -> hdfs://ip-172-31-66-113.ec2.internal:8020/user/hadoop/.sparkStagin
24/09/14 19:40:37 INFO Client: Uploading resource file:/etc/hudi/conf.dist/hudi-defaults.conf -> hdfs://ip-172-31-66-113.ec2.internal:8020/user/hadoop/.sparkSt
24/09/14 19:40:38 INFO ClientConfigurationFactory: Set initial getObject socket timeout to 2000 ms.
24/09/14 19:40:38 INFO Client: Uploading resource s3://olena-pleshan-cct-ca3/src/emr_pyspark_averages_step.py -> hdfs://ip-172-31-66-113.ec2.internal:8020/user
24/09/14 19:40:38 INFO S3NativeFileSystem: Opening 's3://olena-pleshan-cct-ca3/src/emr_pyspark_averages_step.py' for reading
24/09/14 19:40:39 INFO Client: Uploading resource file:/usr/lib/spark/python/lib/pyspark.zip -> hdfs://ip-172-31-66-113.ec2.internal:8020/user/hadoop/.sparkSta
24/09/14 19:40:39 INFO Client: Uploading resource file:/usr/lib/spark/python/lib/py4j-0.10.9.7-src.zip -> hdfs://ip-172-31-66-113.ec2.internal:8020/user/hadoop
24/09/14 19:40:40 INFO Client: Uploading resource file:/mnt/tmp/spark-9d063068-f78f-475c-a32c-0bf3ab4be75b/__spark_conf__8414576048361174385.zip -> hdfs://ip-1
113.ec2.internal:8020/user/hadoop/.sparkStaging/application_1726342742032_0001/__spark_conf__.zip
24/09/14 19:40:40 INFO SecurityManager: Changing view acls to: hadoop
24/09/14 19:40:40 INFO SecurityManager: Changing modify acls to: hadoop
24/09/14 19:40:40 INFO SecurityManager: Changing view acls groups to:
24/09/14 19:40:40 INFO SecurityManager: Changing modify acls groups to:
24/09/14 19:40:40 INFO SecurityManager: SecurityManager: authentication disabled; ui acls disabled; users with view permissions: hadoop; groups with view permi
permissions: EMPTY
24/09/14 19:40:40 INFO Client: Submitting application application_1726342742032_0001 to ResourceManager
24/09/14 19:40:41 INFO YarnClientImpl: Submitted application application_1726342742032_0001
24/09/14 19:40:42 INFO Client: Application report for application_1726342742032_0001 (state: ACCEPTED)
24/09/14 19:40:42 INFO Client:
         client token: N/A
         diagnostics: AM container is launched, waiting for AM container to Register with RM
         ApplicationMaster host: N/A
         ApplicationMaster RPC port: -1
```

Fig. 21. Logs from the EMR Spark job demonstrate we are running Hadoop for certain tasks

To summarize, we have just demonstrated that a multi-step workflow can first be developed locally on a subset of data, and then deployed to Cloud in order to scale to a bigger data. This can be used for production systems and for daily use and automation.

REFERENCES

Abdelhameed, M. M., & Tolbah, F. A. (2002). A recurrent neural network-based sequential controller for manufacturing automated systems. Mechatronics, 12(4), 617–633. doi:10.1016/s0957-4158(01)00002-2

Alexopoulos, A. *et al.* (2022). Big Data Analytics in the Manufacturing Sector: Guidelines and Lessons Learned Through the Centro Ricerche FIAT (CRF) Case. In: Curry, E., Auer, S., Berre, A.J., Metzger, A., Perez, M.S., Zillner, S. (eds) Technologies and Applications for Big Data Value . Springer, Cham. https://doi.org/10.1007/978-3-030-78307-5_15

Bendaouia, A., El Hassan Abdelwahed, Qassimi, S., Abdelmalek Boussetta, Intissar Benzakour, Amar, O. and Oussama Hasidi (2024). Artificial intelligence for enhanced flotation monitoring in the mining industry: A ConvLSTM-based approach. Computers & chemical engineering, 180, pp.108476–108476. doi:https://doi.org/10.1016/j.compchemeng.2023.108476

Chang, V. I. C., & Lin, W. (2019). How Big Data Transforms Manufacturing Industry. International Journal of Strategic Engineering, 2(1), 39–51. doi:10.4018/ijose.2019010104

docs.aws.amazon.com. (n.d.). Getting Started: Analyzing Big Data with Amazon EMR - Amazon EMR. [online] Available at: https://docs.aws.amazon.com/emr/latest/ManagementGuide/emr-gs.html.

Elhoone, H., Zhang, T., Anwar, M., & Desai, S. (2019). Cyber-based design for additive manufacturing using artificial neural networks for Industry 4.0. International Journal of Production Research, 1–21. doi:10.1080/00207543.2019.1671627

Franco-Sepúlveda, G., Del Rio-Cuervo, J. C., & Pachón-Hernández, M. A. (2019). State of the art about metaheuristics and artificial neural networks applied to open pit mining. Resources Policy, 60, 125–133. doi:10.1016/j.resourpol.2018.12.013

Gupta A.,Thakur H. K., Shrivastava R., Kumar P. and Nag S., "A Big Data Analysis Framework Using Apache Spark and Deep Learning," 2017 IEEE International Conference on Data Mining Workshops (ICDMW), New Orleans, LA, USA, 2017, pp. 9-16, doi: 10.1109/ICDMW.2017.9

Holden Karau, Konwinski, A., Wendell, P., & Matei Zaharia. (2015). Learning Spark. O'reilly.

Huang, S. H., & Hong-Chao Zhang. (1994). Artificial neural networks in manufacturing: concepts, applications, and perspectives. IEEE Transactions on Components, Packaging, and Manufacturing Technology: Part A, 17(2), 212–228. https://doi.org/10.1109/95.296402

Lade, P., Ghosh, R., & Srinivasan, S. (2017). Manufacturing Analytics and Industrial Internet of Things. IEEE Intelligent Systems, 32(3), 74–79. doi:10.1109/mis.2017.49

Li, C., Chen, Y., & Shang, Y. (2021). A review of industrial big data for decision making in intelligent manufacturing. Engineering Science and Technology, an International Journal. doi:10.1016/j.jestch.2021.06.001

Lin, H.-K., Harding, J. A., & Chen, C.-I. (2016). A Hyperconnected Manufacturing Collaboration System Using the Semantic Web and Hadoop Ecosystem System. Procedia CIRP, 52, 18–23. doi:10.1016/j.procir.2016.07.075

Ooi, K.-B., Lee, V.-H., Tan, G. W.-H., Hew, T.-S., & Hew, J.-J. (2018). Cloud computing in manufacturing: The next industrial revolution in Malaysia? Expert Systems with Applications, 93, 376–394. doi:10.1016/j.eswa.2017.10.009

Pumperla, M. (2015). elephas. [online] Max Pumperla. Available at: https://maxpumperla.com/projects/elephas/ [Accessed 19 Sep. 2024].

Pumperla et al., (2022). Elephas: Distributed Deep Learning with Keras & Spark. Journal of Open Source Software, 7(80), 4073, https://doi.org/10.21105/joss.04073

Qi, X., Chen, G., Li, Y., Cheng, X., & Li, C. (2019). Applying Neural-Network-Based Machine Learning to Additive Manufacturing: Current Applications, Challenges, and Future Perspectives. Engineering. doi:10.1016/j.eng.2019.04.012

Rahul, K., Banyal, R.K. & Arora, N. A systematic review on big data applications and scope for industrial processing and healthcare sectors. *J Big Data* 10, 133 (2023). https://doi.org/10.1186/s40537-023-00808-2

Sazu, M., & Jahan, S. (2022). Impact of big data analytics on distributed manufacturing: Does big data help? Journal of Process Management and New Technologies, 10(1-2), 70–81. https://doi.org/10.5937/jouproman2201070s

spark.apache.org. (n.d.). MultilayerPerceptronClassifier — PySpark 3.3.1 documentation. [online] Available at: https://spark.apache.org/docs/latest/api/python/reference/api/pyspark.ml.classification.MultilayerPerceptronClassifier.html.

Su, N.-J., Huang, S.-F. and Su, C.-J. (2024). Elevating Smart Manufacturing with a Unified Predictive Maintenance Platform: The Synergy Between Data Warehousing, Apache Spark, and Machine Learning. Sensors, [online] 24(13), p.4237. doi:https://doi.org/10.3390/s24134237.

Uğuroğlu E. , "Near-Real Time Quality Prediction in a Plastic Injection Molding Process Using Apache Spark," 2021 International Symposium on Computer Science and Intelligent Controls (ISCSIC), Rome, Italy, 2021, pp. 284-290, doi: 10.1109/ISCSIC54682.2021.00059.

ur Rehman, M. H., Yaqoob, I., Salah, K., Imran, M., Jayaraman, P. P., & Perera, C. (2019). The role of big data analytics in industrial Internet of Things. Future Generation Computer Systems, 99, 247–259. https://doi.org/10.1016/j.future.2019.04.020

Wang, J., Xu, C., Zhang, J., & Zhong, R. (2021). Big data analytics for intelligent manufacturing systems: A review. Journal of Manufacturing Systems. doi:10.1016/j.jmsy.2021.03.005

Xu, X. (2012). From cloud computing to cloud manufacturing. Robotics and Computer-Integrated Manufacturing, 28(1), 75–86. doi:10.1016/j.rcim.2011.07.002

Xu, K., Li, Y., Liu, C. et al. Advanced Data Collection and Analysis in Data-Driven Manufacturing Process. Chin. J. Mech. Eng. 33, 43 (2020). https://doi.org/10.1186/s10033-020-00459-x