# Introduction

Encoding Images of handwritten digits using MapReduce in Hadoop

First get the dataset

Load into Hadoop, perform MapReduce on it, then encode the data in Python and do the neural network and compare results of Mapreduce vs Original Dataset in HDFS

# Literature review

*MapReduce-based Deep Learning With Handwritten Digit Recognition Case Study (cited by 13)*

https://static.aminer.org/pdf/fa/bigdata2016/BigD551.pdf

HIPI: A Hadoop Image Processing Interface for Image-based MapReduce Tasks (Cited by 151)

<https://jasonlawrence.info/assets/pdf/hipi.pdf>

*Accelerating Support Vector Machine Learning with GPU-based MapReduce (I don’t think this is very good – read later):*

https://sci-hub.se/https://ieeexplore.ieee.org/abstract/document/7379293

# The Proposed Approach

# The Experiment

# Results and discussion

# Conclusion and Future Works

# References

Learning outcomes that they’re testing for:

Big data:

* Critically assess the data storage and management requirements of a given data project from a modern perspective and evaluate limitations of legacy approaches to Big Data
  + **Talk about the data I have, and how I’m going to store and process it and why old storage techniques are not useful.**
* Assess the design concepts and architectural patterns of distributed Big Data systems and analyse the components that form their technology stack
  + **Check a few distributed Big Data Systems (Hadoop/ NoSQL Databases/Stream processing/Data Warehousing/Container Orchestration**
* Critically evaluate and select a Big data environment suitable for retrieving and processing a given Big Data set, perform data management and select appropriate analytic algorithms for the required scale and speed.
  + **This is the actual doing of the experiment. Talk about why Hadoop was chosen for this particular project**

Advanced Data Analytics:

* Debate the theory and application of different types of neural networks
  + **Talk about different Neural network types and where they’re useful**
* Analyse a set of requirements to determine the type of Neural Network for a particular problem set. Document and justify the choices made to stakeholders and peers through insight gained from the process.
  + **Talk about why I chose the Neural Network I chose**

Certainly! Here's an example of a data analytics project that incorporates Hadoop and image classification using Convolutional Neural Networks (CNNs):

\*\*Project Title: Big Data Image Classification with Distributed CNNs using Hadoop\*\*

\*\*Project Overview:\*\*

In this project, you will develop an image classification system using CNNs and leverage Hadoop for distributed data processing. The objective is to classify a massive dataset of images into predefined categories while efficiently handling the scale of big data.

\*\*Project Steps:\*\*

1. \*\*Data Collection and Storage:\*\*

- Gather a large-scale image dataset, potentially containing millions of images. This dataset should be stored on Hadoop Distributed File System (HDFS) for efficient distributed processing.

2. \*\*Data Preprocessing:\*\*

- Preprocess the image data by resizing, normalizing, and augmenting as necessary.

- Convert the images into a format suitable for Hadoop processing, like Hadoop SequenceFile or Parquet.

3. \*\*Hadoop Distributed Processing:\*\*

- Utilize Hadoop MapReduce or Spark to distribute the preprocessing and feature extraction tasks across a Hadoop cluster.

- You can use deep learning libraries like TensorFlow or PyTorch on each node to apply pre-trained CNN models for feature extraction.

4. \*\*Neural Network Model for Image Classification:\*\*

- Design and train a CNN model for image classification using a deep learning framework.

- Transfer learning can be particularly useful in this scenario, where you fine-tune a pre-trained model on the extracted features from Hadoop.

5. \*\*Model Training and Validation:\*\*

- Train your CNN model on a subset of the data to validate its performance.

- Utilize cross-validation techniques to ensure robustness.

6. \*\*Distributed Model Training with Hadoop:\*\*

- Distribute the model training across your Hadoop cluster. This can be achieved using Hadoop's distributed computing capabilities.

7. \*\*Model Evaluation:\*\*

- Evaluate the trained model's performance using metrics such as accuracy, precision, recall, and F1-score on a separate validation dataset.

8. \*\*Inference on Big Data:\*\*

- Deploy the trained model to perform inference on the large-scale image dataset stored in HDFS.

- Utilize Hadoop to parallelize and distribute inference tasks across the cluster, allowing for efficient processing.

9. \*\*Scaling for Real-Time Inference:\*\*

- If real-time inference is required, integrate the model with a stream processing framework (e.g., Apache Kafka and Apache Flink) for handling incoming image data in real-time.

10. \*\*Monitoring and Maintenance:\*\*

- Implement monitoring and maintenance routines to ensure the system's reliability and performance in a production environment.

\*\*Skills and Technologies Involved:\*\*

- Convolutional Neural Networks (CNNs)

- Deep learning frameworks (e.g., TensorFlow, PyTorch)

- Hadoop ecosystem (HDFS, MapReduce, Spark)

- Data preprocessing and augmentation techniques

- Model evaluation and fine-tuning

- Distributed computing and cluster management

- Real-time stream processing (if applicable)

This project demonstrates the integration of Hadoop's distributed processing capabilities with CNNs for scalable image classification on big data. It addresses the challenges of processing and analyzing large-scale image datasets efficiently.