Mitigating the Metadata Mess: Autonomous Metadata Extraction Pipelines for Large-Scale Data Repositories

Matthew Chen
Univeristy of Illinois at Urbana-Champaign

Erica Hsu
Carnegie Mellon University

Tyler J. Skluzacek
& Kyle Chard (advisors)
University of Chicago

ABSTRACT

Building on a distributed metadata extraction service, Xtract, we propose a scheduler designed to maximize the amount of metadata extracted from large scientific repositories subject to finite compute budgets. We accomplish this by leveraging machine learning models to predict the likelihood that each metadata extractor can retrieve nonempty metadata from each file. We then feed these probabilities along with other file attributes into a scheduler that maximizes metadata yield over time. We demonstrate the viability of the scheduler on a real-world data repository and show improved metadata extraction performance.

1 INTRODUCTION

Many scientific repositories are rendered useless due to their enormous size (exceeding petabytes of data across billions of files) and lack of descriptive metadata to aid discovery, understanding, and use. Xtract [1] automates the creation of metadata by crawling repositories and applying a series of metadata extractor functions to files. In this poster, we investigate improving the metadata extraction process to maximize the amount of *valuable* metadata extracted and minimize extraction time. We train a classifier to predict the likelihood that a given extractor will yield metadata from a particular file. We use the classifier to prioritize potential file/extractor pairs. We evaluate our approach on a climate dataset and show that it reduces extraction time while maintaining metadata readability, completeness, and information entropy.

2 BACKGROUND

We use the Carbon Dioxide Information Analysis Center (CDIAC) data store repository [2] to motivate and evaluate this work. CDIAC contains over 500,000 files of diverse types, structures, and sizes, distributed among twelve directories (called "pubs"). Also, CDIAC contains more than 150 different file extensions (see Figure 1). Extensions do not necessarily map to extractors, thus motivating the need to identify a file type based on file content.

We evaluate our approaches with a subset of Xtract's extractors: image, keyword, tabular, NetCDF, and JSON/XML.

Conference'17, July 2017, Washington, DC, USA

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

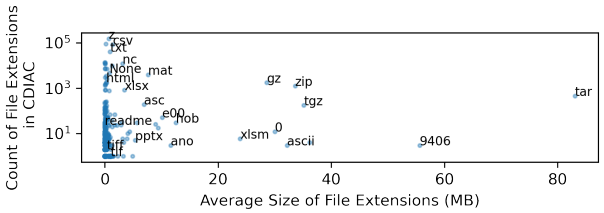

Figure 1: Count of file extensions in CDIAC by average file size

3 METHODOLOGY

We describe how we sample file types and prioritize extractors.

3.1 File Type Identification

We leverage file type identification (FTI) methods to automatically determine a file's applicable extractors. FTI relies on attainable features from files (e.g., byte-strings), and in this work we utilize the first 512 bytes from each file's header. Figure 2 and 3 show a PCA/T-SNE representation of the byte features for all files in CDIAC's pub8. We observe that our six file types studied have visually distinguishable byte features.

We developed a new FTI model that identifies the best extractors to apply to files by leveraging the probabilistic forms of multiple statistical learning models. Specifically, we train and test SVM, Logistic Regression, and Random Forest classifiers. Outputs from this model constitute a *probability vector*—the distribution of probabilities that a file is of each type.

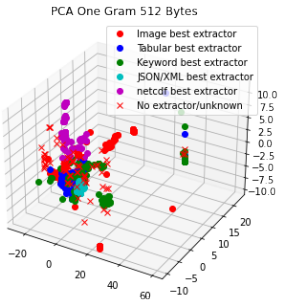

Figure 2: PCA Visualization

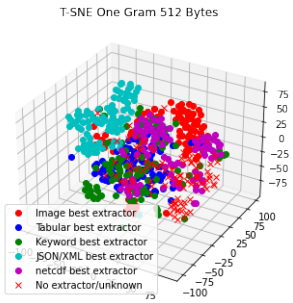

Figure 3: TSNE Visualization

3.2 Scheduler

We developed a novel scheduler for optimizing extraction under constrained computational budgets that uses the probability vectors of each file. The scheduler orders the queue of files to be processed such that during the extraction phase, the amount of metadata generated is maximized with minimal costs. To determine how to order these files, we propose an objective function:

$$\alpha(F,E) = log\frac{S_e(F,E) \times P(F,E) + 1}{T_e(F,E) + 1} \quad (1)$$

Where F represents the input file (with file system metadata such as file size known), E the extractor, $S_e(F,E)$ a regression that predicts size of the metadata generated by an extractor, $T_e(F,E)$ a regression that predicts time of extraction, and P the probability the file is best extracted using extractor E. Figure 4 shows the scheduler workflow.

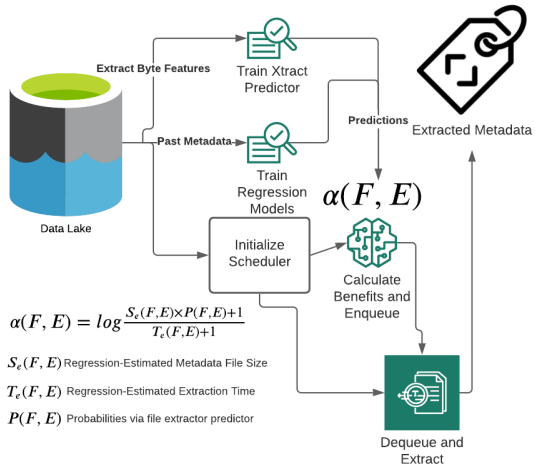

Figure 4: Scheduler Workflow

3.3 Measuring Metadata Quality

To evaluate the utility of our model, we define three metadata quality metrics:

- **Completeness** measures the amount of metadata that is obtained. We establish the maximum number of metadata fields as the union of all the fields seen in our sample dataset.
- **Readability** measures how well humans could comprehend the metadata. We apply the Flesch index to each field; higher scores represent more concise sentences.
- **Entropy** measures the *usefulness* of metadata. Metadata with a higher score are less repetitive. We use TF-IDF to score each word in the metadata, and compute the average score to obtain a single score.

4 EVALUATION

We evaluate our scheduler by experiments on CDIAC pub8 data.

4.1 Probabilistic FTI Model

We compare SVM, logistic regression, and random forests classifiers with the 512-byte file headers as features. We generated training labels by running all available extractors and associating the file's best extractor with the one that generated the most metadata (in bytes). Our results (Table 1) show that the random forests classifier outperforms the other methods in precision, recall, and accuracy.

Table 1: Classifier performance for predicting file types

Metric	SVM	Logistic Regression	Random Forest
Precision	92.67	90.41	95.13
Recall	93.58	91.13	96.06
Accuracy	93.27	90.66	95.67

4.2 Scheduler

We evaluate the scheduler by measuring the metadata quality at varying processing thresholds. More specifically, each threshold represents a percentage of file/extractor pairs processed based on the priority determined by the scheduler. From our results (Figures 5–7), we conclude that the scheduler successfully maximizes the quality of metadata extracted because the marginal cumulative quality of metadata decreases as the scheduler goes deeper in the queue. Additionally, the scheduler attempts to apply one extractor to each file before applying subsequent ones—a useful strategy given limited computational time.

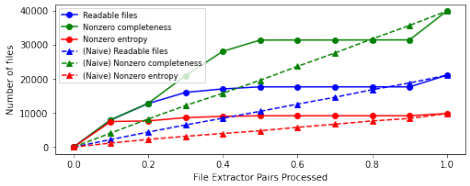

Figure 5: Number of files with quality metadata

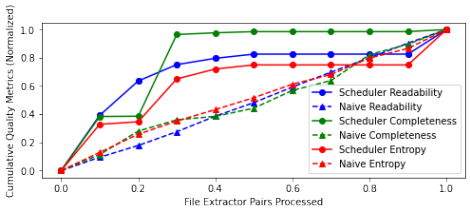

Figure 6: Number of files with quality metadata

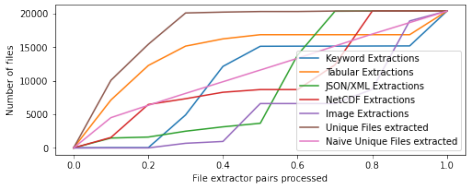

Figure 7: Number of files with quality metadata

5 CONCLUSION

Our results show that our scheduler, leveraging probabilistic FTI, can automatically create extraction pipelines that increase metadata quality while minimizing resource consumption. Our approach prioritizes extractors that are likely to yield valuable metadata. All code is available online: https://github.com/xtracthub/xtract-sampler/tree/Xtract-Scheduler.

REFERENCES

[1] Tyler Skluzacek et al. 2020. A Serverless Framework for Distributed Bulk Metadata Extraction. In *HPDC*. 7–18.

[2] U.S. Dept. of Energy. 2021. Carbon Dioxide Information Analysis Center (CDIAC). (2021). https://cdiac.ess-dive.lbl.gov/