Applying Image Feature Extraction to Cluttered Scientific Repositories

Emily Herron
Illinois Institute of Technology
Mercer University
Emily.Joyce.Herron@live.mercer.edu

Tyler J. Skluzacek, Ian Foster, and Kyle Chard (Advisors)
Computation Institute,
University of Chicago and Argonne National Laboratory
{foster, chard, skluzacek}@uchicago.edu

***Abstract*—While scientific data repositories aim to disseminate scientific knowledge to research communities, in many situations they become a convenient "dumping ground" for disorganized, poorly described, and error-ridden data. In this poster we investigate methods for automatically making sense of data in such repositories by augmenting the Skluma pipleline—a system that automates the extraction of metadata from structured and semi-structured scientific formats—with the ability to process scientific image formats. We present a pipeline that applies a collection of image metadata extraction techniques that leverage file system metadata, header information, color content statistics, extracted text, feature-based clusters, and predicted tags using a supervised learning model. Our goal is to collect a large number of metadata that may then be used to organize, understand, and use data stored in the repository.**

I. Introduction

There are many examples of well-organized, and indeed valuable, scientific data repositories, however there are also an increasing number of disorganized and poorly described data repositories that are difficult for researchers to use. Vast and unkempt reservoirs of scientific data accumulated over the course of decades may be concealed by obscurely-named directory and file names, buried among irrelevant files, and hidden due to a lack of descriptive metadata.

To address this problem, we developed Skluma: an automated pipeline for making sense of large collections of scientific data [1]. Skluma crawls a repository, extracts metadata from files, establishes relationships between files and ultimately assembles a probabilistic "ball" of metadata. Until now, Skluma has focused primarily on semi-structured text-based files. However, there are many other data formats common to science, not least of which are image files. In this poster we present a pipeline for extracting various types of metadata from images. We specifically focus on extracting image metadata (e.g., resolution, dimensions, etc), image classes (e.g., map or plot), and text embedded within the figure.

We evaluate our pipeline on the US Department of Energy's Carbon Dioxide information and Analysis Center's repository (CDIAC) [2]. CDIAC contains tens of thousands of image files amongst its more than 500 thousand scientific files that cumulatively exceed 500 GB.

II. Methodology

The high level pipleine is shown in Figure 1. For each image file, our pipeline employs a series of modules to collect

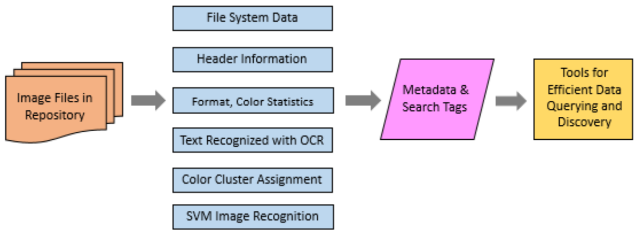

Fig. 1. Skluma image metadata extraction pipeline.

metadata from file-system metadata, image header information, text strings embedded within images, image format and color statistics. We also apply methods to cluster images based on extracted features (e.g., color) and to tag images based on content using a supervised learning model. The result of our pipeline is a collection of metadata (in JSON format) for each image that can be used to organize and discover data.

File System Metadata: The module first crawls the repository to locate image files (with extensions ".jpg," ".png," ".gif," and ".bmp"). Concurrently, we collect file system metadata, including filenames, file paths, file extensions, and file sizes.

We then tokenize extracted strings (e.g., file names and paths) into discrete tags by separating on common characters (e.g., underscores, spaces, camel case, and digits). These file name tags may provide content-related keywords, dates, or numbers that may be useful in searches.

Image Metadata: The second module aims to extract structured metadata from with different image formats. For this purpose we use the Python Image Library (PIL). Header contents vary based on image format but commonly include details such as format, dimensions, encoding, creation data, and software. Moreover, our module includes functionality for parsing file and parent-folder names into tags by splitting on common word delimiters (e.g., underscores) and capitalization schemes (e.g., CamelCase) in order to provide key words usable in searches.

Image Feature Clusters: The next module aims to cluster images by their features—an important step in determining image similarity. Image similarity is not only useful for similarity searches but also for linking images and associating shared metadata. As a base model for clustering we utilize features derived from the previous modules (e.g., mean, median, mode, standard deviation, and extrema of image color values) using PIL histogram, getcolors, and getextrema functions.

Our module first resizes all image samples in RGB or RGBA mode and divides them into a 4 by 4 grid. Mean RGB values calculated from each of the grid's 16 sections are converted to floating point values and appended to a feature vector. The set of resulting feature vectors are assigned to clusters using scikit-learn's K-Means clustering function. The K-Means clustering algorithm aims to divide a group of sample vectors into a specified number, k, groups of equal variance, where each cluster is described by a centroid or mean of each the sample in the cluster. To avoid the "curse of dimensionality," we reduce the dimensions of the feature vectors by projection onto a 2 dimensional space using principal component analysis (PCA).

Supervised Classification: CDIAC contains several clear classes of images such as maps, graphs, and figures. To automate the identification of these classes we trained a support vector machine model to classify and predict the content of the sample images. We first labeled a collection of 300 sample images with one of five class assignments: map, map with depth chart, map with histogram, map with plot or other.

Our module first resizes images to standard dimensions and converts to grayscale, numpy arrays. It then applies PCA to reduce the dimensions of these arrays. Using our labeled images we then trained a support vector machine classification model using scikit-learn's c-support classification model (SVC) function.

Text Extraction: The CDIAC dataset contains images with a large amount of text including titles, locations on maps, axes labels on graphs, and descriptive text on other images. Our image text extraction module relies on Python-tesseract—an optical character recognition tool that wraps Google Tesseract-OCR engine [3]. To improve accuracy, we first use the Python Image Library to convert RGB and RGBA images to grayscale. We then pass these images to python-tesseract to extract text as unicode strings.

III. Evaluation

We evaluate our pipeline by testing it on images located and downloaded via file transfer protocol from the entire CDIAC repository. Given space constraints we present only evaluation of our clustering and tagging modules.

A. Clustering

To determine an optimal value for k, we applied silhouette analysis to the PCA-reduced clusters. Silhouette analysis measures the average distance between neighboring clusters. Using the optimal value of $k = 4$, Figure **??** illustrates the resulting clusters and associated silhouette scores. Figure **??** shows examples of images in each of the generated clusters. As can be seen, the clustering module performs well with clear groupings based not only color but also on content.

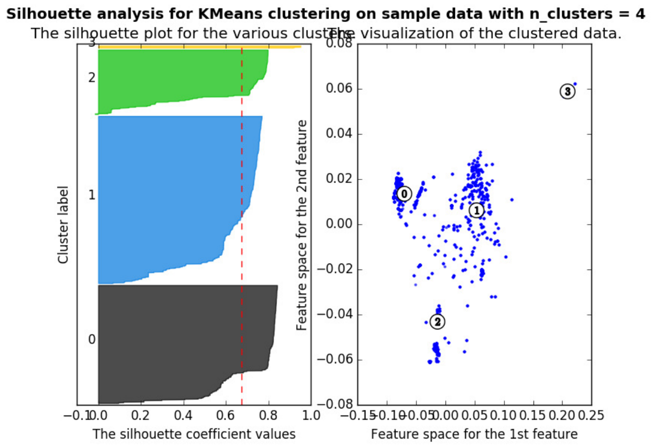

Fig. 2. Silhouette analysis for K-Means clusters where k = 4.

Class Label	Precision	Recall	f1 Score	Support
map	0.94	0.95	0.95	88
map&depth_chart	1.00	1.00	1.00	99
map&plot	1.00	0.50	0.67	6
map&histogram	1.00	0.43	0.60	7
other	0.00	0.00	0.00	1
avg / total	0.97	0.94	0.95	201

TABLE I
PRECISION, RECALL, F-MEASURE, AND SUPPORT SCORES FOR 2:1 RATIO SPLIT OF TEST AND TRAINING SET.

B. Classification

To evaluate our SVM classifier we split our manually tagged images into a training and test set. Table I shows the precision and recall for each image class. The results show high precision for all classes, and high recall for all but the map and plot class. These results highlight the ability of our module to accurately classify images based on content. Similar results were obtained with different training/test splits.

IV. Summary

We have presented a unique image feature extraction module for Skluma containing a suite of methods for collecting a variety of metadata from each image. In future work, we plan to use convolutional neural networks (CNNs) in order to recognize physical objects, match maps to their spatial coordinates, and interpret scientific charts in image files.

Class Label	Number Predicted in CDIAC
map	2877
map&depth_chart	22
map&plot	395
map&histogram	144
other	200
total	3638

TABLE II
NUMBER OF CDIAC IMAGES OF EACH CLASS PREDICTED BY THE TRAINED SVM MODEL.

References

[1] P. Beckman, T. J. Skluzacek, K. Chard, and I. Foster, “Skluma: A statistical learning pipeline for taming unkempt data repositories,” in *Proceedings of the 29th International Conference on Scientific and Statistical Database Management*, SSDBM ’17, (New York, NY, USA), pp. 41:1–41:4, ACM, 2017.

[2] U.S. Department of Energy, “Carbon dioxide information and analysis center,” 2017.

[3] S. Hoffstaetter, J. Bochi, M. Lee, and L. Kistner, “pytesseract 0.1.7.,” 2017.