Tackling Email Processing and Access with NLP and Machine Learning

Christopher (Cal) Lee
UNC Chapel Hill School of Information and Library Science

Camille Tyndall Watson
State Archives of North Carolina

Best Practices Exchange, May 19, 2020
Motivation - Selection/Appraisal

• Despite progress on various technologies to support data management and digital preservation, relatively little progress on software support for the core activities of selection and appraisal.

• Selection/appraisal decisions are based on various patterns.

• When patterns can be identified algorithmically, software can assist the process.

• LAMs frequently want to take actions that reflect contextual relationships.

• Timeline representations and visualizations can also provide useful, high-level views of materials.
Motivation - Email

- 48 years of email creation
- Hundreds of billions of messages generated every day
- Most has little long-term retention value, but some absolutely does
- Despite presence of numerous other modalities, email still deeply embedded in activities, serving as massive source of evidence and information
- Often found in collections and acquisitions with other types of materials

http://hci.stanford.edu/~heer/projects/enron/v1/
Review, Appraisal, and Triage of Mail (RATOM)

- Funded by Andrew W. Mellon Foundation (2019-2020)
- Developing and repurposing software (including NLP and machine learning) for selection/appraisal in BitCurator environment with hooks and enhancements to TOMES output
- Support iterative processing - information discovered at various points in the processing workflow can support further selection, redaction or description actions
- Mapping of timestamp, entity, sensitive features and other elements across the tools

Ray Tomlinson
Implemented first email program on ARPANET. Credited with invention of first email system.
Scope of the project

The RATOM project has several core development goals designed to serve the needs of collecting institutions tasked with preparing email collections for public access:

- Creating an integrated Python library and CLI tools to simplify parsing and processing PST, OST, and mbox email formats
- Developing utilities to support entity identification and export reports suitable for conducting automated and human-directed redaction actions at scale
- Developing an interface allowing processing archivists to browse email collections and mark messages as suitable for retention
- Developing utilities to apply machine learning techniques (by training on annotated message collections and/or unsupervised) to recognize candidate materials for retention
Tools (in development)

libratom (core library)

- Python library + API to support parsing and processing PST, OST, and mbox email formats
- Wraps and extends functionality of libpff (https://github.com/libyal/libpff), Python mailbox library, and spaCy NLP processing platform
- Email message content, header, and attachment extraction; entity identification; PST structure reporting
- Engineered to scale with core count. Flat memory use per-core, supports processing collections of arbitrary size
- https://www.github.com/libratom/libratom

Annotation and iterative processing (web interface)

- Direct import from PST via libratom
- Review and annotation of email message and metadata content
- Support for expert classification of individual messages as record, non-record, restricted, redacted
- Classification parameter export / reporting
- Dump selected messages as EML for delivery / serve open records request
- https://github.com/StateArchivesOfNorthCarolina/ratom-deploy (currently in development!)
Many collecting orgs have unprocessed PST and mbox files in digital acquisitions, need better tools for assessment/research

- **libpff**
  - Open source digital forensics software libraries

- **libratom**
  - Scalable open source email processing library

- **sqlite3 databases and derived reports**

- **email, multiprocessing**
  - Core Python libraries

- **spaCy**
  - Open source NLP platform

By combining functionality from open source forensic libraries, an industrial-scale NLP platform, and multiprocessing and email libraries in Python 3 we can process email in common formats quickly and generate feature files that are:

- **VERIFIABLE**
- **REPRODUCIBLE**
- **REUSABLE**

Derived reports are timestamped and tagged with software library versions. Facilitates understanding changes in assessments of materials over time.
Generating research datasets and support for machine learning tasks

Machine learning methods can assist in archival email processing in identifying relevant messages for a given set of criteria. This requires a large volume of data and identification of relevant features. We need a tool that -

- Can reliably and efficiently extract both simple features and those that require a language parse

- Supports easy swap-in of different models for features that require training for identification (for example, entities)

- Scales effectively when working with large (100GB+) collections

- Exports data in formats and documented configurations that support reuse and comparison
Data analysis and comparison across time and corpora versions

**Engineered for simplicity and extensibility**

**libratom** generates datasets in a simple sqlite3 schema that can be queried and manipulated directly or exported as dataframes for machine learning tasks

- Current schema provides a map of 18 unique entity types to their locations (message and file) within a corpus
- Processing times, file hashes, and system configuration are recorded for preservation support and reproducibility
- SQLAlchemy used to cleanly decouple underlying data model from the schema
Dataset example: Enron (EDRM v1.3) corpus

The EDRM v1.3 Enron Corpus is approximately 54GB, containing 758,341 messages in 191 files.

The following times are representative of typical ratom CLI tool runs on a 16-core (Threadripper 2950X) desktop:

PST structure scan: **30 seconds**

Report to sqlite3 (no entity extraction): **12 min 30 sec**

Report + entity extraction (spaCy en_core_web_sm 2.2.5 model): **2 hr 6 min**

In libratom 0.4.0, this run yields a **2.3GB sqlite3** file, containing **18,370,006** entity instances.

Memory usage is bounded for the spaCy configuration and number of processes. For 32 processes, accessible memory is ~1.6GB/process, resident memory is ~500MB/process on average.
With the current CLI, we can load different models (including user trained models) on demand for tasks / languages.
libratom

Available on Github at:
https://github.com/libratom/libratom

Releases on PyPI at:
https://pypi.org/project/libratom/

Win10, macOS, and Linux support. Feature extraction support for 10 languages (at time of writing).

Libratom is still in development, but as we add and test new features we’re making some of them available as Jupyter notebooks that you can try in a web browser without installing any software.

Click the “Launch Binder” badge in the ratom-notebooks repository to get started:
https://github.com/libratom/ratom-notebooks
RATOM Processing Interface

- Builds off of lessons from TOMES grant (NHPRC, 2015-2018)
- TOMES tools created tagged EAXS XML files, which worked for preservation, but were difficult to process and access
- Illustrated need for iterative processing tools to address large amount of information found in email accounts
RATOM Processing Interface

The RATOM project is developing an iterative processing interface to assist archivists in reviewing materials for retention and/or release. This interface facilitates:

- Direct import of email corpora from PSTs with automated entity identification via libratom
- Creation of processing accounts associated with individual email stores / backups
- Interactive review and tagging (including for redaction and restriction) of email messages within these accounts
- Export of selected messages as EML for retention and/or release
RATOM Processing Interface

Accounts (individual users) associated with imports of one or more imported PST files.
RATOM Processing Interface

Top level message review within an account.

Subject, sender, and entity categories identified are shown in this early development view.
RATOM Processing Interface

Individual message view.

Messages are pre-tagged with automatically generated entity categories.

* Note: in this pre-release rendering, the message is unformatted.
RATOM Processing Interface

Selection by classification (e.g. record vs non-record) and date range.
RATOM Processing Interface

Back-end administration can be used to review actions, add or remove accounts, and modify label sets.

Audit histories for individual messages are retained to ensure a clear record of processing actions.
Project info, news, and blog posts:
https://ratom.web.unc.edu/

Core library:
https://github.com/libratom/libratom

Sample Jupyter notebooks:
https://github.com/libratom/ratom-notebooks

Processing interface deployment tools (in development):
https://github.com/StateArchivesOfNorthCarolina/ratom-deploy

@RATOM_Project