# Building a DeepDive Application Infrastructure

Ian Ross, University of Wisconsin-Madison Center for High Through Computing

iross@cs.wisc.edu




# Key Questions

- Can a machine reading system construct a literature-based data compilation suitable for science?

- How can we build a text data mining (TDM) infrastructure that drives this data compilation (and other literature-heavy applications)?

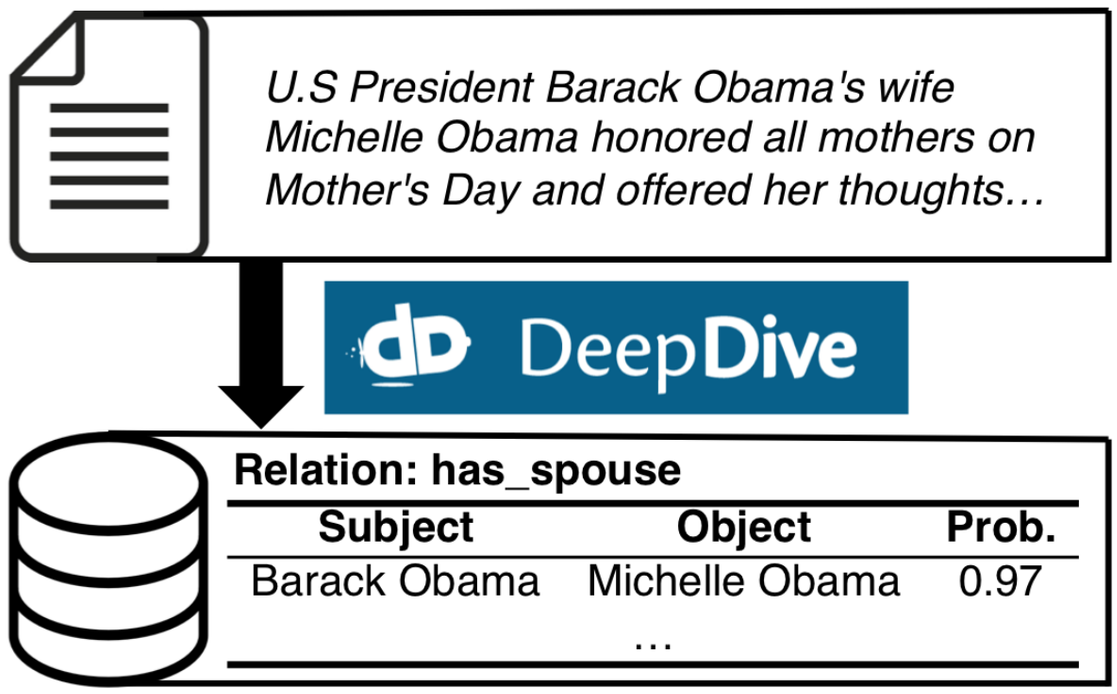
# Key Questions (SPOILERS)

- Can a machine reading system construct a literature-based data compilation suitable for science?
  - Yes — DeepDive
- How can we build a text data mining (TDM) infrastructure that drives this data compilation (and other literature-heavy applications)?
  - High throughput computing is the heart

# Table of Contents

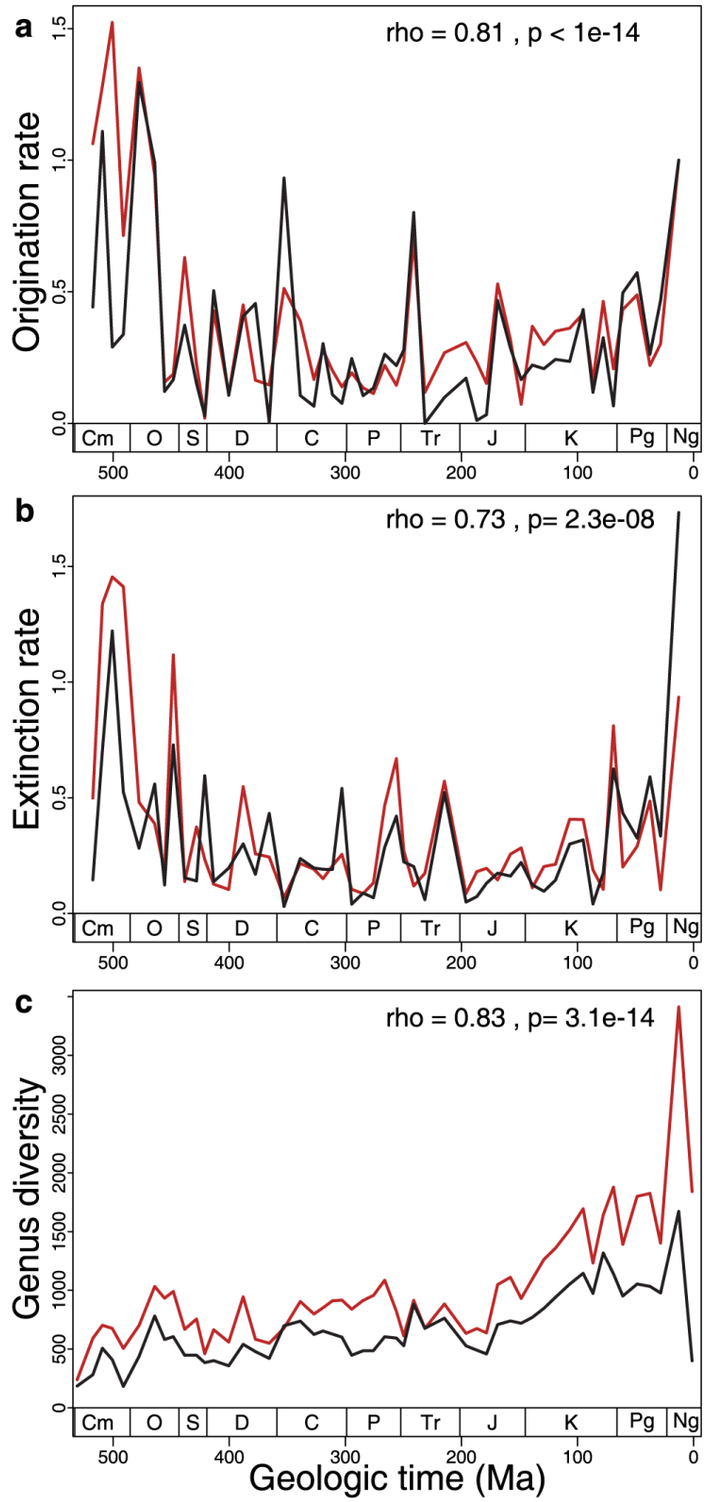
# Introduction — DeepDive

- What is DeepDive?
  - System that leverages machine learning techniques on domain-specific knowledge to drive data analysis
  - Learning at a “distance” — experts in a field can build simple rules and let DeepDive do the hard work (i.e. no tedious training)
- http://deepdive.stanford.edu/

# Introduction — PaleoDeepDive

- DeepDive technology applied to paleontology
  - Extracts relations between biological taxa, geological formations, geographic locations, and geological time intervals
- **“PaleoDeepDive performs comparably to humans in several complex data extraction and inference tasks** and generates congruent synthetic results that describe the geological history of taxonomic diversity and genus-level rates of origination and extinction.”
  - "A Machine Reading System for Assembling Synthetic Paleontological Database" (Peters, Zhang, Livny, Re)
- http://deepdive.stanford.edu/doc/paleo.htm



# Now what?

- PaleoDeepDive shows that machine reading can build a database that not only compares well to human-built ones, but it also **systematically improves as information is added.**

**So let's build a TDM pipeline to streamline the process of building a DeepDive application!**

# The Infrastructure Challenges

- Sources of documents
  - Legal and responsible access to scientific literature
- Organization
    - Bibliographical information for each document
    - What versions of what tools have we used to process each document? Tracked via a *tag*
- Computing
  - Need resources, automation, and flexibility
    - **This framework should be useful for non-DeepDive applications as well!**
- Monitoring
- Security
- Discovery — Need easy ways to extract documents of interest (currently using ElasticSearch to search the extracted text)

# Challenges — Organization


```
> db.articles.findOne()
{
        "_id" : ObjectId("54db884ee138238a47f95eb0"),
        "sha1" : "1c918973dcb0a5d2a39ed9afedf955b0b1fe4f7d",
        "filepath" : "/home/iross/DeepDiveEnv/DeepDive/downloads/Journal of Hydrology/0022169465901101.pdf",
        "title" : "Choice of adjustment to floods Gilbert F. White: University of Chicago, Chicago Ill., 1964, 150 p. (Research pape
r no. 93)",
        "vol" : "3-4",
        "pubname" : "Journal of Hydrology",
        "URL" : "http://api.elsevier.com/content/article/pii/0022169465901101",
        "startingPage" : 344,
        "authKeywords" : "",
        "time" : "2015-02-11 10:49:49.625599",
        "authors" : "",
        "cuneiform_processing" : {
                "elsevier_002" : {
                        "harvested" : true,
                        "contents" : [ ],
                        "filename" : [ ]
                }
        },
        "ocr_processing" : {
                "elsevier_002" : {
                        "harvested" : true,
                        "contents" : [ ],
                        "filename" : [
                                "/home/iaross/elsevier_002/ChtcRun/submit_12Feb_out/job000005/page-1.hocr.html"
                        ]
                }
        },
        "nlp_processing" : {
                "elsevier_002" : {
                        "harvested" : true,
                        "contents" : [ ],
                        "filename" : [
                                "/home/iaross/elsevier_002/ChtcRun/submit_12Feb_NLP_out_NLP/job000004/input.text"
                        ]
                }
        }
}
```


Store information in mongodb

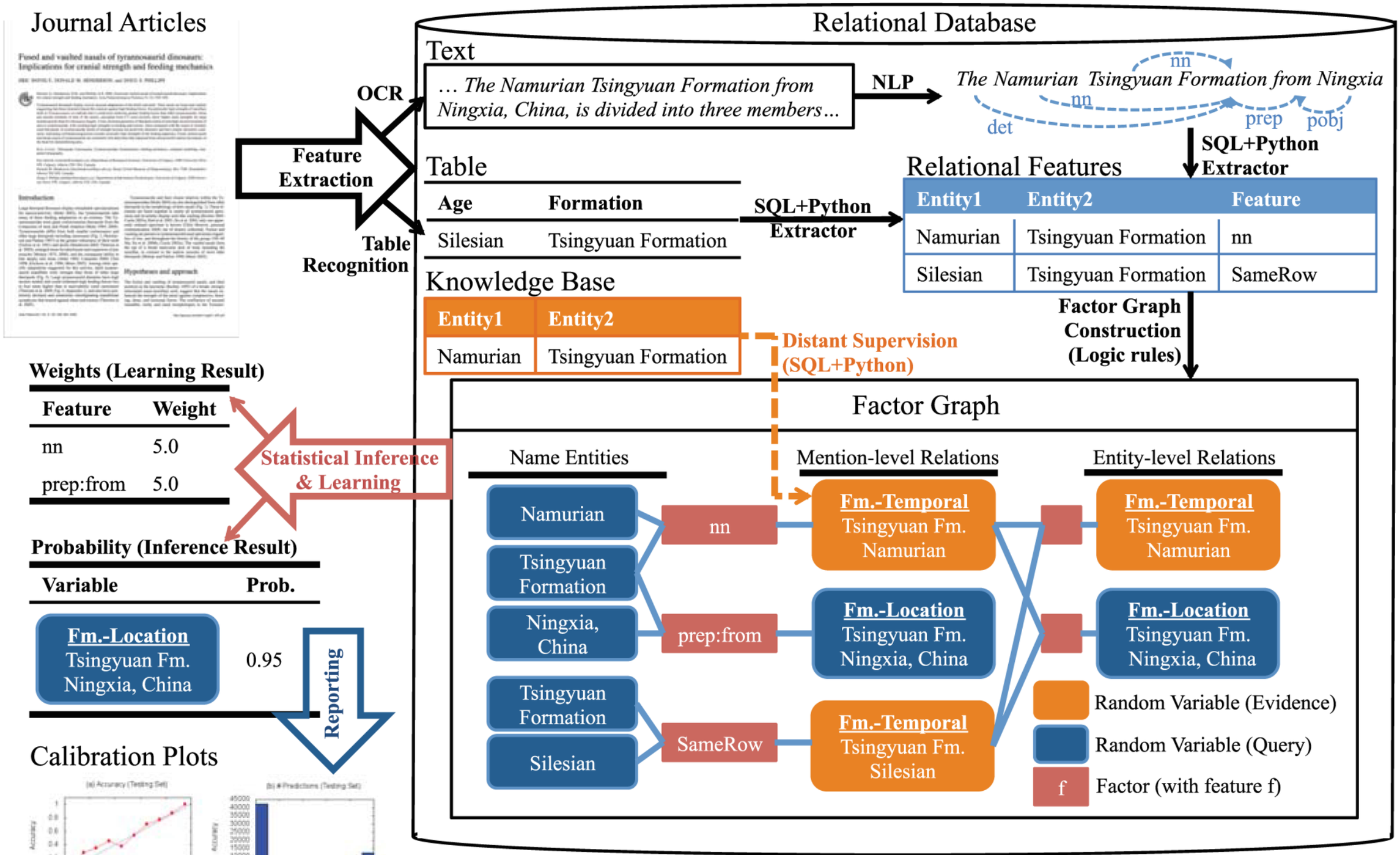
Each articles has the bibliographical information, file-level info (filepath, SHA1 sum, time fetched), and processing information

Subdocuments tell whether the article has been processed for a certain *type+tag* combination.

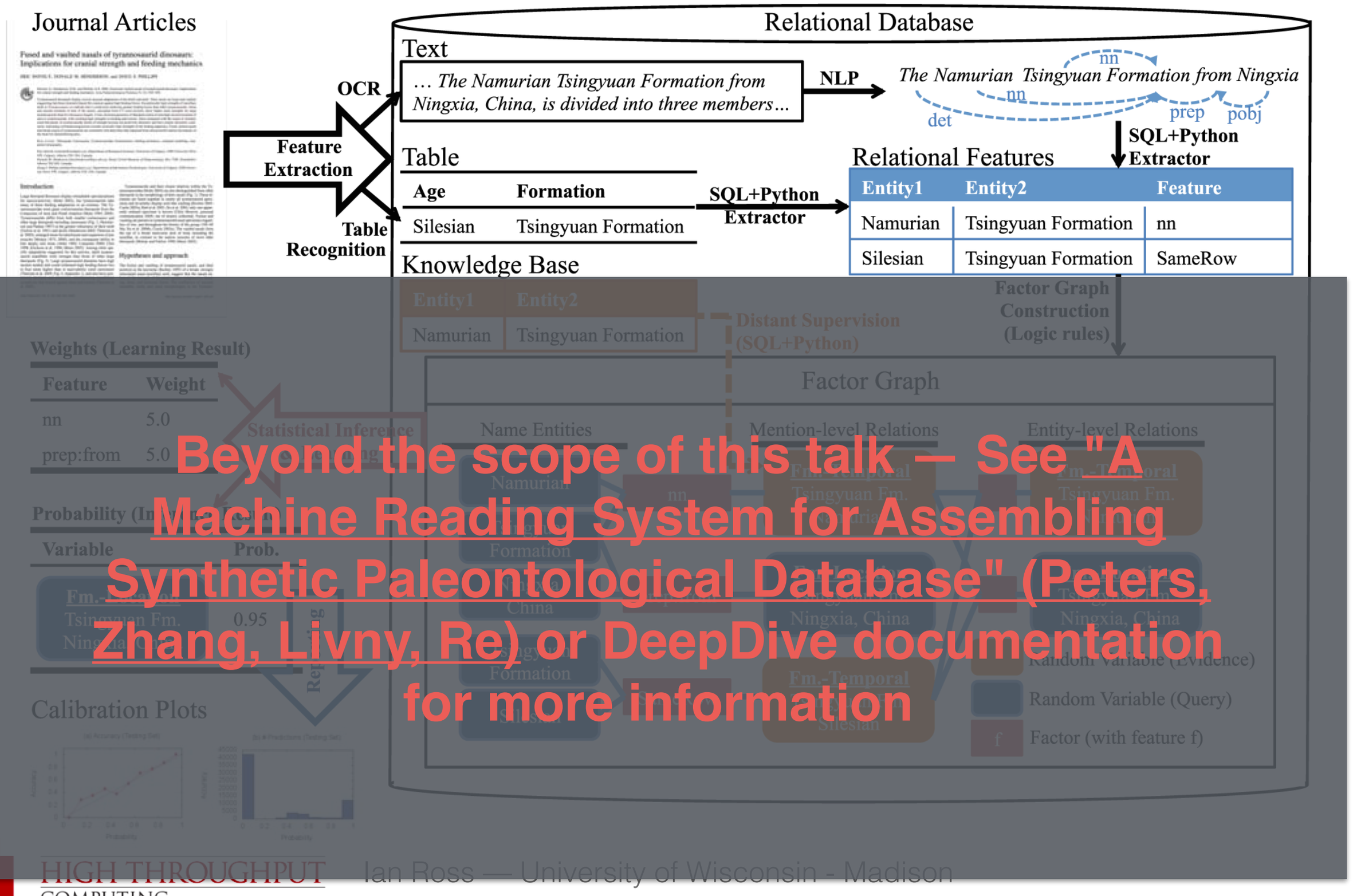
This article has had the Cuneiform, Tesseract, and NLP processing done, though the cuneiform failed

New types of processing can easily go within a new subdocument, and new tags can be added to each

# The Infrastructure Challenge — Computing

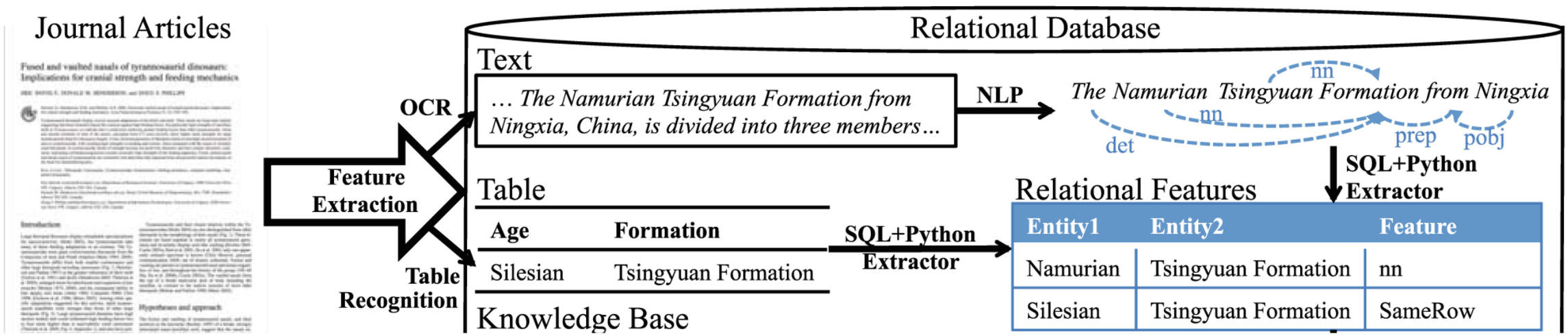


# The Infrastructure Challenge — Computing




**Beyond the scope of this talk — See "A Machine Reading System for Assembling Synthetic Paleontological Database" (Peters, Zhang, Livny, Re) or DeepDive documentation for more information**

# The Infrastructure Challenge — Computing




- At least 10,000 articles a week
- Fairly small/short analysis jobs (5-10 minutes on average for OCR jobs, slightly longer for NLP)
- **High throughput computing is exactly what we need!**
- Use HTCondor and the UW CHTC resources for all this processing work.

# The Infrastructure Challenge — Processing

- Specific needs:
  - Automation and organization — 10,000 articles x 4 different processing steps = Potential management nightmare
    - Organizational structure makes it easy to ID articles that need processing.
  - Security
  - Flexibility — New tools and document sources should be easy to add to the pipeline
    - New documents are easy — if there's an entry in the database, they'll get processed

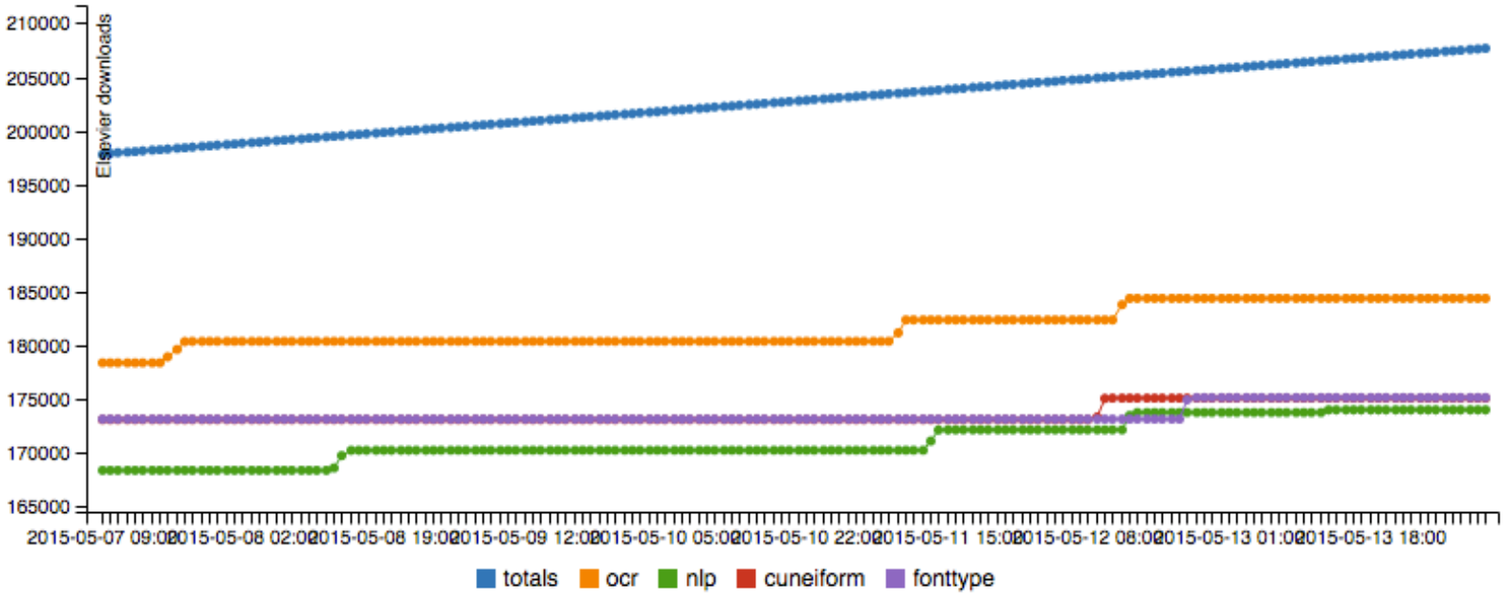
# The Infrastructure Challenge — Processing

- Specific needs:
  - Automation and organization — processing steps = Potential n
    - Organizational structure ma processing.
  - Security
  - Flexibility — New tools and doc to the pipeline
    - New documents are easy — they'll get processed

- Provided by HTCondor!
  - DAGMan + postscript for organization
  - Encrypted filesystem ensures PDFs won't be left exposed on the execute nodes
- A cron'ed python script satisfies the automation/ flexibility requirements

# The Infrastructure Challenge — Monitoring

- Use a round-robin database, make plots of fetched and processed counts for each type of job
  - Have hour/day/week/month views, so we can identify impact at a glance
  - Also provides an estimate of how many CPU hours were used




| Fetched | Successful Jobs | Failed Attempts |
|---|---|---|
| 9882 | 15659 | 2218 |
| | CPU Time | CPU Time |
| | 1202 | 3996 |

# Current Status — System Overview

Processing jobs
(uses encrypted file system)

PDF Fetching (Requires API key + white-listed IP address)

Website

PDFs for OCR processing
(are removed after);
metadata on new articles

Database backup (daily)

Submit node

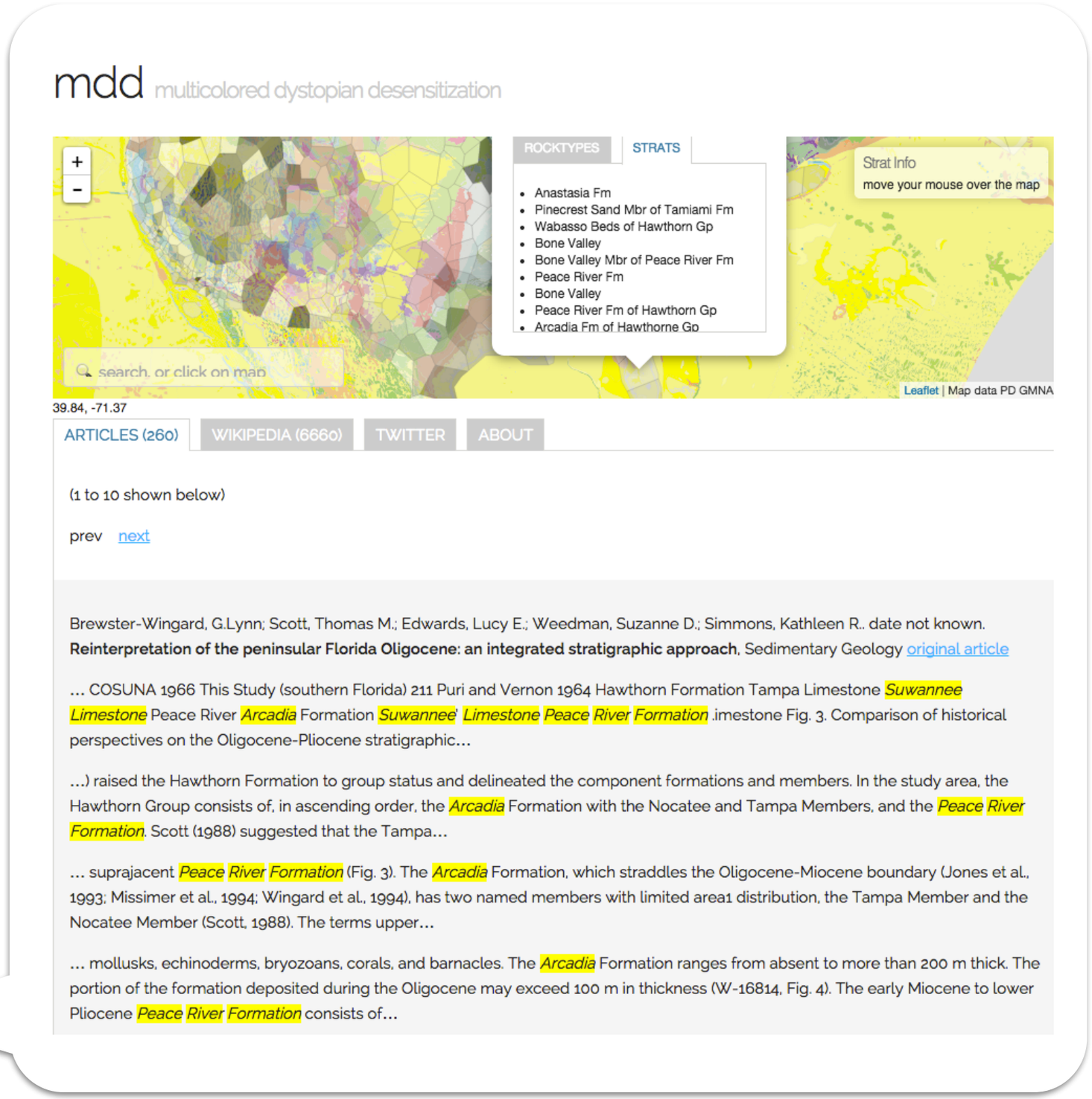
Secure server

**What lives here:** *Processing/harvesting.*
Processed output, databases, web server for monitoring, cronjob for NLP/FontType processing

**What lives here:** *Fetching*. PDFs, daily backup of databases.

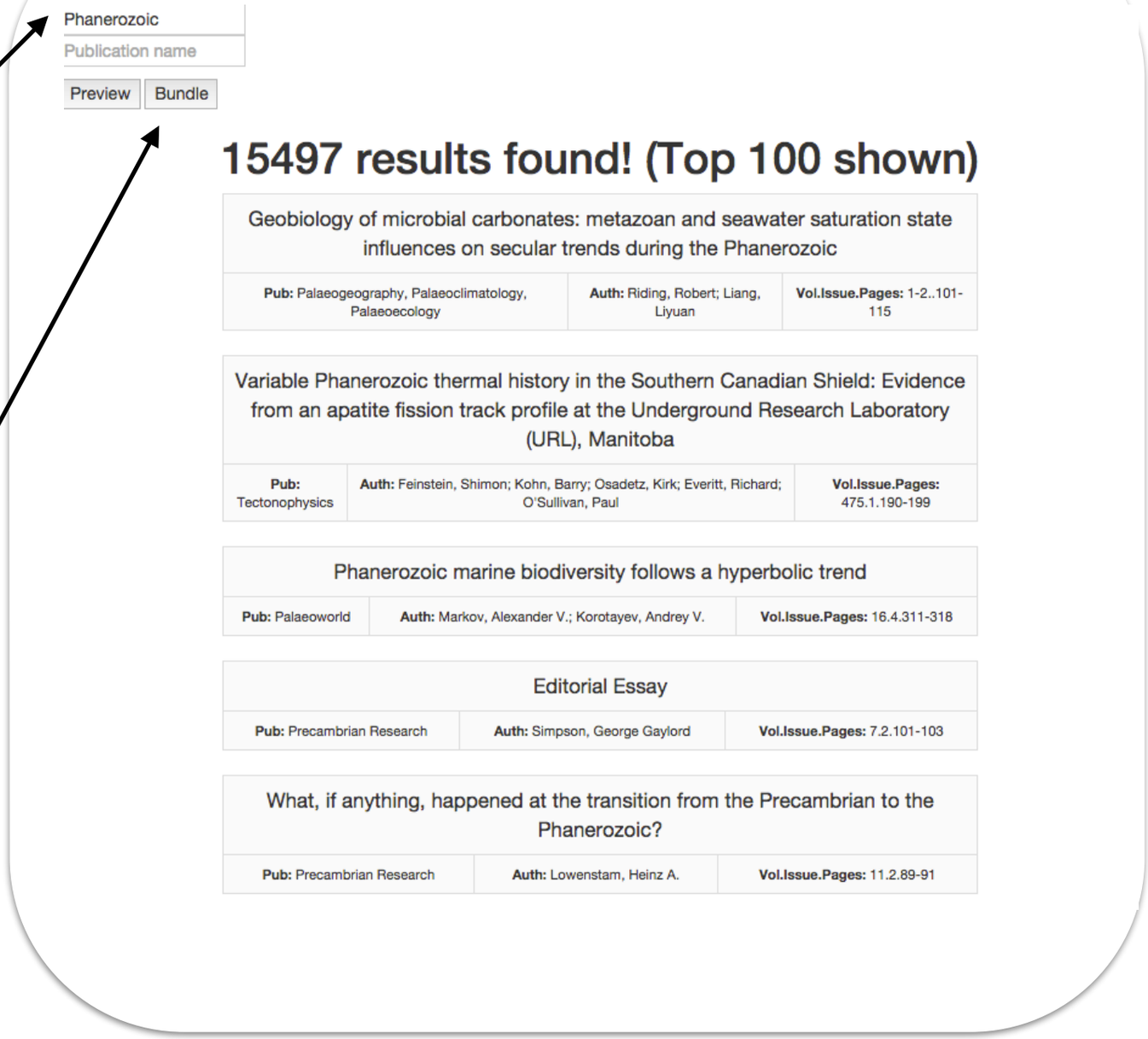
# Current Status

- Infrastructure fully in place
- Work on DeepDive application ongoing
- Applications being built off of the existing articles database:
  - Exposing articles and metrics information via API
  - Fulltext search article discovery application



# Current Status — PaleoDeepDive Input Builder

- Use article search API to define a collection of articles to use in a DeepDive application
  - This example is a fulltext query for **Phanerozoic**
- Preview top search results
- Click “Bundle” to build the DeepDive-ready PostgreSQL databases from the existing NLP and FontType/Layout processes



# Conclusions — Key Infrastructure Features

- Automated document fetching at arbitrary maximum rates determined by content providers (e.g., Elsevier 10K/week/API key)

- Secure document storage; encrypted processing methods to protect content owners/providers

- HTC infrastructure to run core tools (e.g., NLP, OCR, table recognition/ parsing, image analysis); tool versioning and success/failure/quality analysis

- API layer with basic capacity to identify documents of potential relevance to a project, with initial results returned as (augmented) bibJSON

- Packaging and delivery of PaleoDeepDive-ready raw data (e.g., PostgreSQL database of NLP results); everything traceable back to specific sources (original URL and locations within documents)

# Next Steps

- Currently working on:
  - More modular processing framework
  - Additional tools (e.g. figure extraction)
  - Enhanced article discovery that leverages all possible tools
  - Using the infrastructure as a starting point for a DeepDive application!

# Questions?

- deepdive.stanford.edu
- http://deepdive.stanford.edu/doc/paleo.html
- http://journals.plos.org/plosone/article?id=10.1371/journal.pone.0113523