# OCEAN OBSERVATION DOMAIN DISCOVERY WITH SPARKLER

Chris A. Mattmann[1,2], Omid Davtalab[1], Simin Ahmadi Karvigh[1], Siri Jodha S. Khalsa[3], Ruth Duerr[4], Wayne Burke[2], Srinidhi Nandakumar[1,2], Prerana T H M[1], Dixita Patel[1]

University of Southern California, NASA Jet Propulsion Laboratory, University of Colorado, The Ronin Institute

## Abstract

- The goal is to build an end to end dockerized product that any researcher can use to perform domain relevant searches and analyze the collected data.
- The scientific web is a collection of Scientific Data Repositories (SDR) available on the internet. A web-sample study[1] of 100 SDRs in the internet indicates that they are often multidisciplinary and encompass data in variety of formats including multimedia, text, statistical, GIS etc.
- Conventional scrapers and crawlers do not have domain knowledge, lack context and are agnostic to data content. They do not have the ability to selectively crawl parts of SDRs which are specific to a particular domain of interest. They do not possess the ability to understand the contents of a document and predict whether it's of interest or not.

## Sparkler

- We are using *Sparkler*, an open-source, extensible, horizontally scalable crawler which facilitates high throughput and focused crawling of documents pertinent to the polar domain.
- Sparkler uses machine learning techniques to determine the relevancy of collected data and enable domain-specific focused crawling.
- Sparkler avoids disruption of service by partitioning URLs by hostname such that every node gets a different host to crawl and inserts delays between subsequent requests.
- Working on the NSF Wrangler super computer, we scaled our domain discovery pipeline to crawl about 300k ocean specific documents from the scientific web.

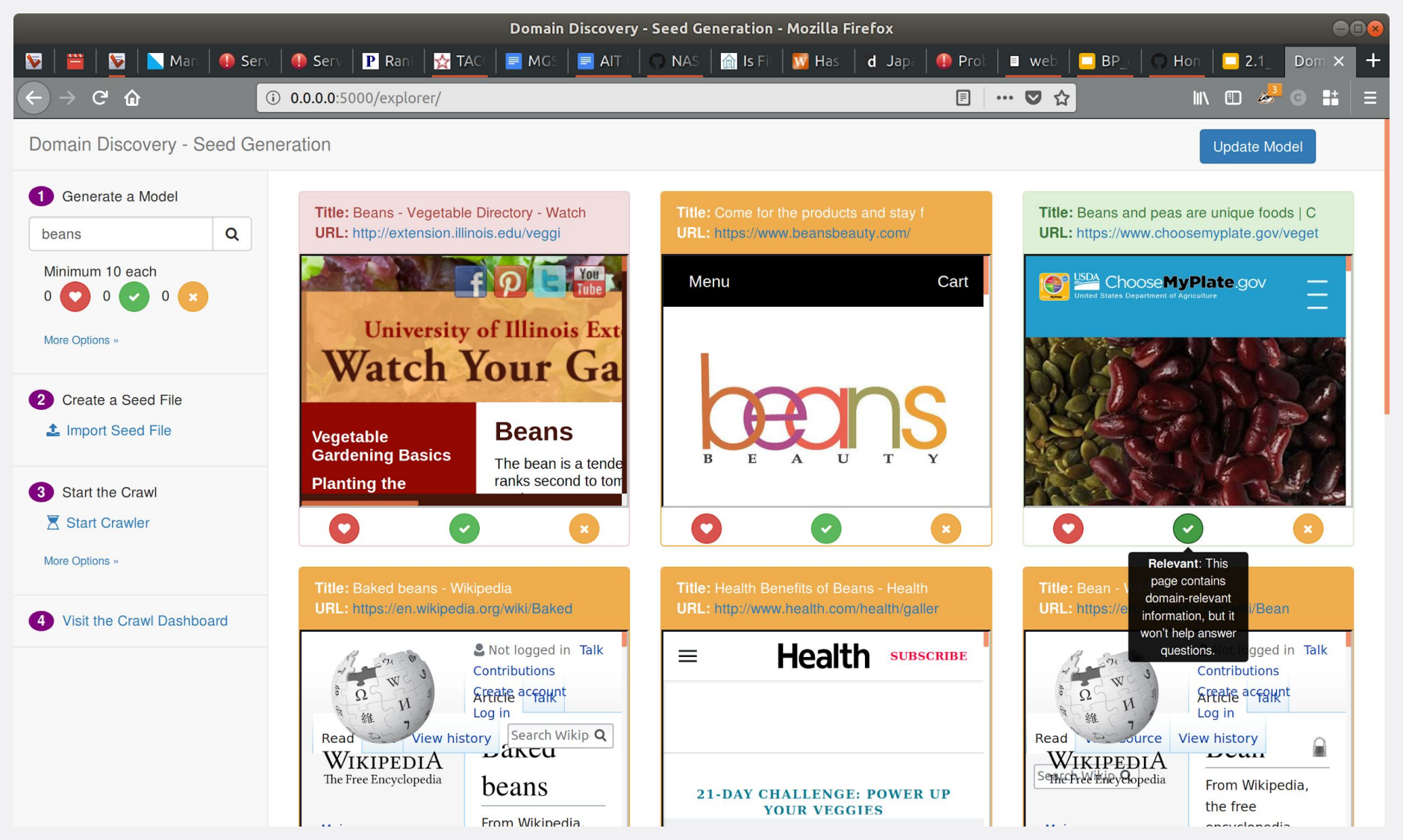


**Figure 1:** Sparkler user interface for training the DRM

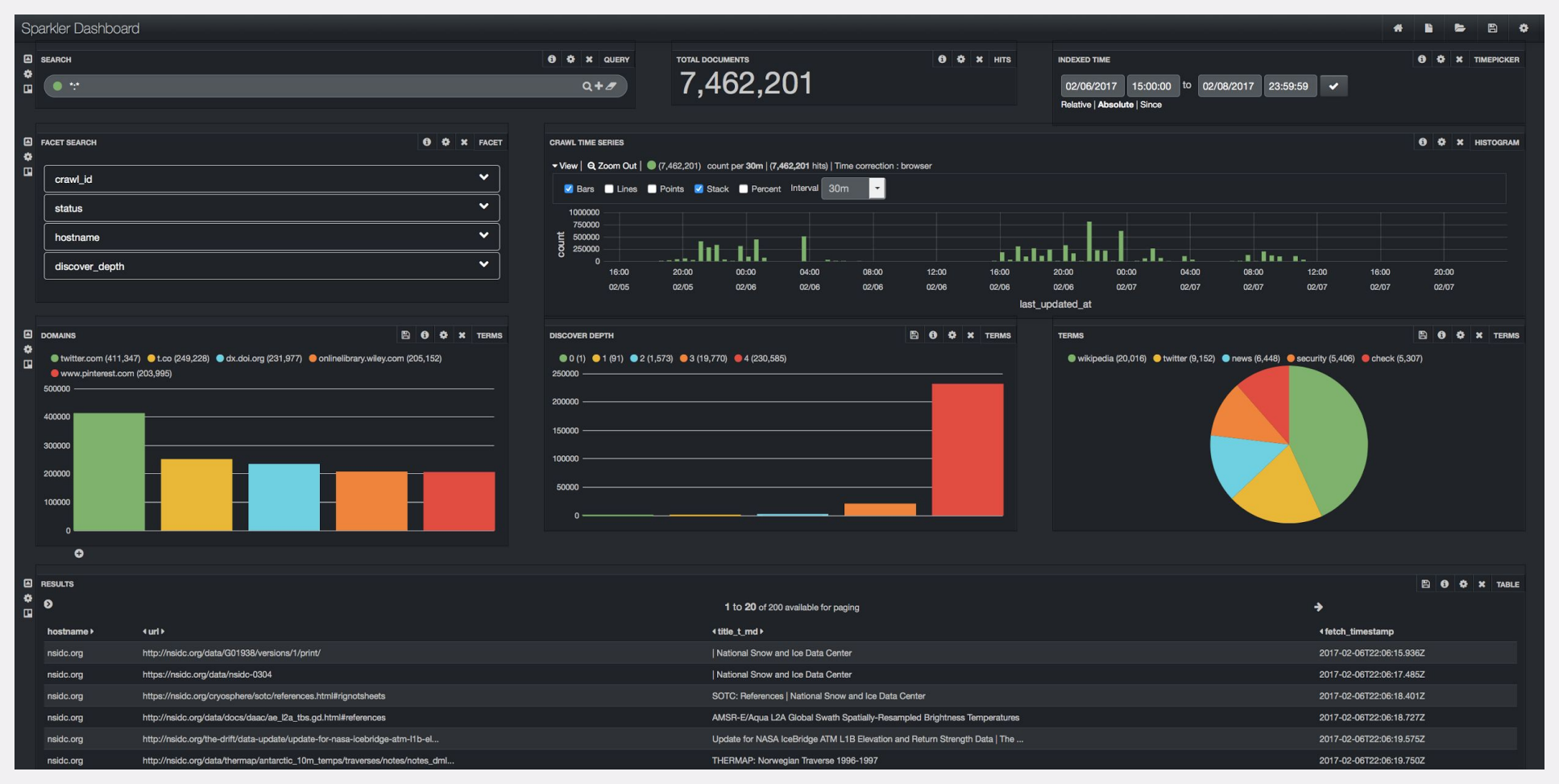
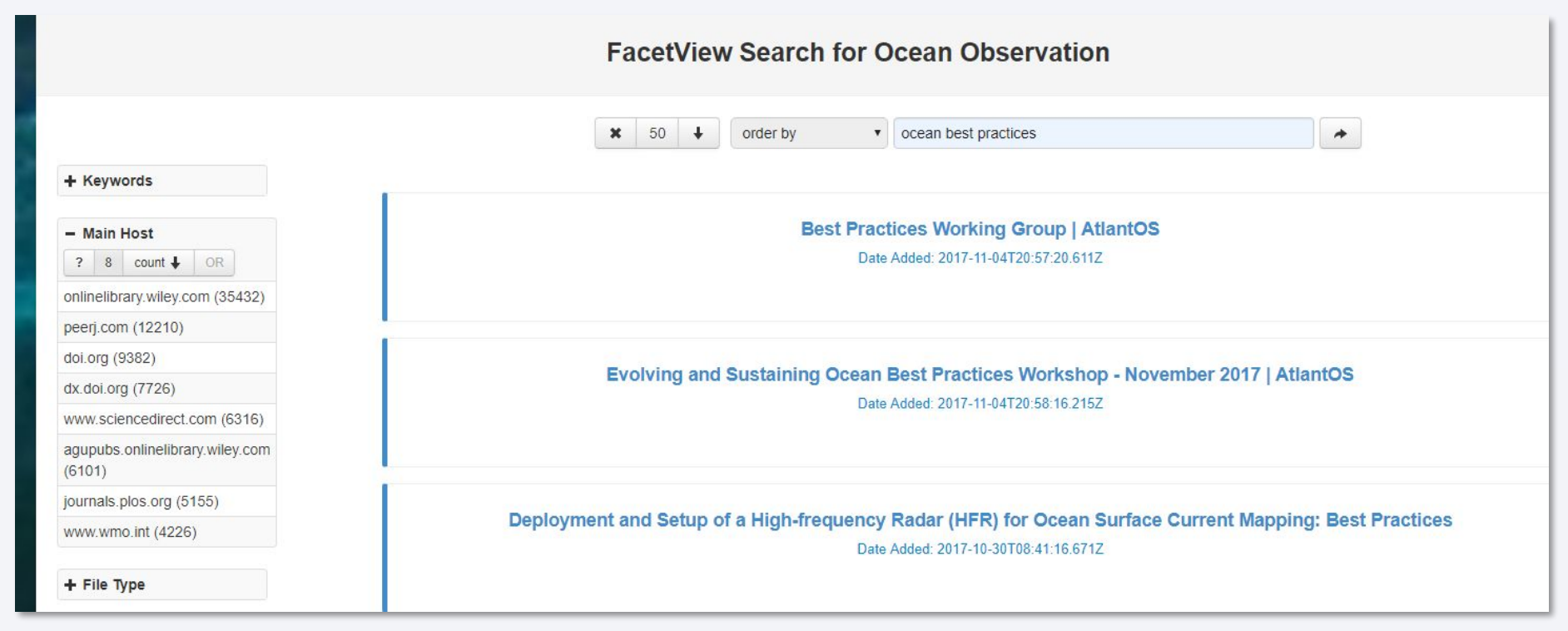


**Figure 2:** A near real time crawl analytics dashboard for Sparkler

## FacetView

We provided a FacetView pointed to the Solr (database) indices from crawling. A user can apply filters to their searches using facets and easily save, share, and consume documents from the Ocean Observation Best Practices domain.



FacetView Search for Ocean Observation

✖ 50 ↓ | order by | ocean best practices

+ Keywords

− Main Host

? 8 count ↓ OR

- onlinelibrary.wiley.com (35432)
- peerj.com (12210)
- doi.org (9382)
- dx.doi.org (7726)
- www.sciencedirect.com (6316)
- agupubs.onlinelibrary.wiley.com (6101)
- journals.plos.org (5155)
- www.wmo.int (4226)

+ File Type

Best Practices Working Group | AtlantOS
Date Added: 2017-11-04T20:57:20.611Z

Evolving and Sustaining Ocean Best Practices Workshop - November 2017 | AtlantOS
Date Added: 2017-11-04T20:58:16.215Z

Deployment and Setup of a High-frequency Radar (HFR) for Ocean Surface Current Mapping: Best Practices
Date Added: 2017-10-30T08:41:16.671Z

**Figure 3 :** Filtered search in FacetView

## Domain Relevance Models ( DRM )

We provide context and domain knowledge to the crawler by building machine learning models (DRM) that are capable of predicting the relevance of a given document to the said domain.

We have built five models namely, Support Vector Machine (SVM), Naïve Bayes (NB), Random Forest (RF), Neural Network (NN) and Cosine Similarity (CS). By performing a general Depth First Search (DFS) crawl for Ocean Observation Best Practices domain, we are evaluating the performance of the models using labeled data from experts in the domain. We use five classes labelled 1 to 5 with Class 5 being highly relevant and Class 1 being least relevant, to represent the relevancy of data to the domain along with its membership probabilities.

Each of the following algorithms produce a probability of membership for the document or data point and the respective classes. The class with the highest membership value is chosen as the active label for that document or data point.

**Support Vector Machine**: Using the SVC decision_function, we evaluate class relevancy with the per-class scores for each sample. The weighted sum of all classes is then used to calculate a score generated by SVM.

**Naive Bayes Classifier**: Using the NB_Score function, we use URL words and a relevancy label (Class relevancy) to integers within the range [1,100]) to calculate a probability score. This score is then used to define an active label based on the membership probabilities.

**Random Forest Classifier**: The class probability of a single tree is the fraction of samples of the same class in a leaf. Thus, the predicted class probabilities of an input sample are computed as the mean predicted class probabilities of the trees in the forest. We use weighted log probabilities to calculate the score of each data point.

**Neural Network Classifier**: We use the MLP NN Classifier to evaluate data relevancy and use Cross Entropy Loss to minimize errors. Using the predict_proba function, we process the membership probabilities of each data point and classify them.

Cosine Similarity provides a similarity to the most relevant documents.

**Cosine Similarity**: It evaluates document similarity by computing the average similarity metric with respect to the top 5 ranked golden documents. This model thus measures the relevancy between each document and the most relevant documents.

## Process

We ran a broad crawl using Sparkler on an initial seed list of 28 URLs deemed relevant to Ocean Observation Best Practices domain experts. Raw data collected over a period of 2 weeks was then used to evaluate domain relevancy. We used a predefined keyword list to build a keyword vector that was used by different machine learning models. These models provided class membership probabilities for each document and the class with maximum membership value was chosen as the active class of relevancy. Further, we evaluated the model performance using two metrics - Train and Test Accuracy. Figure 4 shows the detailed process and pipeline.

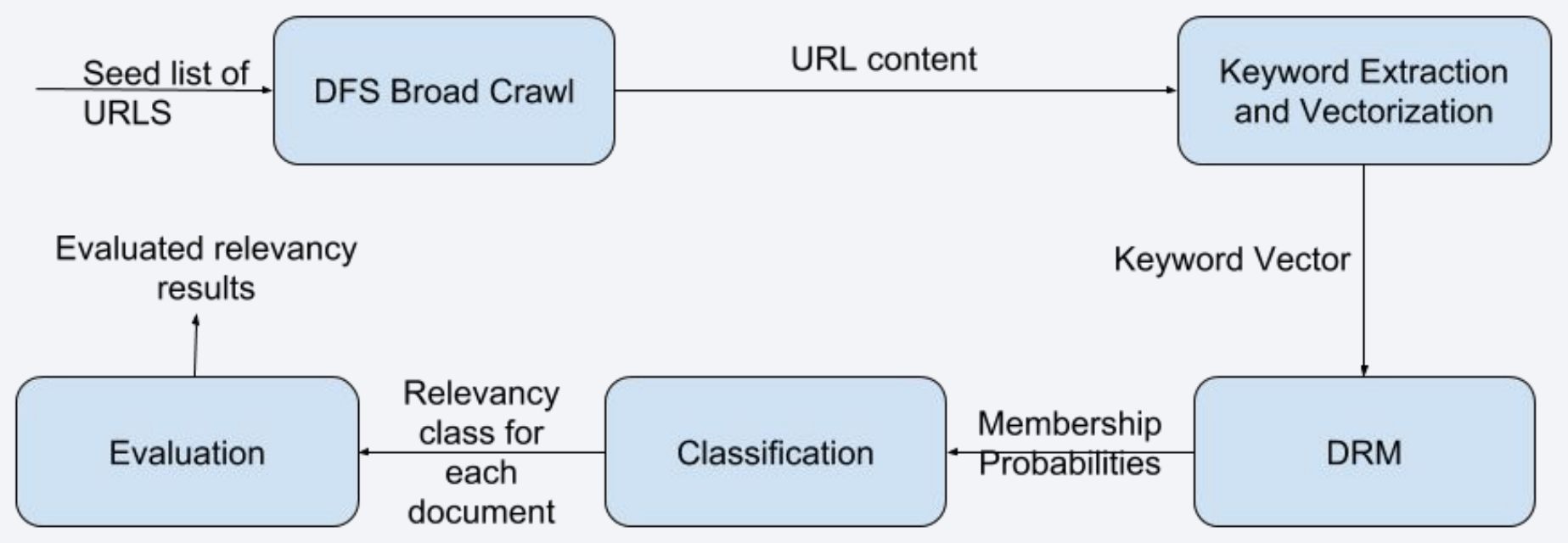


**Figure 4 :** Process Pipeline and Outputs

To evaluate Model Accuracy, we use two techniques, namely: 3-Class Evaluation and 5-Class Evaluation. 3-Class Evaluation considers Class 1, (Class 2, Class 3, Class 4) and Class 5 measures, whereas 5-Class Evaluation uses Class 1, Class 2, Class 3, Class 4, Class 5 measures. We use Test and Train Accuracies to evaluate the different models.

**Table 1: Model Evaluation Results**

| Model | 5-Class Test Accuracy | 3-Class Test Accuracy | Train Accuracy |
|---|---|---|---|
| SVM | 0.25 | 0.4 | 0.29444 |
| Naive Bayes | 0.3 | 0.4 | 0.4 |
| Neural Networks | 0.55 | 0.55 | 0.97778 |
| Random Forest | 0.55 | 0.65 | 0.97778 |

From the above results we see that the Random Forest Classifier provides the best results in terms of test accuracy. We conclude that this is due to the unique property of RF to avoid overfitting problems. We also note that NN shows average performance due to the lack of sufficient data. Despite this limitation, NN performs better than Naive Bayes or SVM Classifier.

We also observe that 3-Class Test Accuracies are higher than 5-Class Test Accuracies and thus conclude that it is a suitable metric for evaluation. This attributes to the fact that documents falling within the classes 2,3,4 happen to have very marginal difference in similarity.

This pipeline can be replicated to any domain. It provides the user with a well defined model supplemented with flexibility to choose from different topic models and filters for search. To ensure improved accuracy, the user can train a wide range of machine learning models that suit their dataset generated from an intelligent and controlled scientific web crawl.

## Polar Deep Insights

The Polar Deep Insights (PDI) is a tool that can be used for generic content extraction and evaluation of any dataset. We have built a content extraction, enrichment and rich visualization interface to explore the spatial-conceptual-temporal richness of the Polar TREC dataset. Currently we are dockerizing this tool to enable easy installation and use for any and all domains. PDI has two parts: Insight Generator and Insight Visualizer.

**Insight Generator**: This is a python library which provides an interface to extract entities, locations, file metadata and measurements from documents.
**Insight Visualizer**: This facilitates building an 'ontology-of-interest' using the 'concept editor' interface. Users can gain insights of the extracted content from the insight generator module through the 'query interface'.
The incredible part about this tool is that it is extensible towards any domain and any dataset. Figure 5 shows the architecture and different components of PDI.

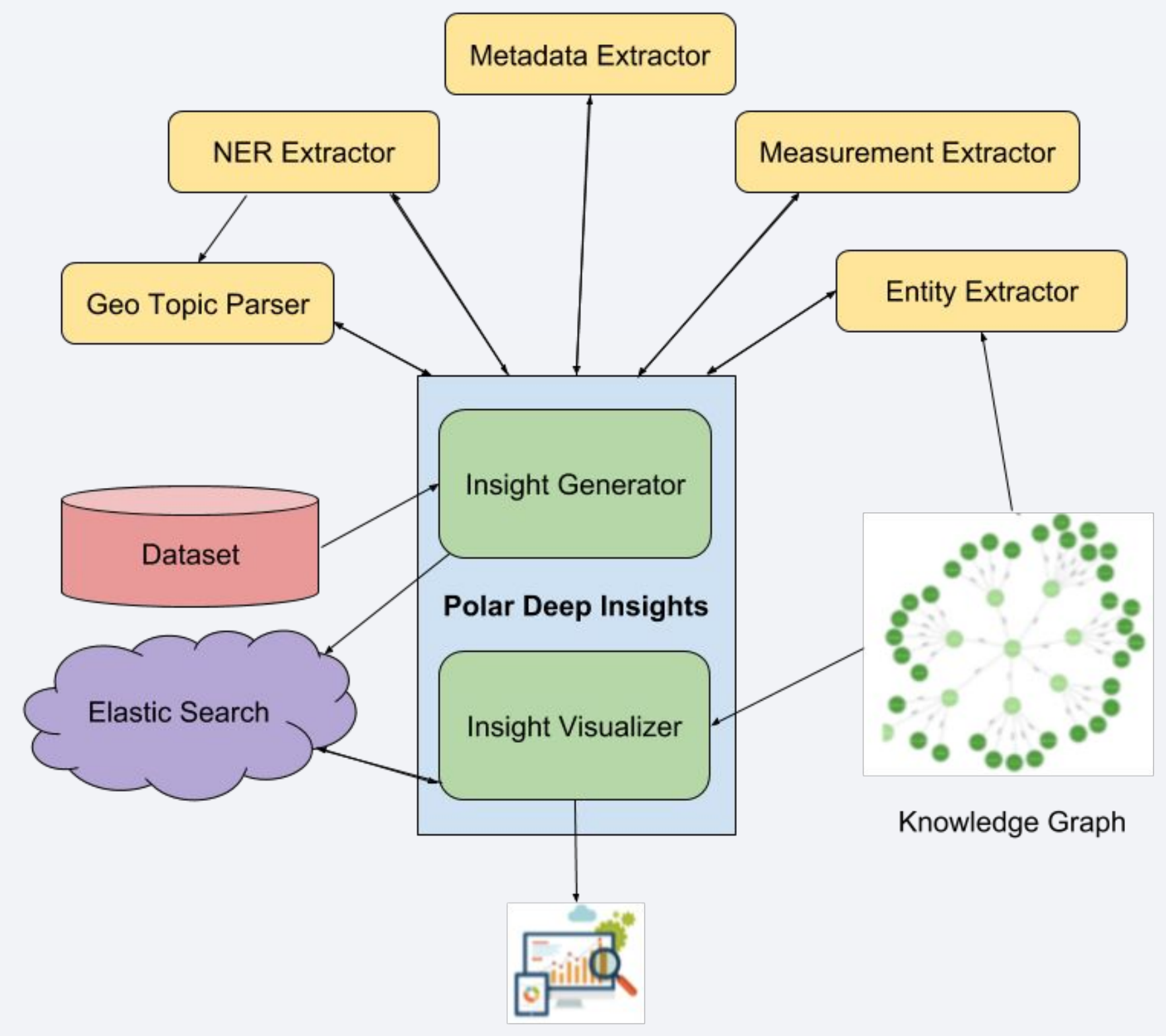


**Figure 5 :** Polar Deep Insights Architecture

We have built a docker image that supports easy and quick install of the interface. It also allows to use to customize features for extraction and datasets for visualization. PDI thus provides an in-depth analysis of the intelligent crawl results established using DRMs.

## Future Work

- Test different model evaluation techniques to determine the best model.
- Complete the dockerization of PDI so that it is easily usable by researchers.
- Add topic modelling capabilities to PDI.
- Improve all documentation and solidify the full product.
- Identify end users to test the product and build a user base.

## References

- Polar Deep Insights Earth Cube : http://bit.ly/2rEuWvn
- Sparkler: https://github.com/USCDataScience/sparkler
- PDI: https://github.com/USCDataScience/polar-deep-insights
- Ocean Observation Best Practices Working Group: https://www.atlantos-h2020.eu/project-information/best-practices/