IN23D-0898

AGU100 ADVANCING EARTH AND SPACE SCIENCE

Reusing *Data and* *Metadata* *to Create New Metadata* *through Machine-learning* *& Other Programmatic Methods*

JUSTIN GOSSES[1], *ANTHONY R. BUONOMO*[1], *BRIAN A. THOMAS*[1], *EVAN TAYLOR YATES*[1], *RENA W. YUAN*[1,2], *JENNA M. HORN*[1]

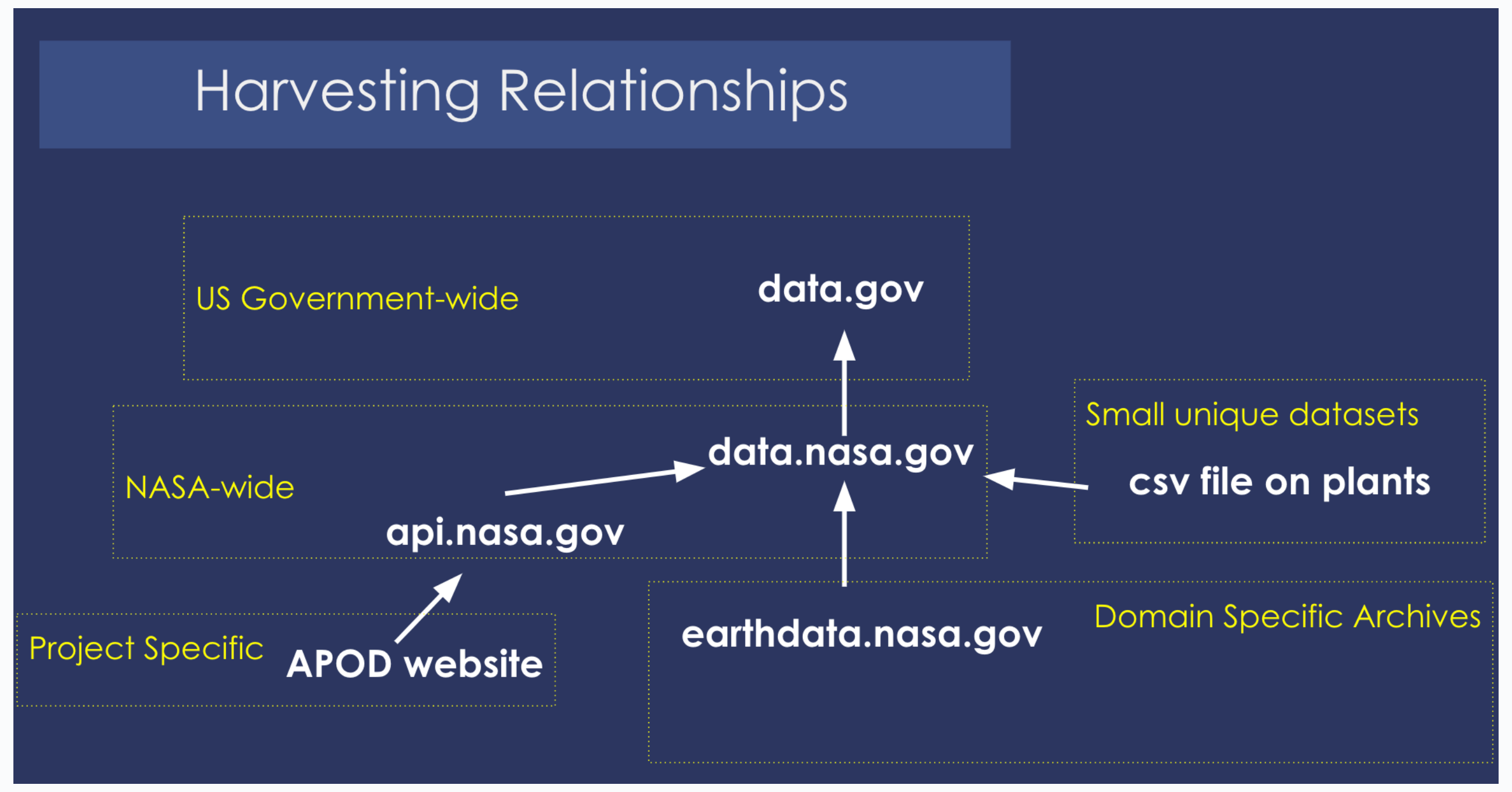
[1]NASA Office of the Chief Information Officer, [2]U.S. Dept. Agriculture.   CONTACT: Justin.C.Gosses@nasa.gov or Brian.a.Thomas@nasa.gov

INTRODUCTION

OUR CONTEXT

We describe two projects focused on improving discoverability at DATA.NASA.GOV & CODE.NASA.GOV via reuse of metadata.

These sites release aggregations of NASA's data & code to the public. They mostly hold metadata, not the actual datasets and code. They harvest metadata from other NASA sites and in turn supply it to government-wide data sites like code.gov & data.gov.



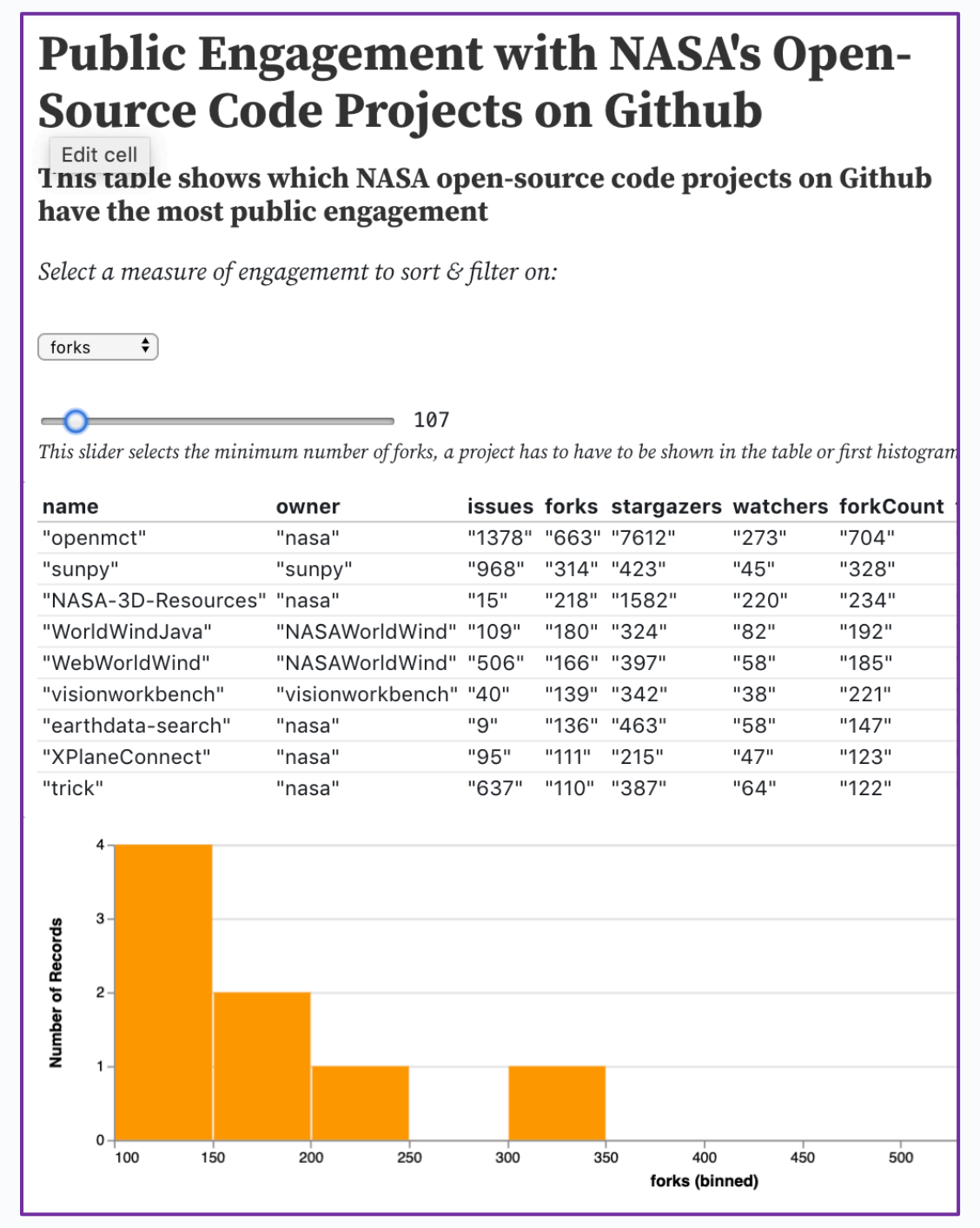
These projects were a collaboration between NASA's OCIO (Office of Chief Information Officer)'s open-innovation program (code.nasa.gov, data.nasa.gov, api.nasa.gov, open.nasa.gov) and OCIO's data analytics team that develops prototypes.

Future State We're Working Towards:

1. Improved discoverability through ML-assisted tagging.
2. Reusable code. Example: ML model in project 2 was reused.
3. Deploy keyword model as public API (so other agencies can use)
4. Encourage NASA & public to build their own user interfaces & visualizations by exposing metadata in a JSON. Examples from the code.nasa.gov JSON:

Interactive notebooks exploring different aspects of NASA's open-source code projects in aggregate.

https://observablehq.com/collection/@justingosses/nasa-metadata


Public Engagement with NASA's Open-Source Code Projects on Github

This table shows which NASA open-source code projects on Github have the most public engagement

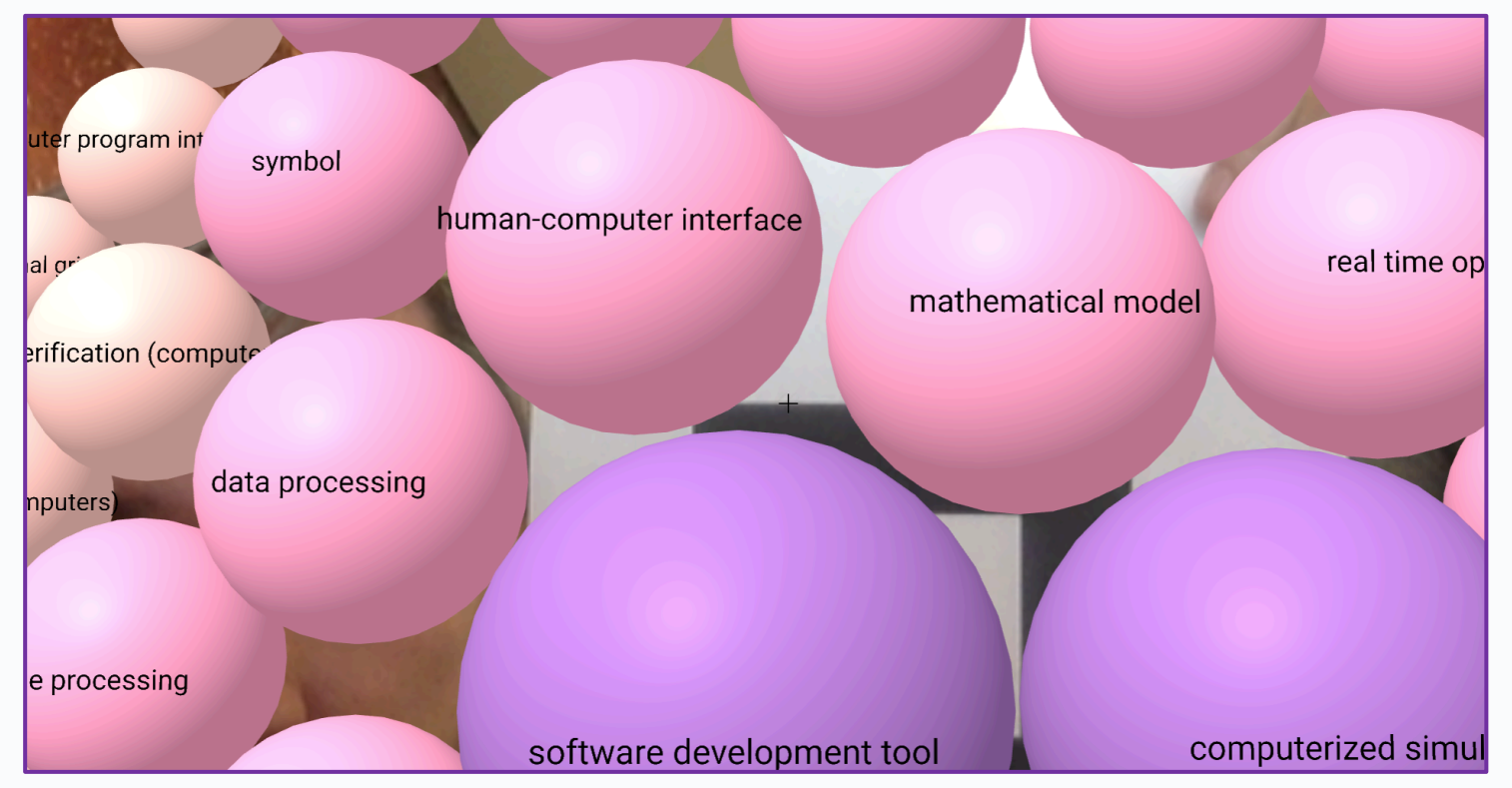
Select a measure of engagememt to sort & filter on:

This slider selects the minimum number of forks, a project has to have to be shown in the table or first histogram

name	owner	issues	forks	stargazers	watchers	forkCount
"openmct"	"nasa"	"1378"	"663"	"7612"	"273"	"704"
"sunpy"	"sunpy"	"968"	"314"	"423"	"45"	"328"
"NASA-3D-Resources"	"nasa"	"15"	"218"	"1582"	"220"	"234"
"WorldWindJava"	"NASAWorldWind"	"109"	"180"	"324"	"82"	"192"
"WebWorldWind"	"NASAWorldWind"	"506"	"166"	"397"	"58"	"185"
"visionworkbench"	"visionworkbench"	"40"	"139"	"342"	"38"	"221"
"earthdata-search"	"nasa"	"9"	"136"	"463"	"58"	"147"
"XPlaneConnect"	"nasa"	"95"	"111"	"215"	"47"	"123"
"trick"	"nasa"	"637"	"110"	"387"	"64"	"122"

Code.Nasa.Gov/AR

Web-based Augmented Reality User-Interface. "in progress"



TWO PROJECTS

PROJECT 1: PROGRAMMATICALLY CREATE METADATA OF FILE TYPE & DATASET ATTRIBUTES FROM DATA FILES

DATA.NASA.GOV

DATA: Tens of thousands of datasets described on DATA.NASA.GOV.

METHODS: Leverage the download link field on data.nasa.gov, webscraping, and some python tools to generate metadata that describe files.

RESULTS: FAILURE- REUSE NOT POSSIBLE DUE TO METADATA LIMITATIONS

DISCUSSION: Goal was to generate metadata like file type (CSV, PNG, TIFF), file size, and attributes (# of instances, rows & columns; strings vs. numbers, etc.) This data would be used by end-users to filter results and theoretically inform a model for predicting what datasets work well for different tasks. Howerver, link listed in "download link field" rarely led directly to a file. Large variance in required navigation across 88 sites that feed into data.nasa.gov made it impossible to get files via web-scraping.

LESSONS LEARNED: FOR THIS IDEA TO WORK, "DOWNLOAD LINK" METADATA FIELD NEEDS TO BE A DIRECT LINK TO FILE(S) OR SOMETHING THAT CAN BE PROGRAMMATICLY USED TO GENRATE LINK TO FILES, NOT TO A WEB PAGE WHICH REQUIRES ADDITIONAL HUMAN NAVIGATION.

PROJECT 2: USE A MACHINE-LEARNING MODEL TO CREATE KEYWORD TAGS FROM CODE PROJECT DESCRIPTION TEXT

CODE.NASA.GOV

Try it out! Website includes A.I.-Generated metadata

DATA: 500+ open-source code projects on CODE.NASA.GOV.

METHODS: Used paragraph length project descriptions as input into machine-learning model that predicts keywords. Model described below.

RESULTS: SUCCESS- IMPROVED DISCOVERABILITY VIA BETTER & MORE TAG

DISCUSSION: Performance was good enough to automate keyword predictions into every new code project on CODE.NASA.GOV. Mistakes tend to be around short description (<20 words) or repeated words with unusual meaning, ie *WorldWind* software, which has nothing to do with wind.

LESSONS LEARNED: A MODEL TRAINED ON PAPER ABSTRACTS GENRALIZED WELL TO PREDICTING KEYWORD TAGS FOR CODE PROJECT DESCRIPTIONS. ONE METADATA REUSED TO CREATE ANOTHER.

The STI Tagger (Our ML model)

The STI tagger can automatically assign NASA keywords to text. The system's models were trained upon about 3 million manually tagged NASA documents, and it can automatically tag from a selection of about 7,000 keywords. Training data comes from STI.NASA.GOV

LEARN ABOUT OUR NATURAL LANGUAGE PROCESSING MODEL

https://go.usa.gov/xpmdw

CONCLUSIONS

ML CREATED METADATA HAS BOTH BENEFITS & COSTS

ML GENERATED KEYWORDS CAN AUGMENT HUMAN KEYWORDS IMPROVING DATASET DISCOVERABILITY

GENERATING NEW METADATA FROM THE DATA FILES THEMSELVES IS DIFFICULT TO IMPOSSIBLE WHEN SITE ARCHITECTURES ASSUME HUMAN NAVIGATION TO DOWNLOAD DATA FILES

DISCUSSION

PROJECT 2: ML-GENERATED KEYWORD DISCUSSION

HUMAN- VS. ML-GENERATED KEYWORDS

KEYWORD TAGS	HUMAN-GENERATED	MACHINE-GENERATED
How often tags added?	Once, on dataset load	More than once, model updates for example.
Tags reflect whose perspective?	Dataset supplier	Training data + filtered to match user needs
Tag meaning uniformity?	Different people use different words for same thing	Standardized tag list. Optional hierarchal relationships.
Number of tags?	Commonly 0-8. Sparse tags are a problem.	More than a human. Limit to top X number of tags by predicted accuracy.

KEY BENEFITS OF ML GENERATED KEYWORDS
Results in more keywords of better average quality than dataset owner supplied keywords
Possible to add into DevOps pipeline that automates new additions without human admin work.

KEY COSTS OF ML GENERATED KEYWORDS
Requires additional machine-learning skillset that may not always be present
DevOps: automating new API calls and tests requires additional type of maintenance work
Additional server cost and security requirements
False tags occur a bit more often which places a new burden on end-user