Softcite:
Data-Driven Software Visibility in Science

James Howison[1], Patrice Lopez[2], Caifan Du[1], Norman Gilmore[3], Johanna Cohoon[1], Nick Adams[3], Karthik Ram[4]

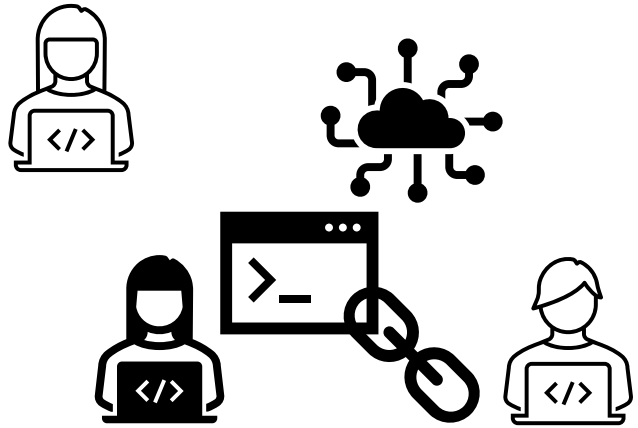
[1] School of Information, The University of Texas at Austin
[2] SCIENCE-MINER
[3] TagWorks Inc.
[4] Berkeley Institute for Data Science

Howison Lab @ TEXAS
science-miner
TagWorks
BERKELEY Institute for Data Science
ALFRED P. SLOAN FOUNDATION

What is *Softcite*?

A discovery engine for identifying research software solutions

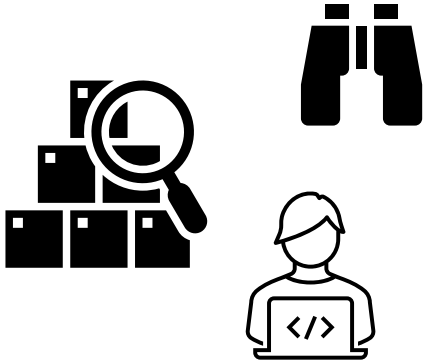
The goal of *Softcite* **is to increase visibility of research software, and inform technology decisions for data-intensive research**



Why Software Visibility?

Currently, software still stays largely outside research databases and systems of information retrieval.

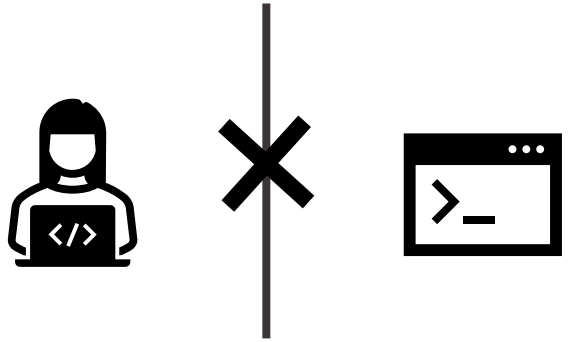
—which means you cannot search for software as easily as searching for research papers.



In the Absence of Software Visibility…

One result of this lack of software visibility:

—Users miss good software
—Good software misses its users



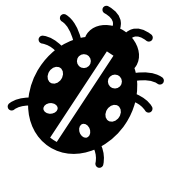
“How do I know what to use for my data science project?”

In the Absence of Software Visibility…

More disadvantages:

—Undermine reproducibility of analysis

—Non-interoperable software stacks make collaboration hard



—Hard to identify, and thus acknowledge, the contribution of software developers

Software Disadvantages Motivate *Softcite*…

—Discover software used in research and analysis

→ Increase software visibility

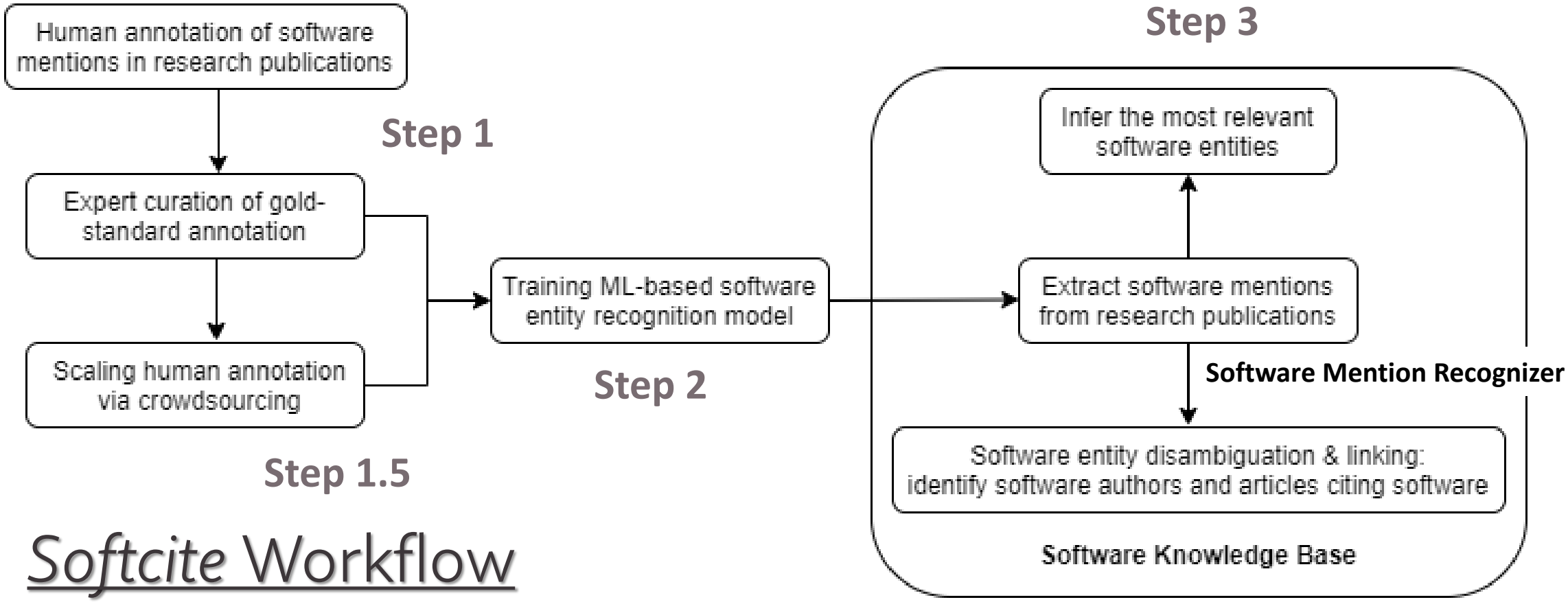
—Link software to its authors and application

Softcite achieves these goals by mining mentions of software in open access research publications

—Make visible relevant software pieces

→ Inform technology decisions (for building your stack)

How *Softcite* Achieves its Goals?



Softcite Workflow

Howison Lab @ TEXAS

science-miner

TagWorks

BERKELEY Institute for Data Science

How *Softcite* Achieves its Goals?

Softcite Workflow

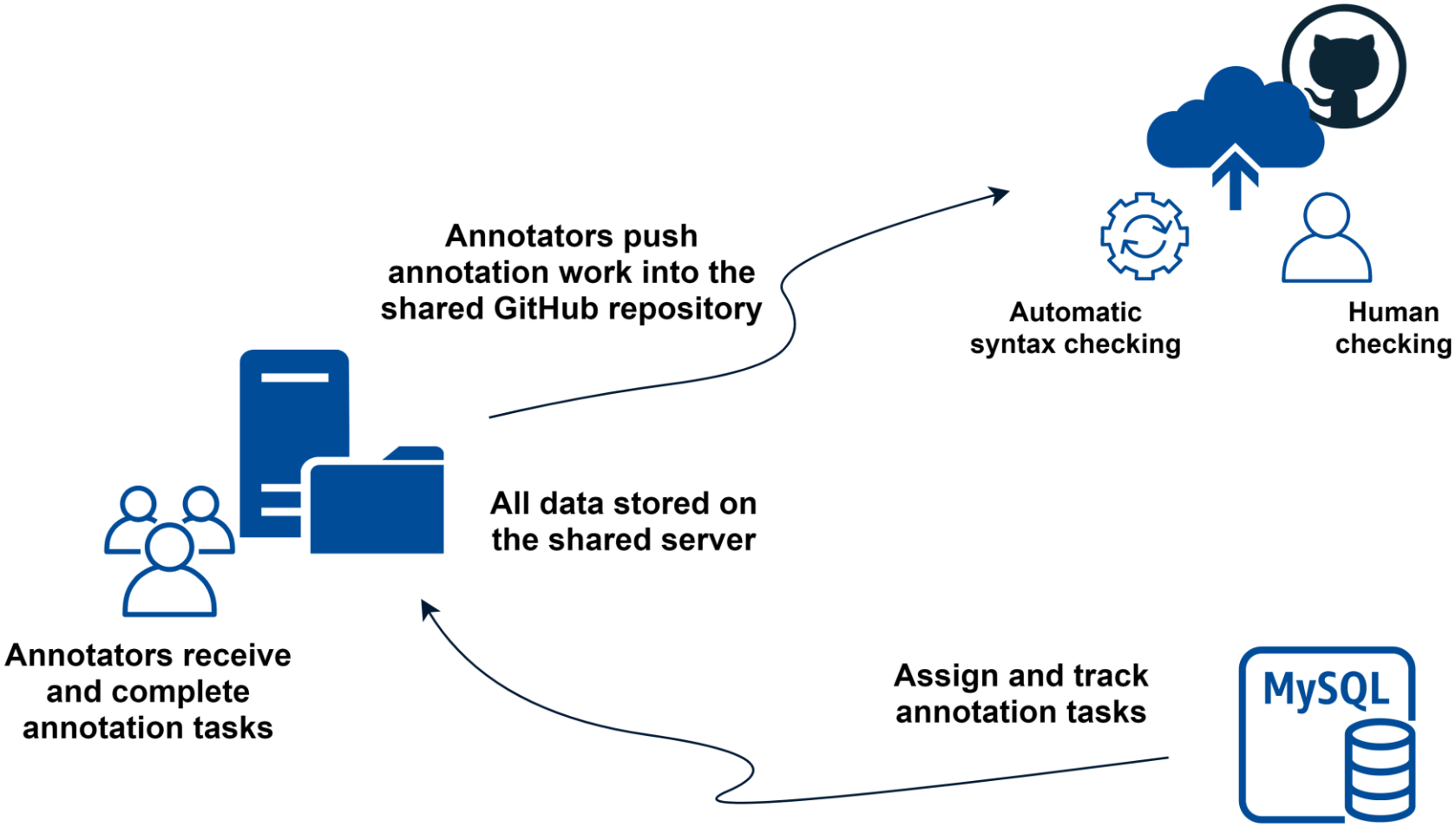
Step 1: Build a gold-standard dataset of annotated software mentions in research publications

Step 1.5: Scaling human annotation of software mentions via crowdsourcing

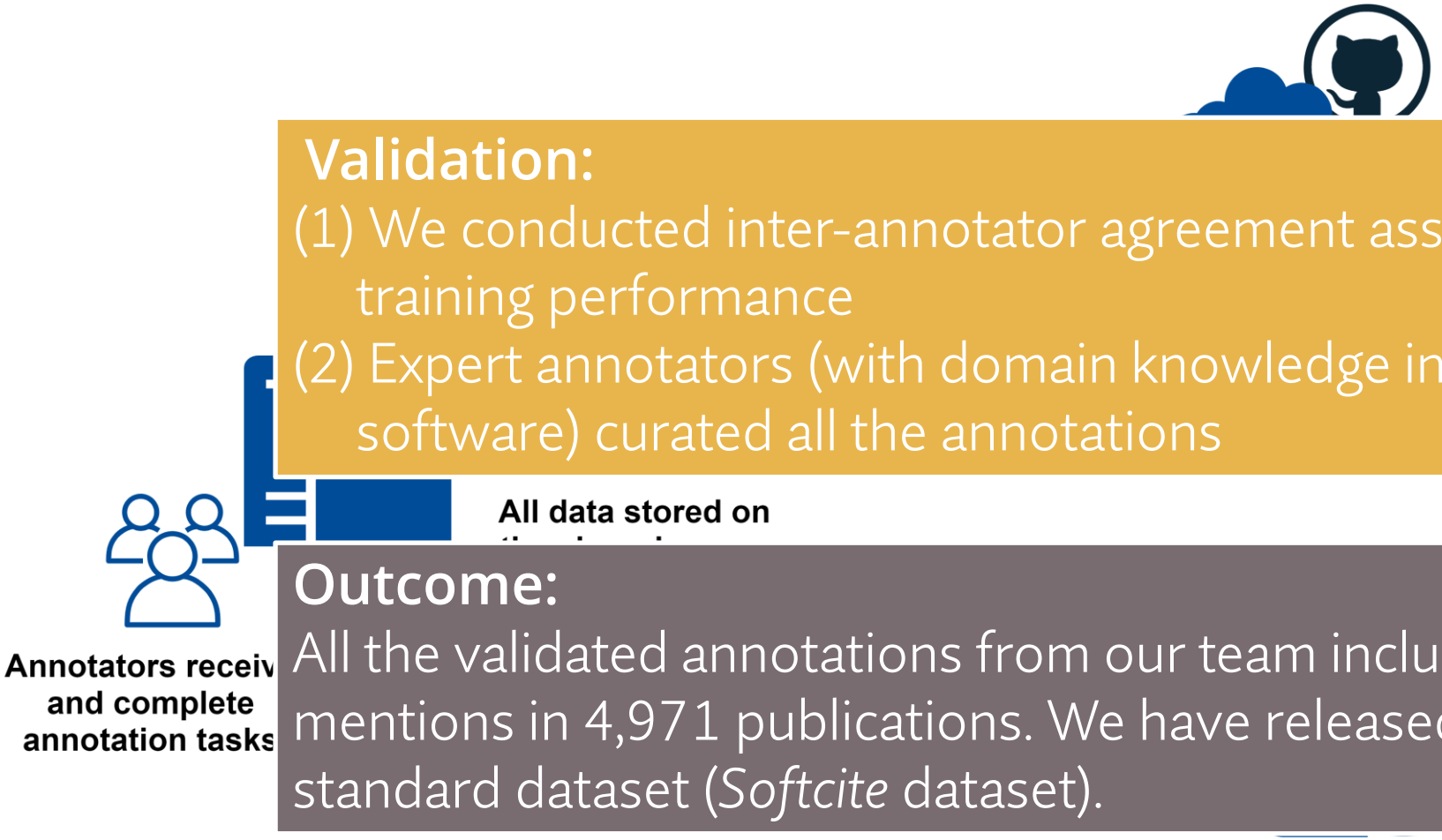
Step 2: Using gold-standard annotation to train sequence labeling models for automatic extraction of software mentions

Step 3: Construct software knowledge base

Step 1: Human Annotation



Step 1: Human Annotation



Validation:

(1) We conducted inter-annotator agreement assessment to check training performance

(2) Expert annotators (with domain knowledge in NLP & research software) curated all the annotations

Outcome:

All the validated annotations from our team include 4,093 software mentions in 4,971 publications. We have released it as an open gold-standard dataset (*Softcite* dataset).

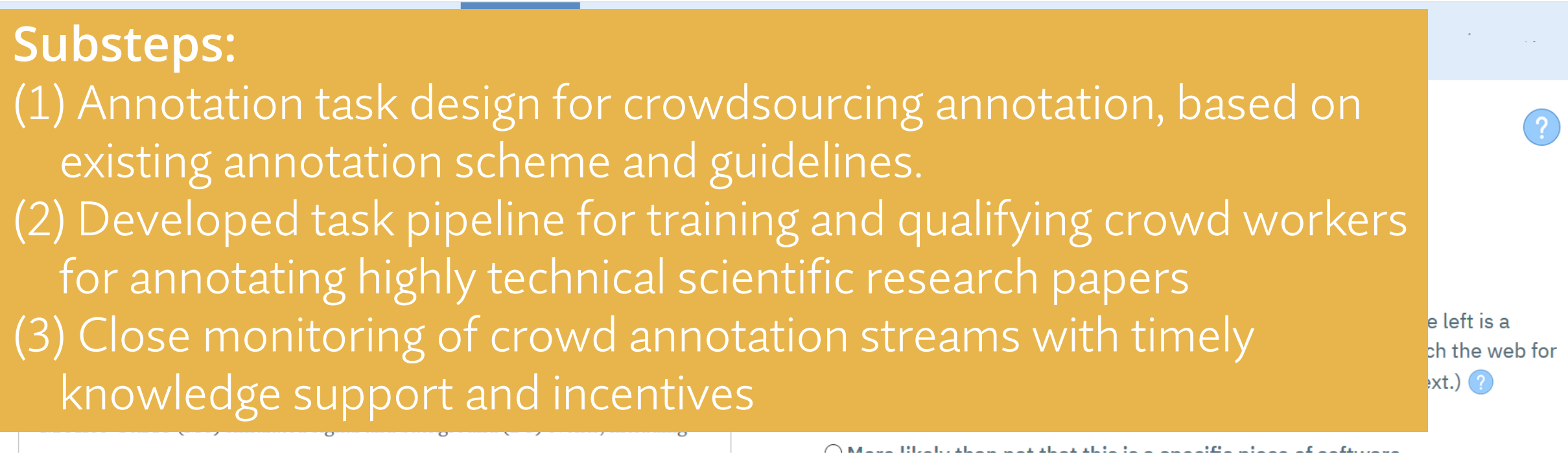
Step 1.5: Crowdsourcing Annotation

Substeps:

(1) Annotation task design for crowdsourcing annotation, based on existing annotation scheme and guidelines.

(2) Developed task pipeline for training and qualifying crowd workers for annotating highly technical scientific research papers

(3) Close monitoring of crowd annotation streams with timely knowledge support and incentives



Outcome:

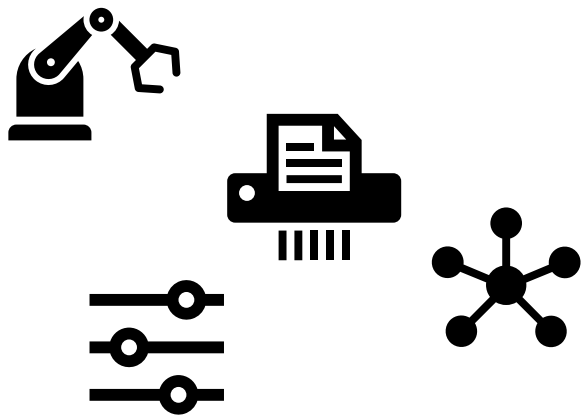
TagWorks collected 11,454 task responses on 2,743 article fragments from Mechanical Turk workers in one month

Howison Lab @ TEXAS science-miner TagWorks BERKELEY Institute for Data Science

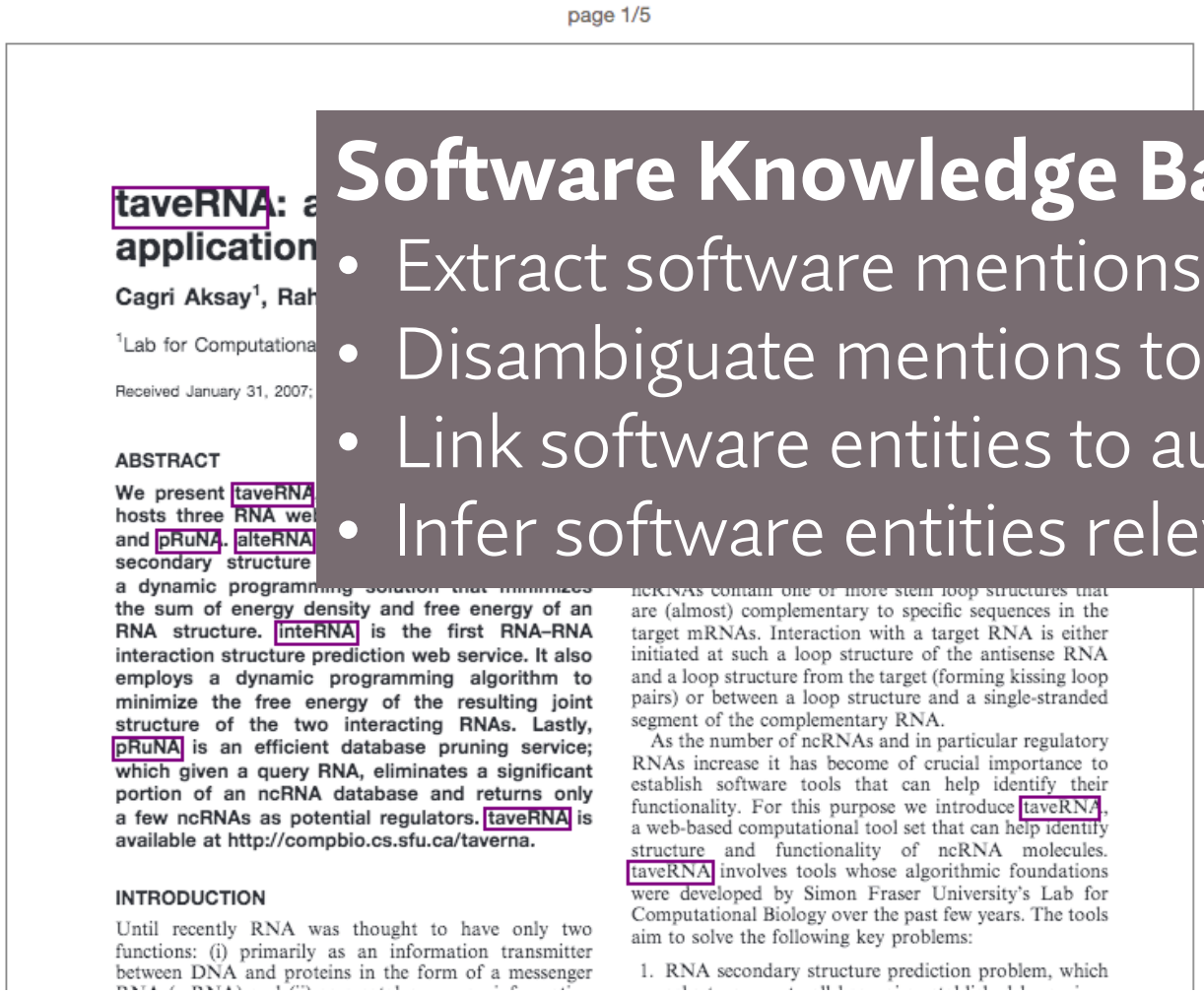
Step 2: Modeling Training

A set of sequence labeling algorithms for software entity recognition have been trained, benchmarked, and implemented in *GROBID software-mentions module.*
*(*https://github.com/ourresearch/software-mentions*)*

- CRF
- BiLSTM-CRF with GloVes embeddings
- BiLSTM-CRF with GloVes + Elmo embeddings
- BERT-base-en + CRF
- SciBERT + CRF



Step 3: Knowledge Base Construction


page 1/5

taveRNA: a
application

Cagri Aksay[1], Rah

[1]Lab for Computationa

Received January 31, 2007;

ABSTRACT

We present taveRNA
hosts three RNA we
and pRuNA. alteRNA
secondary structure
a dynamic programming solution that minimizes the sum of energy density and free energy of an RNA structure. inteRNA is the first RNA–RNA interaction structure prediction web service. It also employs a dynamic programming algorithm to minimize the free energy of the resulting joint structure of the two interacting RNAs. Lastly, pRuNA is an efficient database pruning service; which given a query RNA, eliminates a significant portion of an ncRNA database and returns only a few ncRNAs as potential regulators. taveRNA is available at http://compbio.cs.sfu.ca/taverna.

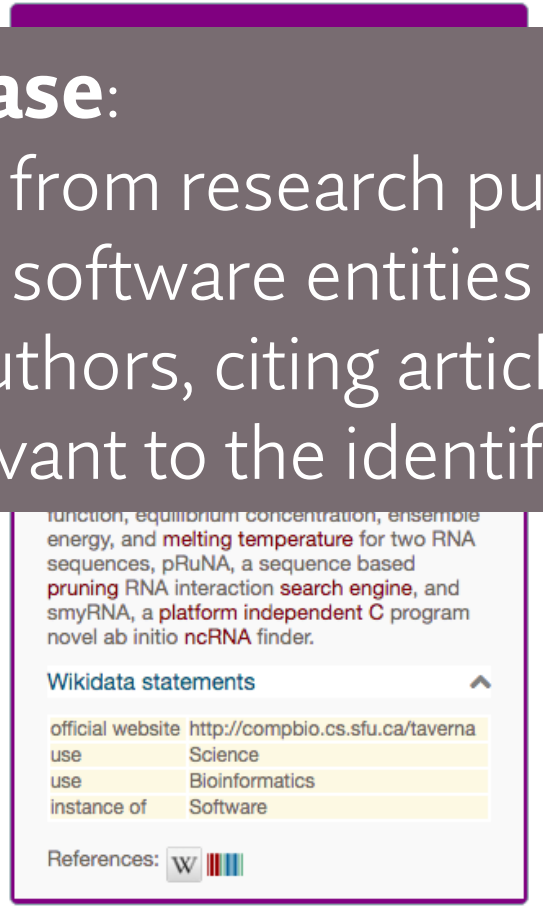
INTRODUCTION

Until recently RNA was thought to have only two functions: (i) primarily as an information transmitter between DNA and proteins in the form of a messenger

ncRNAs contain one or more stem loop structures that are (almost) complementary to specific sequences in the target mRNAs. Interaction with a target RNA is either initiated at such a loop structure of the antisense RNA and a loop structure from the target (forming kissing loop pairs) or between a loop structure and a single-stranded segment of the complementary RNA.

As the number of ncRNAs and in particular regulatory RNAs increase it has become of crucial importance to establish software tools that can help identify their functionality. For this purpose we introduce taveRNA, a web-based computational tool set that can help identify structure and functionality of ncRNA molecules. taveRNA involves tools whose algorithmic foundations were developed by Simon Fraser University's Lab for Computational Biology over the past few years. The tools aim to solve the following key problems:

1. RNA secondary structure prediction problem, which


function, equilibrium concentration, ensemble energy, and melting temperature for two RNA sequences, pRuNA, a sequence based pruning RNA interaction search engine, and smyRNA, a platform independent C program novel ab initio ncRNA finder.

Wikidata statements

official website	http://compbio.cs.sfu.ca/taverna
use	Science
use	Bioinformatics
instance of	Software

References:

Software Knowledge Base:

- Extract software mentions from research publications
- Disambiguate mentions to software entities
- Link software entities to authors, citing articles, etc.
- Infer software entities relevant to the identified software entities

Howison Lab @ TEXAS

science-miner

TagWorks

BERKELEY Institute for Data Science

Softcite Outcomes

***Softcite* Dataset**: A gold-standard dataset of software mentions in research publications (version 1.0)
http://doi.org/10.5281/zenodo.4445202
(Available at https://zenodo.org/record/4445202)
Software entity recognizer (*GROBID software-mentions module*):
https://github.com/ourresearch/software-mentions
Software knowledge base: https://github.com/kermitt2/software_kb

Reference (narrative documentation of the dataset):
Du, C., Cohoon, J., Lopez, P., & Howison, J. (2021). Softcite Dataset: A Dataset of Software Mentions in Biomedical and Economic Research Publications. *Journal of the Association for Information Science and Technology*. DOI: 10.1002/asi.24454

Howison Lab @ TEXAS science-miner TagWorks BERKELEY Institute for Data Science

How *Softcite* Helps?

—Achieve software entity recognition in research publications at scale

—Make visible software used in research and analysis workflow

—Provide a technical foundation for software identification, discovery, and retrieval

—Inform software decisions and use for research and analysis

How can *Softcite* Help Better?

—We hope to contribute to software visibility, discovery, and retrieval, including specialized search engine that identifies software:
https://citeas.org/

—We welcome suggestions, bug reports, or any sorts of contributions to *Softcite* and research software visibility ☺
https://github.com/howisonlab/softcite-dataset

Howison Lab @ TEXAS science-miner TagWorks BERKELEY Institute for Data Science

Thanks!

ALFRED P. SLOAN FOUNDATION
Howison Lab @ TEXAS
science-miner
TagWorks
BERKELEY Institute for Data Science