Generating simulation metadata with large language models

Infrastructure

J. Kambhampaty, O. Pinon Fischer, D. Mavris

Georgia Institute of Technology, Aerospace Systems Design Laboratory

SERC AI4SE & SE4AI Workshop 2025 | Presented by Jaya Kambhampaty

Aerospace Systems Design Laboratory

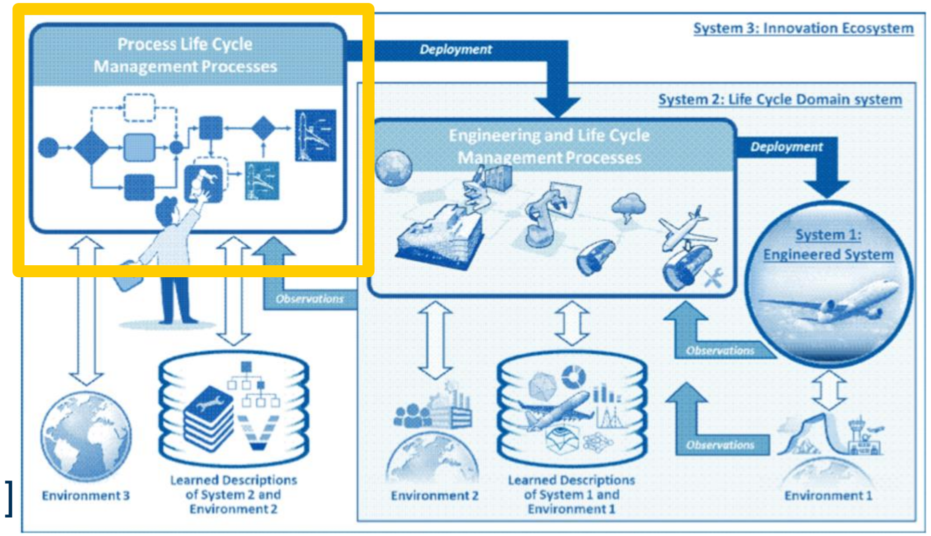
Context

- The design of aerospace systems involves the management and contextualization of large quantities of information
- This is especially relevant to problem spaces which hope to iterate quickly, but maintain consistency between stakeholder needs, past experience, and empirical observation [1]
- In design, much of this data is expressed in **natural language**
  - Requirements decomposed from stakeholder needs (RFP, etc.)

[1] Schindel (2022) "Realizing the value promise of digital engineering" https://incose.onlinelibrary.wiley.com/doi/epdf/10.1002/inst.12372

Context

- The design of aerospace systems involves the management and contextualization of large quantities of information
- This is especially relevant to problem spaces which hope to iterate quickly, but maintain consistency between stakeholder needs, past experience, and empirical observation [1]
- In design, much of this data is expressed in **natural language**
  - Requirements decomposed from stakeholder needs (RFP, etc.)


System 3: Innovation Ecosystem
Process Life Cycle Management Processes
Deployment
System 2: Life Cycle Domain system
Engineering and Life Cycle Management Processes
Deployment
System 1: Engineered System
Observations
Observations
Observations
Environment 3
Learned Descriptions of System 2 and Environment 2
Environment 2
Learned Descriptions of System 1 and Environment 1
Environment 1

[1]

What kind of tools might a next-generation **innovation ecosystem** have to **support our design and understanding** of complex systems?

What is the state of the art of **language understanding** for design?

1. Schindel (2022) “Realizing the value promise of digital engineering” https://incose.onlinelibrary.wiley.com/doi/epdf/10.1002/inst.12372

Modeling includes many natural language processing tasks

- Claim:

Modeling can be considered **a task of translating information** collected from many sources…	Literature Stakeholder needs assessments & requirements Documentation Regulations & incident reports Proposals
…**into a usable form** for analysis…	Models (incl. MBSE) Simulations Rulesets Calculations
…so that a **series of decisions** can be made…	Constraint analysis Requirements satisfaction Certification
…**to create a process or product** that achieves a desired goal	A new design

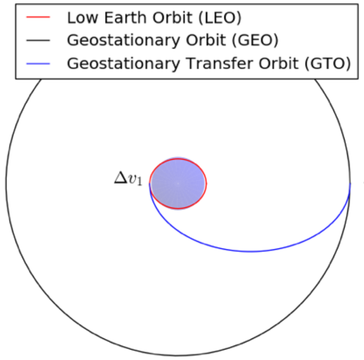
Modeling includes many natural language processing tasks

- Claim:

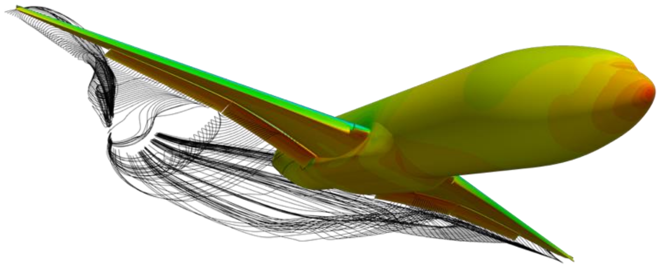
Modeling can be considered **a task of translating information** collected from many sources…	Literature Stakeholder needs assessments & requirements Documentation Regulations & incident reports Proposals
…**into a usable form** for analysis…	**Models** **Simulations** Rulesets Calculations
…so that a **series of decisions** can be made…	Constraint analysis Requirements satisfaction Certification
…**to create a process or product** that achieves a desired goal	A new design

Imagining the future of Model-Based

- Models and documents encapsulate engineering knowledge
  - “Engineering artifacts” = {“models”, “documents”}
  - They tell us, in various forms of data (text, geometries, images) about a system of interest
- Engineering artifacts can be expensive to produce
  - Value created by an engineering organization!
  - We’d like to continue to use the ones we have already when appropriate (“reuse”)
  - We’d like to build new ones faster (“generation”/”automation”)
    - Especially if they’re made up of many existing parts



[1]



[2]

1. OpenMDAO Docs https://openmdao.org/newdocs/versions/latest/examples/hohmann_transfer/hohmann_transfer.html?highlight=hohmann
2. SU2 https://su2code.github.io/

Imagining the future of Model-Based

- Once we can model physical systems with a high degree of realism, how do they support our engineering processes?
- Imagined future design process
  - Library of models
  - Modeler puts them together to represent a relevant system and environment
  - Decision-maker acts on the outputs of the model
- What questions remain?
  - Better understanding of engineering simulations that we use
  - Automatic sequencing/composition of relevant analytical pipelines
  - Modernized design reviews highly attentive to assumptions and analysis

Challenges in model-based workflows

- Extracting meaningful insights from old engineering artifacts is hard
  - Need to be able to understand many different ways of representing engineering knowledge
- Effective knowledge management is essential for understanding how, when, and why a model has been used in a simulation workflow
- MBSE shows us "effective knowledge management" is not free
  - translating models into standardized representations can have adoption challenges
  - Integration is a critical adoption issue; tools, models, and/or data repositories need to be linked in some way [1]
  - "Substantial effort upfront to set up a model-based environment" [1]
  - "Introduction of SysML in large organization is hampered by the sheer size of the language and the sometimes awkward user interface to modelling" [2]
  - Manual workflows do not scale, and require large amounts of upfront training

1. Henderson, Kaitlin, Thomas McDermott, and Alejandro Salado. "MBSE Adoption Experiences in Organizations: Lessons Learned." Systems Engineering 27, no. 1 (2024): 214–39. https://doi.org/10.1002/sys.21717.
2. Herzog, Erik, Jessica Hallonquist, and Johan Naeser. "4.5.1 Systems Modeling with SysML – an Experience Report." INCOSE International Symposium 22, no. 1 (2012): 600–611. https://doi.org/10.1002/j.2334-5837.2012.tb01359.x.

Research Objective

How do we automatically generate useful representations of engineering artifacts?

- What makes a useful representation of engineering artifacts?
  - What even is a representation of an engineering artifact?
  - Metadata is one…why is generating engineering metadata hard?

- How could we do this with large language models?
  - How good are models at doing metadata generation?
  - How do strong open-source language model options and test-time inference techniques influence performance on the metadata task?

- Today
  1. Metadata generation: building a benchmark problem set
  2. Generating metadata with LLMs: inference-time techniques for LLM performance
  3. Gaps and calls to action

Why are we trying to describe models?

- Need to know features of models we can use to make decisions
  - Interoperability:
    - How do I integrate multiple models quickly?
    - How do I reconcile models of subsystems which are semantically in conflict?
    - How do I reconcile models across lifecycle phases?
  - Traceability:
    - How do I trace model-driven analysis to downstream decision-making?
- Plumbing behind the scenes of multi-step automatic computational workflows
  - We're proposing digital engineered ecosystems with high-fidelity high-data replicas of existing environments
  - Increased emphasis on MBSE or systems engineering environments
  - Hard to trust complex modeling ecosystems without it!

What is metadata? (Berners-Lee)

- “Metadata is machine understandable information about web resources or other things” [1]
- “Information which software agents can use in order to make [1] :

  - Life easier for us
  - Ensure we obey our principles, the law
  - Check that we can trust what we are doing
  - Make everything work more smoothly and rapidly” [1]

1. Berners-Lee, T. (1997) “Metadata Architecture” https://www.w3.org/DesignIssues/Metadata.html

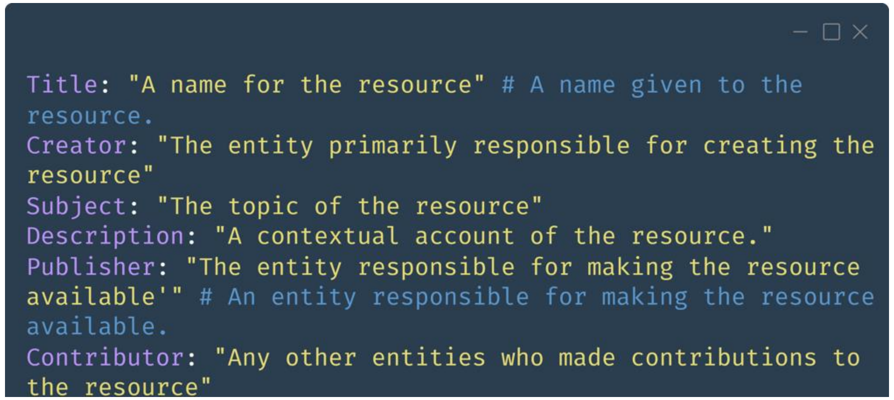
What is metadata?

Dublin Core Metadata Initiative [1]

- Focuses on title, contextual information
- 15 features

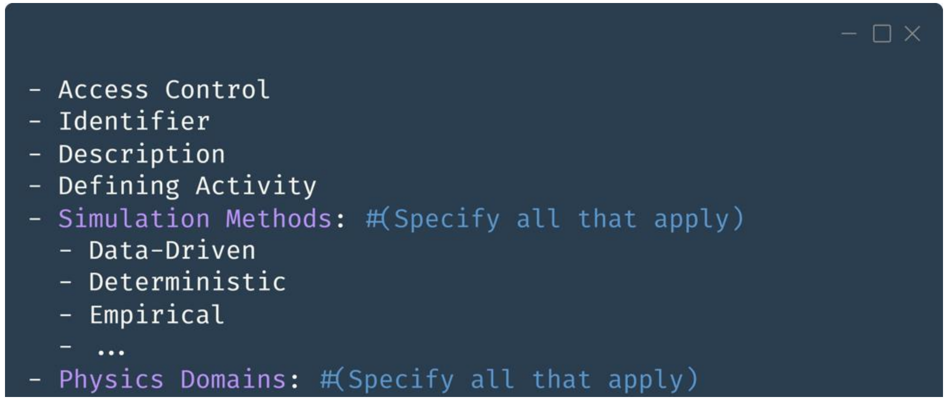
Engineering Simulation Metadata Specification [2]

- ASSESS Initiative from NAFEMS
- Engineering focused
- 644 features (in a hierarchy)



```
Title: "A name for the resource" # A name given to the
resource.
Creator: "The entity primarily responsible for creating the
resource"
Subject: "The topic of the resource"
Description: "A contextual account of the resource."
Publisher: "The entity responsible for making the resource
available'" # An entity responsible for making the resource
available.
Contributor: "Any other entities who made contributions to
the resource"
```

Dublin Core excerpted features [1]



```
- Access Control
- Identifier
- Description
- Defining Activity
- Simulation Methods: #(Specify all that apply)
  - Data-Driven
  - Deterministic
  - Empirical
  - …
- Physics Domains: #(Specify all that apply)
```

ESMS excerpted features
Model representation feature group [2]

1. Dublin Core Metadata Initiative. (2020) DCMI Metadata Terms. https://www.dublincore.org/specifications/dublin-core/dcmi-terms/
2. NAFEMS ASSESS Initiative. Engineering Simulation Metadata Specification, Feb. 2024.

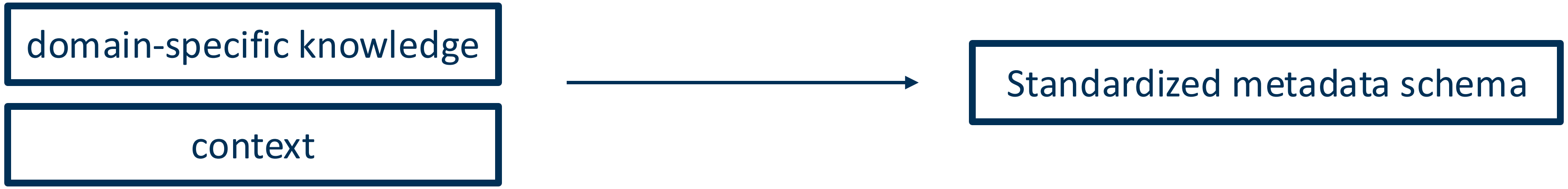
What is *engineering* metadata?

- “Metadata is machine understandable information about web resources or other things” [1]
- “Information which software agents can use in order to make [1] :
  - Life easier for us
  - Ensure we obey our principles, the law
  - Check that we can trust what we are doing
  - Make everything work more smoothly and rapidly” [1]
- Projects an engineering artifact into a set of features
  - May be useful to both human agents and software agents
  - Author is a feature, “Tim Berners-Lee” is its value
  - These features have a few important properties
    - human-interpretable
    - lower dimension than the full engineering artifact

[1] Berners-Lee, T. (1997) “Metadata Architecture” https://www.w3.org/DesignIssues/Metadata.html

Why is generating metadata hard?

- Translate



domain-specific knowledge

context

Standardized metadata schema

Knowledge Intensive

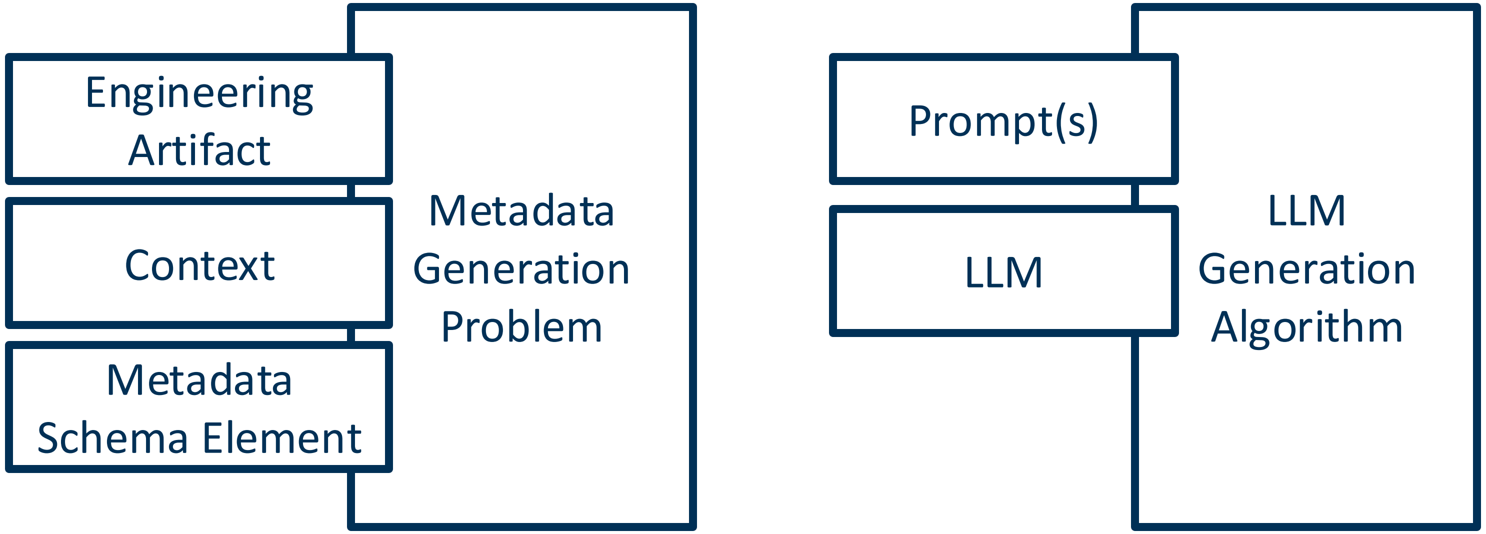
- large amount of engineering knowledge about the simulation being described
- additional contextual understanding beyond the simulation to describe them with respect to:
  - their use (computational environment),
  - the part of the world they model (simulated system's environment)

Labor Intensive

- To wit, filling out large forms is hard
- Disparate features of models may all be included in a single schema
- This translation can be arduous or dull, such that humans fatigue quickly.
- e.g. ESMS is a large schema which organizes 644 model attributes into a hierarchy

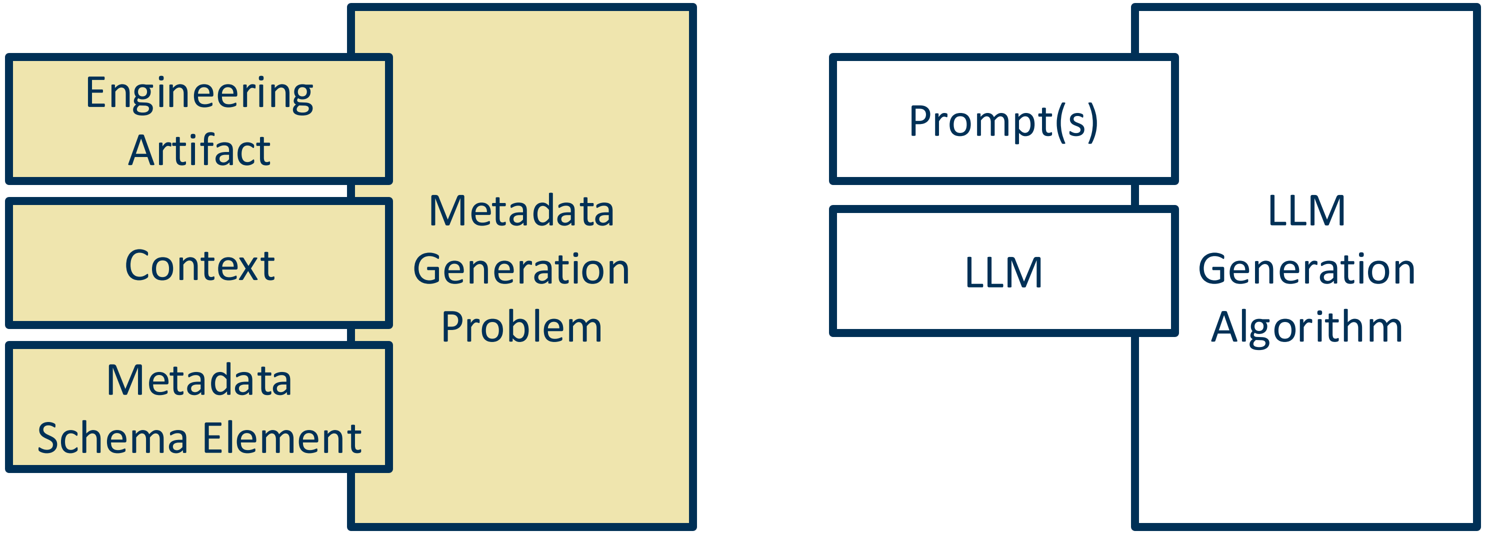
MetaGator: a metadata aggregator

- Solving simulation metadata generation problems with LLMs



MetaGator: a metadata aggregator

- Solving simulation metadata generation problems with LLMs



Engineering Artifact

Context

Metadata Schema Element

Metadata Generation Problem

Prompt(s)

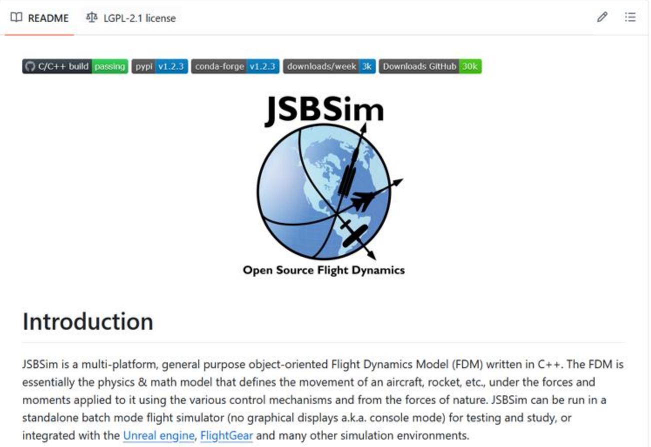
LLM

LLM Generation Algorithm

Defining the metadata generation problem

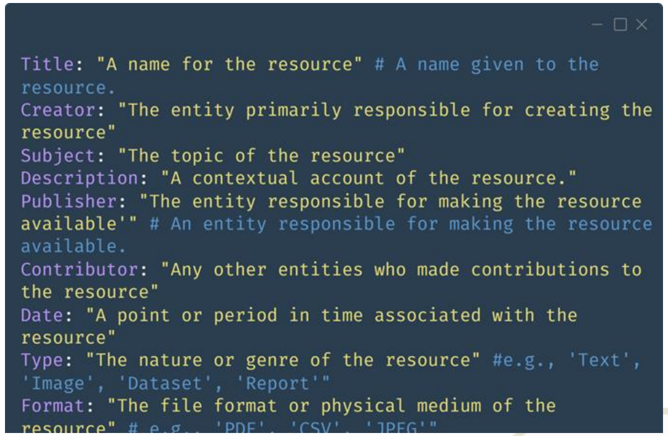
- The metadata generation problem takes as input
    - An engineering artifact (one or more, potentially structured)
    - Context (other artifacts known to be relevant to that artifact)
    - A metadata schema element (a specific attribute of a metadata schema which should be populated with a value)

- A solution to a metadata generation problem is an assignment of some **value** to the **metadata schema element**
    - **Author: Tim Berners-Lee**



Introduction

JSBSim is a multi-platform, general purpose object-oriented Flight Dynamics Model (FDM) written in C++. The FDM is essentially the physics & math model that defines the movement of an aircraft, rocket, etc., under the forces and moments applied to it using the various control mechanisms and from the forces of nature. JSBSim can be run in a standalone batch mode flight simulator (no graphical displays a.k.a. console mode) for testing and study, or integrated with the Unreal engine, FlightGear and many other simulation environments.



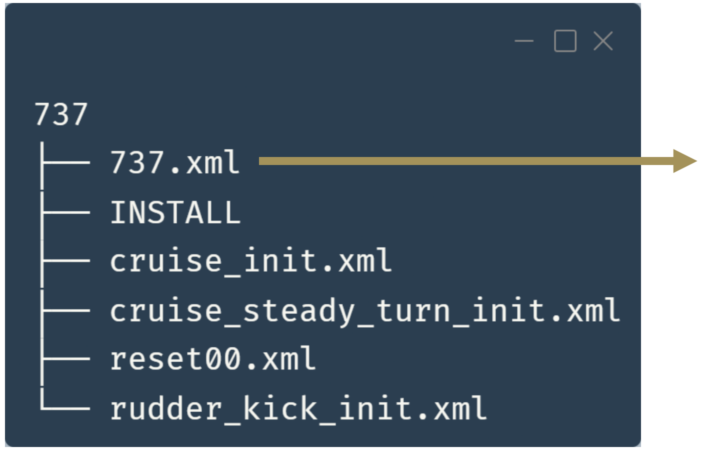
```
Title: "A name for the resource" # A name given to the
resource.
Creator: "The entity primarily responsible for creating the
resource"
Subject: "The topic of the resource"
Description: "A contextual account of the resource."
Publisher: "The entity responsible for making the resource
available'" # An entity responsible for making the resource
available.
Contributor: "Any other entities who made contributions to
the resource"
Date: "A point or period in time associated with the
resource"
Type: "The nature or genre of the resource" #e.g., 'Text',
'Image', 'Dataset', 'Report'"
Format: "The file format or physical medium of the
resource" # e.g., 'PDF', 'CSV', 'JPEG'"
```

Features of metadata generation problems

- (**IN**) An engineering artifact (one or more, potentially structured)
  - Vary in knowledge domain
  - Vary in format (different modeling languages or document structures)
  - What is useful as a “model” may be made up of multiple files or a large amount of input content
- (**IN**) Context (other artifacts known to be relevant to that artifact)
  - Input sets range from a single model-document pair to larger collections
  - These may also be inaccurate; engineering repositories can be messy
  - Quality of the artifact itself, depending on who developed the model, the context could be good/bad and applicable/not applicable to the problem you’re trying to solve
- (**OUT**) Key, value pairs of Label: Value for some labels
  - *Implicitly* that means we need the metadata schema
  - A value to be assigned to a metadata schema element a specific attribute of the metadata schema

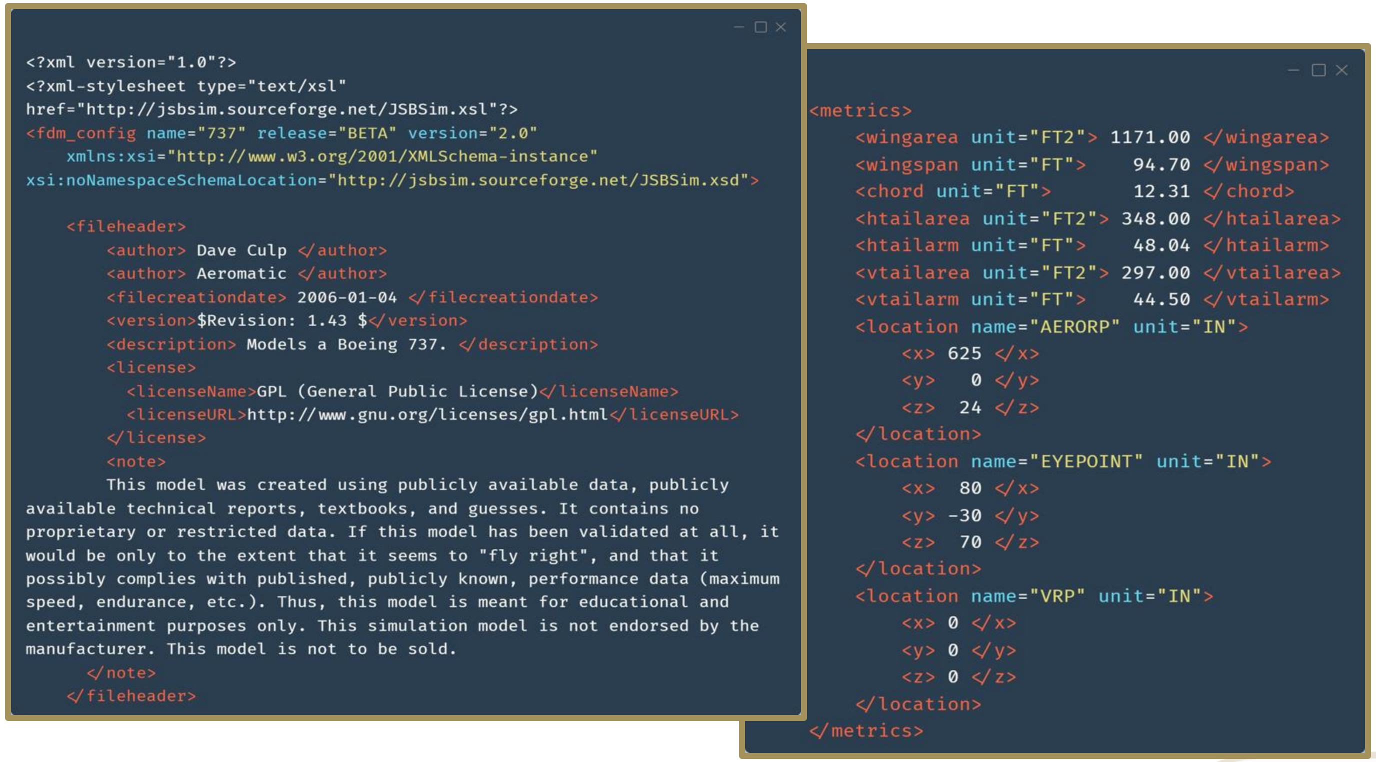
Building a problem set of metadata generation problems

- **Need**: A dataset of solved examples of the metadata generation task that reflect the real-world variation in the problem
- **Issue**: existing repositories of engineering artifacts vary wildly



```
737
├── 737.xml
├── INSTALL
├── cruise_init.xml
├── cruise_steady_turn_init.xml
├── reset00.xml
└── rudder_kick_init.xml
```

context



```
<?xml version="1.0"?>
<?xml-stylesheet type="text/xsl"
href="http://jsbsim.sourceforge.net/JSBSim.xsl"?>
<fdm_config name="737" release="BETA" version="2.0"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="http://jsbsim.sourceforge.net/JSBSim.xsd">

    <fileheader>
        <author> Dave Culp </author>
        <author> Aeromatic </author>
        <filecreationdate> 2006-01-04 </filecreationdate>
        <version>$Revision: 1.43 $</version>
        <description> Models a Boeing 737. </description>
        <license>
          <licenseName>GPL (General Public License)</licenseName>
          <licenseURL>http://www.gnu.org/licenses/gpl.html</licenseURL>
        </license>
        <note>
        This model was created using publicly available data, publicly
available technical reports, textbooks, and guesses. It contains no
proprietary or restricted data. If this model has been validated at all, it
would be only to the extent that it seems to "fly right", and that it
possibly complies with published, publicly known, performance data (maximum
speed, endurance, etc.). Thus, this model is meant for educational and
entertainment purposes only. This simulation model is not endorsed by the
manufacturer. This model is not to be sold.
      </note>
    </fileheader>
```

```
<metrics>
    <wingarea unit="FT2"> 1171.00 </wingarea>
    <wingspan unit="FT">    94.70 </wingspan>
    <chord unit="FT">       12.31 </chord>
    <htailarea unit="FT2"> 348.00 </htailarea>
    <htailarm unit="FT">    48.04 </htailarm>
    <vtailarea unit="FT2"> 297.00 </vtailarea>
    <vtailarm unit="FT">    44.50 </vtailarm>
    <location name="AERORP" unit="IN">
        <x> 625 </x>
        <y>   0 </y>
        <z>  24 </z>
    </location>
    <location name="EYEPOINT" unit="IN">
        <x>  80 </x>
        <y> -30 </y>
        <z>  70 </z>
    </location>
    <location name="VRP" unit="IN">
        <x> 0 </x>
        <y> 0 </y>
        <z> 0 </z>
    </location>
</metrics>
```

Engineering artifact

Building a problem set of metadata generation problems

- **Need**: A dataset of solved examples of the metadata generation task that reflect the real-world variation in the problem
- **Issue**: existing repositories of engineering artifacts vary wildly
- Produce a variety of challenges arise in various ways
  - Access
    - E.g. simulation has binary files we can't read, access control
  - Knowledge
    - E.g. the modeling technique or subject of the model is highly technical
  - Context
    - E.g. the model is in a large repo, that repo has poor documentation
  - Interpretability
    - E.g. easier:python scripts vs. harder:airfoil coordinates

The problems in a comprehensive "benchmark" on this task should cover all of these.

Where does complexity come from?

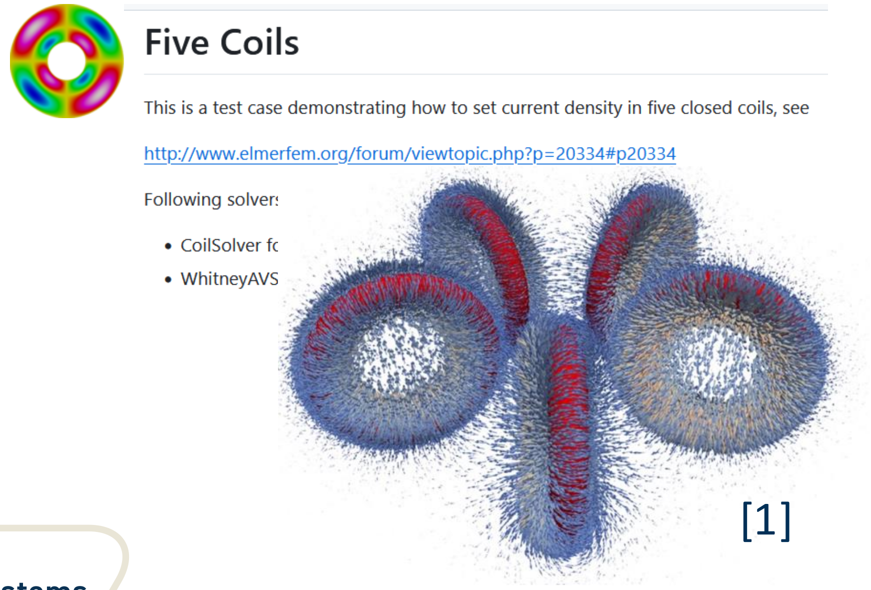
Structure	Readability	Access
One folder, all files in it	Text files	Open source, internet
Embedded in repository	Coordinates, mesh	Controlled tools
Multimodel repo	Binary files	Controlled models
	Corrupted, missing	

*Increasing complexity**

**Of some kind. It is a hypothesis that these problem features are the ones which make the problem harder for LLMs.*

Sourcing data for example problems

- Sourcing data for a proof-of-concept
  - Test cases and tutorials for open-source engineering tools
  - Fairly clean relative to an arbitrary engineering repo
  - Emphasis is **infrastructure development**



Five Coils

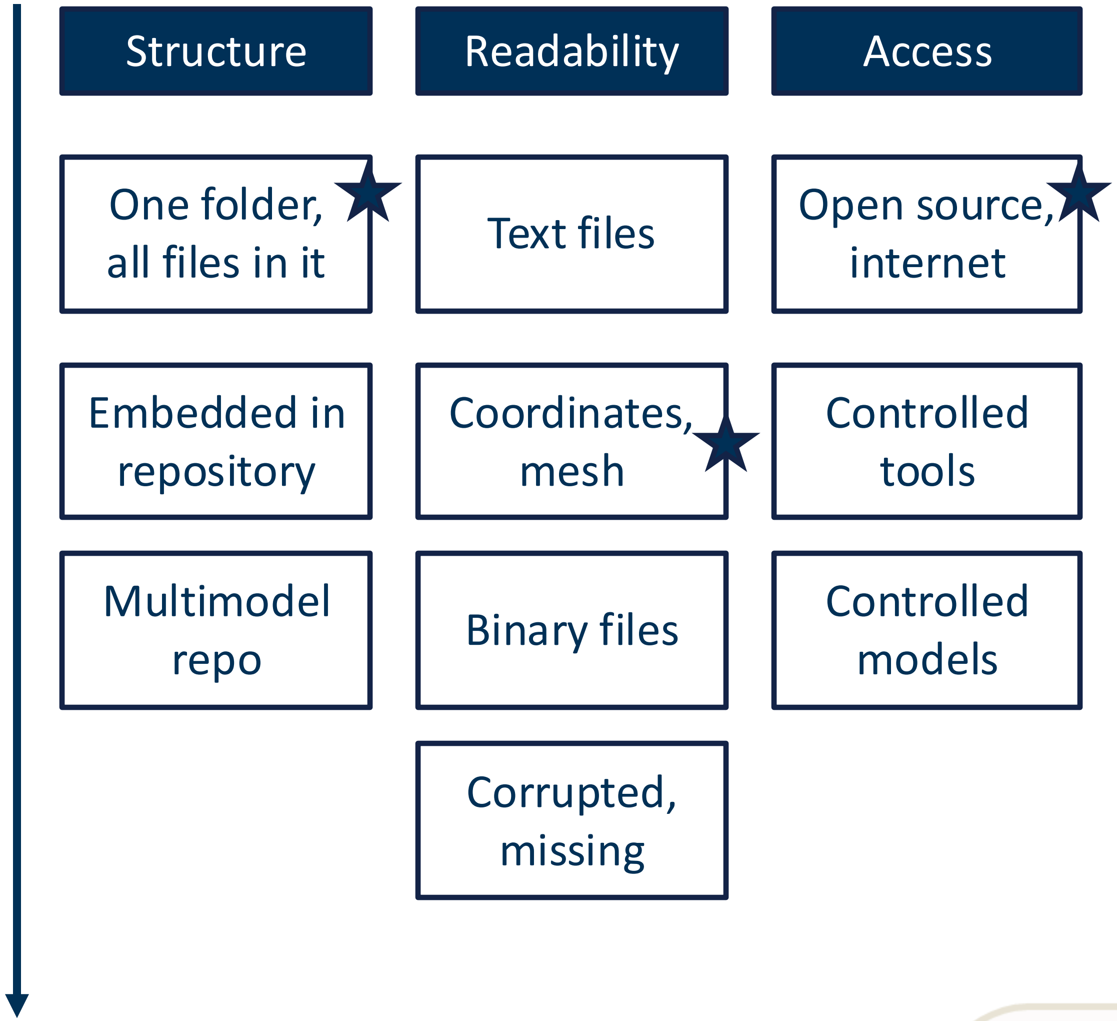
This is a test case demonstrating how to set current density in five closed coils, see

http://www.elmerfem.org/forum/viewtopic.php?p=20334#p20334

Following solver

- CoilSolver fc
- WhitneyAVS

[1]



Structure	Readability	Access
One folder, all files in it ★	Text files	Open source, internet ★
Embedded in repository	Coordinates, mesh ★	Controlled tools
Multimodel repo	Binary files	Controlled models
	Corrupted, missing	

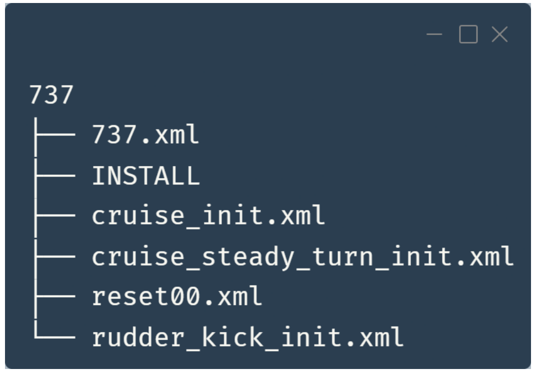
[1] ElmerCSC "elmer-elmag" GitHub. https://github.com/ElmerCSC/elmer-elmag/tree/main/FiveCoils

Dataset Sources

- JBSIM
- Dynamics library
- XML
- Multi-file, some data inputs

JSBSim

Open Source Flight Dynamics

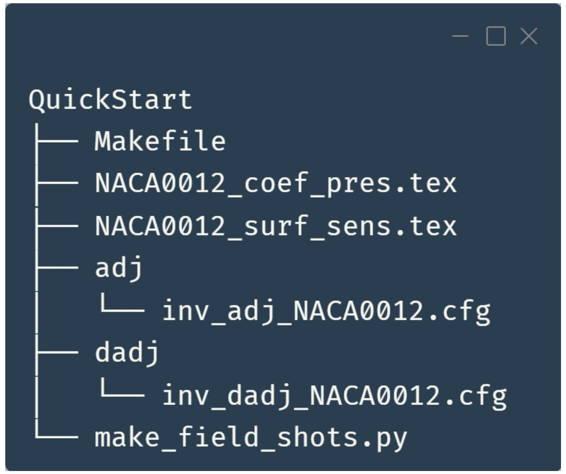


```
737
├── 737.xml
├── INSTALL
├── cruise_init.xml
├── cruise_steady_turn_init.xml
├── reset00.xml
└── rudder_kick_init.xml
```

- SU2
- Multiphysics library
- Includes several mesh files
- Simple 2-file quickstart example

SU2

code

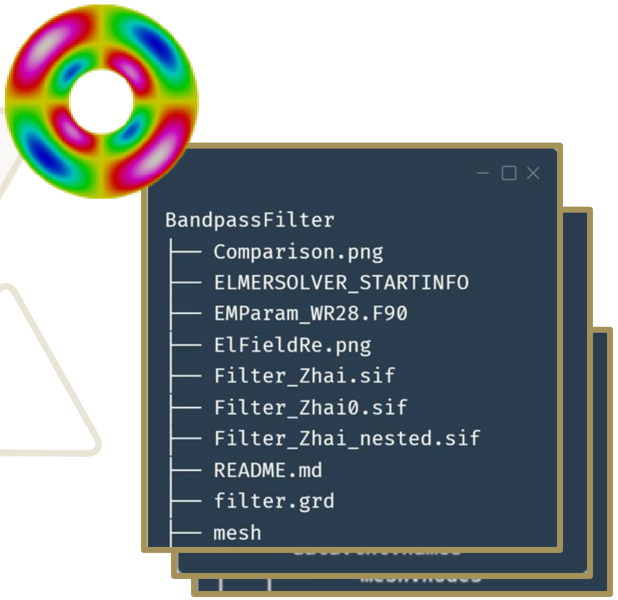


```
QuickStart
├── Makefile
├── NACA0012_coef_pres.tex
├── NACA0012_surf_sens.tex
├── adj
│   └── inv_adj_NACA0012.cfg
├── dadj
│   └── inv_dadj_NACA0012.cfg
└── make_field_shots.py
```

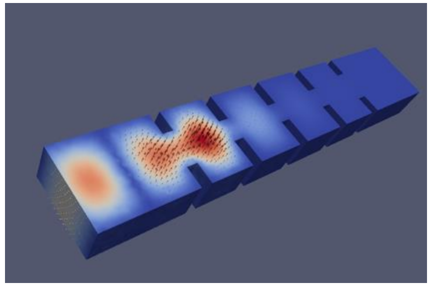
1. JSBSim-Team "jsbsim" https://github.com/JSBSim-Team/jsbsim
2. SU2 https://su2code.github.io/
3. Su2code SU2 https://github.com/su2code

Dataset Sources

- ELMER
- Multiphysics library
- Includes several mesh files
- Multi-file, includes data

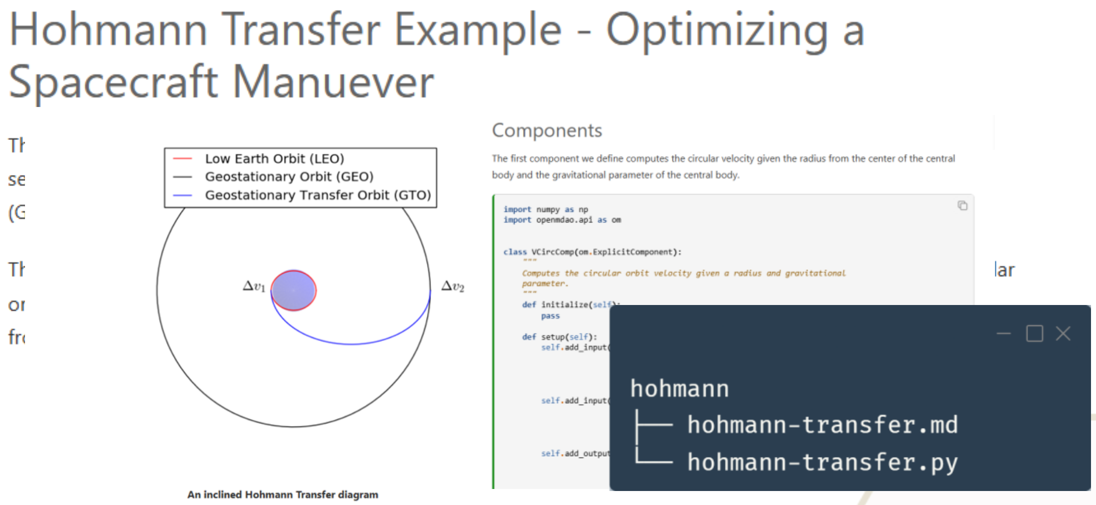


```
BandpassFilter
├── Comparison.png
├── ELMERSOLVER_STARTINFO
├── EMParam_WR28.F90
├── ElFieldRe.png
├── Filter_Zhai.sif
├── Filter_Zhai0.sif
├── Filter_Zhai_nested.sif
├── README.md
├── filter.grd
├── mesh
```



- OpenMDAO
- Optimization framework for coupled multidisciplinary problems
- Tutorial split into python and markdown docs

openMDAO



Hohmann Transfer Example - Optimizing a Spacecraft Manuever

Th
se
(G

Th
or
fr

An inclined Hohmann Transfer diagram

Components

The first component we define computes the circular velocity given the radius from the center of the central body and the gravitational parameter of the central body.

```
import numpy as np
import openmdao.api as om


class VCircComp(om.ExplicitComponent):
    """
    Computes the circular orbit velocity given a radius and gravitational
    parameter.
    """
    def initialize(self
        pass

    def setup(self):
        self.add_input(



        self.add_input(



        self.add_output
```

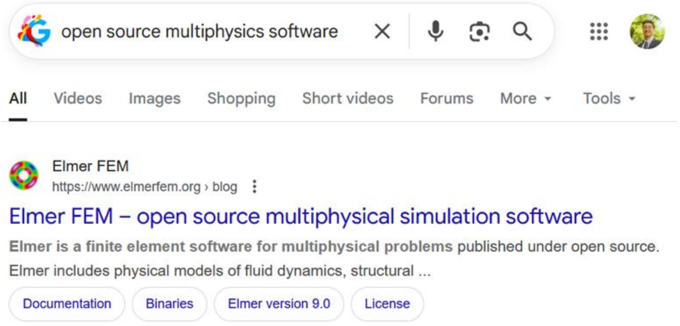
lar

```
hohmann
├── hohmann-transfer.md
└── hohmann-transfer.py
```

1. Elmer FEM https://www.elmerfem.org/blog/
2. ElmerCSC "elmer-elmag" GitHub. https://github.com/ElmerCSC/elmer-elmag/tree/main/FiveCoils
3. OpenMDAO Docs https://openmdao.org/newdocs/versions/latest/examples/hohmann_transfer/hohmann_transfer.html?highlight=hohmann
4. OpenMDAO "OpenMDAO" https://github.com/openmdao/openmdao

Our problem set, today

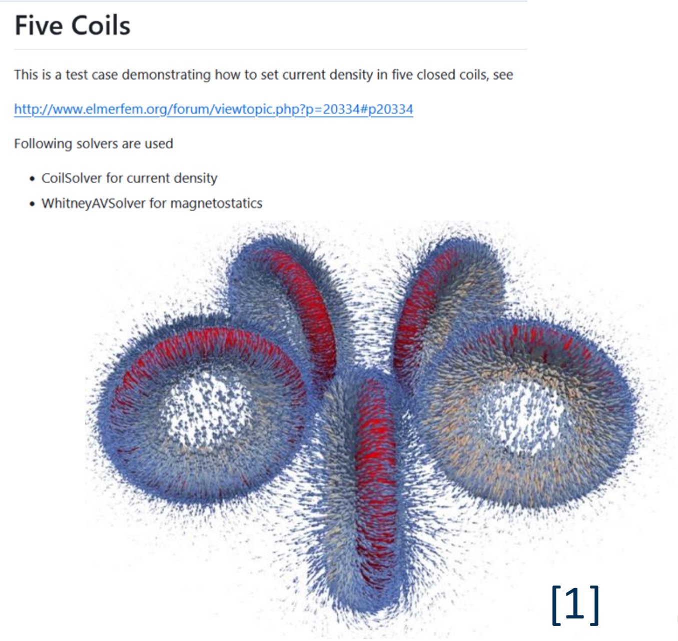
- Where does good source data to build these problems exist?
  - Test cases and tutorials for open-source engineering repositories
  - *Today*: We build a limited set of examples by grabbing tutorials from SU2, Elmer, JSBSim, and OpenMDAO
  - *Future*: All tutorials from these tools, and additional tools
- Threats to success
  - The stakeholders with the best test cases and most accurate past engineering repositories won’t/can’t share
  - Have to solve this: automated knowledge management is a major force multiplier to the whole technical ecosystem
  - Open benchmarks push the field forward
- Future vision
  - More models, more tools, more model forms
  - Repo structure on top of model content, representation



Elmer FEM
https://www.elmerfem.org › blog

Elmer FEM – open source multiphysical simulation software

Elmer is a finite element software for multiphysical problems published under open source. Elmer includes physical models of fluid dynamics, structural ...



Five Coils

This is a test case demonstrating how to set current density in five closed coils, see http://www.elmerfem.org/forum/viewtopic.php?p=20334#p20334

Following solvers are used

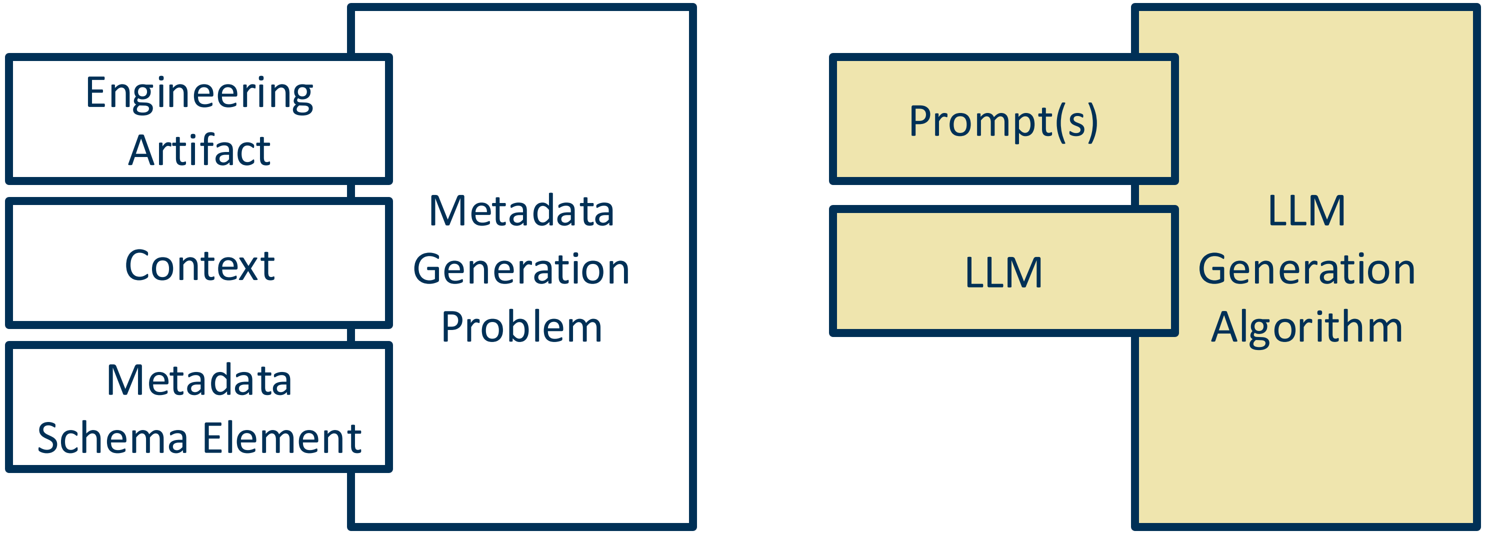
- CoilSolver for current density
- WhitneyAVSolver for magnetostatics

[1]

1. ElmerCSC “elmer-elmag” GitHub. https://github.com/ElmerCSC/elmer-elmag/tree/main/FiveCoils

MetaGator: a metadata aggregator

- Solving simulation metadata generation problems with LLMs



Engineering Artifact

Context

Metadata Schema Element

Metadata Generation Problem

Prompt(s)

LLM

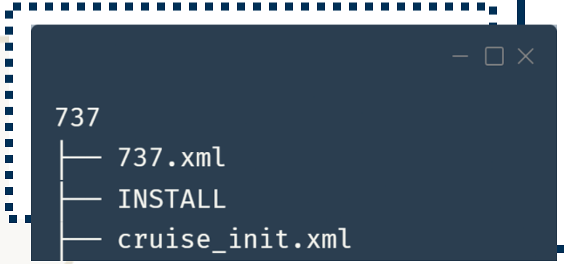
LLM Generation Algorithm

Language model-driven artificial intelligence

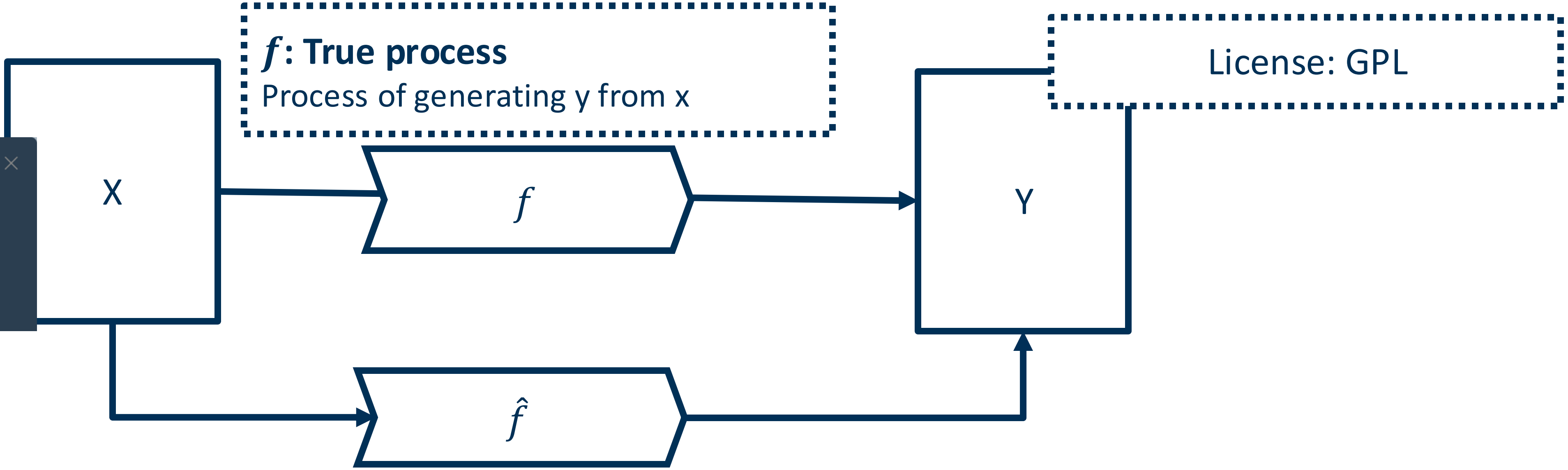
- Language models have been shown to provide useful performance on a wide variety of language-based tasks including:
  - Code
    - HumanEval: generate python programs from docstrings [1]
    - SWEBench: generate resolutions to issues 12 open-source Python-repositories [2, 3]
  - Question Answering
    - SQUAD: reading comprehension on Wikipedia articles [4]
    - GPQA Diamond: google-proof PhD-level technical question-answering [5]
  - Schema Usage/Generation
    - SchemaBench (Feb 2025): generating structurally correct, schema-compliant JSON
    - MCP-Bench (Aug 2025): using tools in a wide variety of contexts
  - Progress in agentic software engineering tools may be relevant
    - hard to systematically test since agentic workflow wraps an LLM in closed-source products like Cursor, Lovable, Claude Code, Gemini CLI, etc.
- But our understanding of their performance on SE domain-specific, and problems of engineering simulation context remains limited
  - Benchmarks do not currently reflect systems engineering use cases

1. Chen et al. (2021) "Evaluating Large Language Models Trained on Code" arXiv preprint: arXiv:2107.03374
2. Jimenez et al. (2024). "SWE-bench: Can Language Models Resolve Real-World GitHub Issues?" arXiv preprint arXiv:2310.06770.
3. Chowdhury et al. (2024) Introducing SWE-Bench Verified from OpenAI
4. Rajpurkar et al. (2016) "SQuAD: 100,000+ Questions for Machine Comprehension of Text" arXiv preprint: arXiv:1606.05250
5. Rein et al. (2023) "GPQA: a graduate-level google-proof Q&A benchmark" arXiv preprint: arXiv:2311.12022
6. Lu et al. (02.2025) Learning to Generate Structured output with Schema Reinforcement Learning arXiv preprint: arXiv: 2502.18878
7. Wang et al. (08.2025) MCP-Bench: Benchmarking Tool-Using LLM Agents with Complex Real-World Tasks via MCP Servers": arXiv preprint: arXiv:2508.20453

The language model: a general surrogate of language tasks



```
737
├── 737.xml
├── INSTALL
├── cruise_init.xml
```



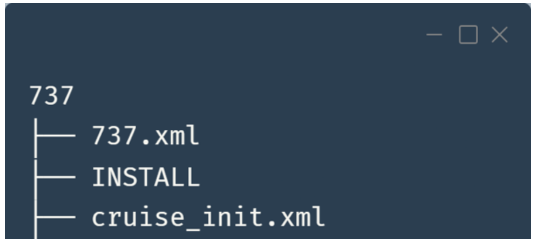
***f*: True process**
Process of generating y from x

***f̂*: Surrogate of true process**
Some computational model of the true process
Neural network language models happen to be:

- unreasonably good at approximating ***f***
- across a wide variety of ***f***

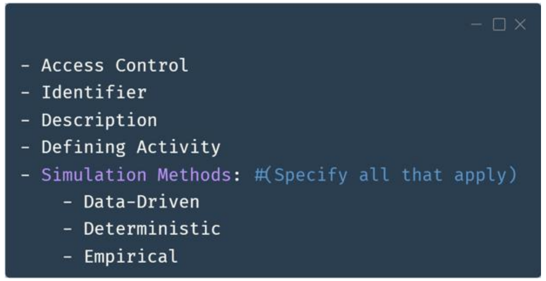
Prediction quality is input prompt and model weights

Input (X)

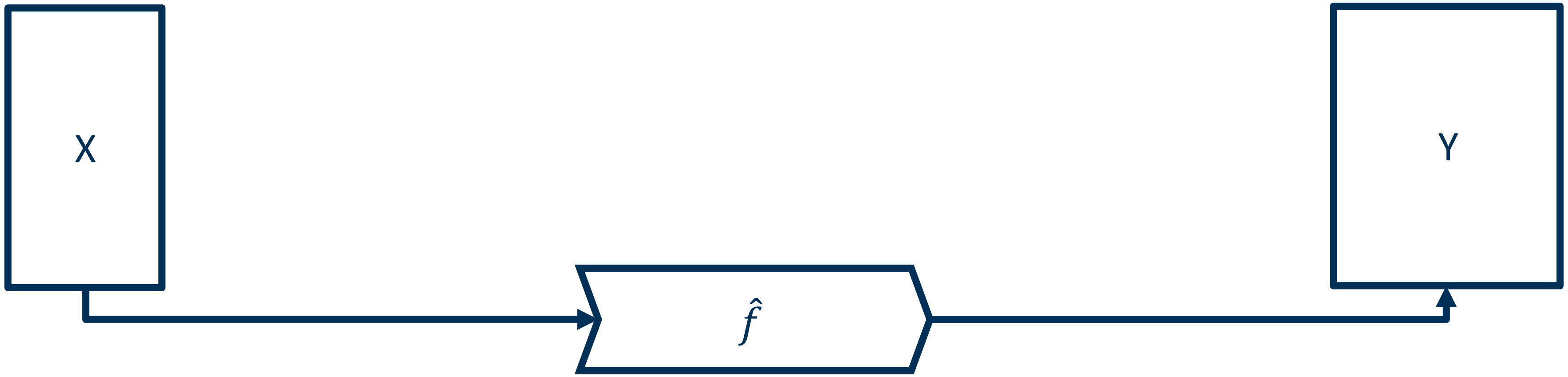


```
737
├── 737.xml
├── INSTALL
├── cruise_init.xml
```

Output (Y)



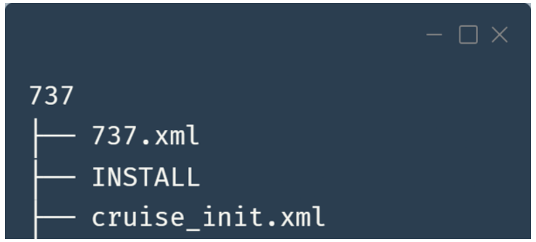
```
- Access Control
- Identifier
- Description
- Defining Activity
- Simulation Methods: #(Specify all that apply)
    - Data-Driven
    - Deterministic
    - Empirical
```



How do we change LLM prediction performance?

Prediction quality is input prompt and model weights

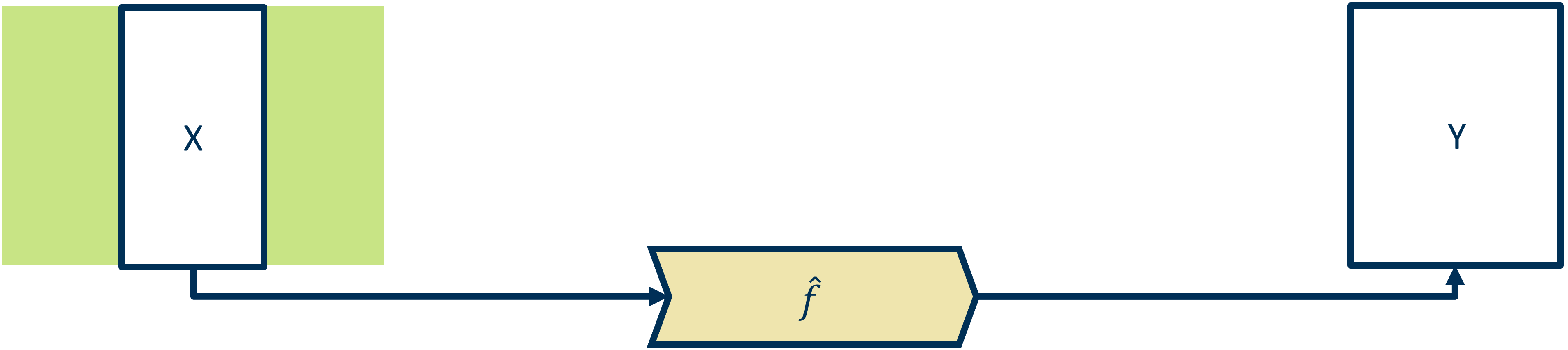
Input (X)



```
737
├── 737.xml
├── INSTALL
├── cruise_init.xml
```

Output (Y)

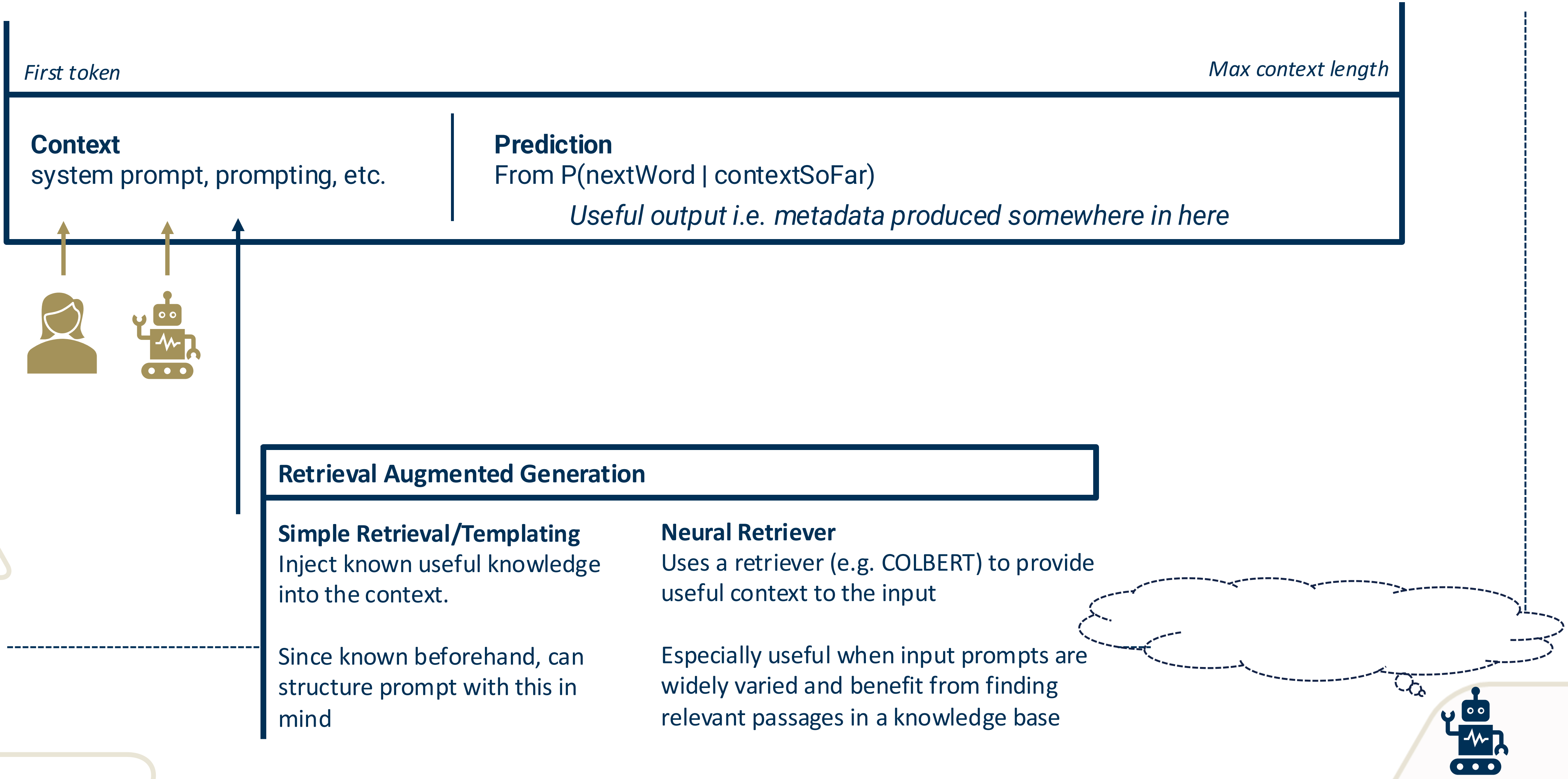
Training (finetuning, etc.)
change the params of $\hat{f}$



Prompting
change the input to $\hat{f}$

Training (finetuning, etc.)
change the params of $\hat{f}$

Generating useful text with large language models



First token

Max context length

Context
system prompt, prompting, etc.

Prediction
From P(nextWord | contextSoFar)

Useful output i.e. metadata produced somewhere in here

Retrieval Augmented Generation

Simple Retrieval/Templating
Inject known useful knowledge into the context.

Since known beforehand, can structure prompt with this in mind

Neural Retriever
Uses a retriever (e.g. COLBERT) to provide useful context to the input

Especially useful when input prompts are widely varied and benefit from finding relevant passages in a knowledge base

Ideal properties of a metadata generation system

Who owns the models?

Model providers

Open Source

- Flexible, cheap to orchestrate
  - Avoid model-lock
  - Enables on-prem

How large?

New Algorithms

Model Size

- Fast, cheap to run
  - Runs quickly
  - On consumer-grade hardware

How to use them?

Additional training

Inference time techniques

- Scales well to other problems
  - Not locked to a knowledge base
  - Different metadata
  - Different problem sets

Ideal properties of a metadata generation system

Who owns the models?

Model providers

Open Source

- Flexible, cheap to orchestrate
  - Avoid model-lock
  - Enables on-prem

How large?

New Algorithms

Model Size

- Fast, cheap to run
  - Runs quickly
  - On consumer-grade hardware

How to use them?

Additional training

Inference time techniques

- Scales well to other problems
  - Not locked to a knowledge base
  - Different metadata
  - Different problem sets

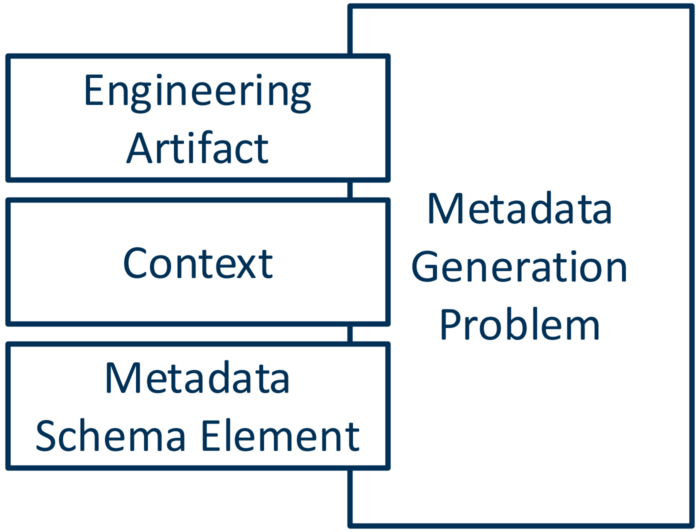
Morphology of the MetaGator tool

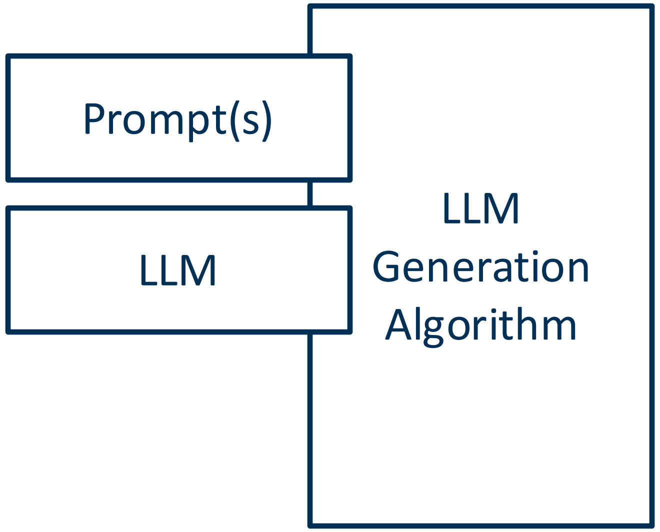
Architectural Feature		Alternatives
Algorithm	Prompting	Zero-Shot, Chain-of-Thought, s1
	Search	Self-Refine, Evolutionary prompting
Model	Qwen-2.5-Instruct	0.6B, 1.7B, 4B, 7B, 14B
	Gemma-3	3B
	Phi-4-mini-instruct	3.8B
Retrieval		None ESMS Definitions

MetaGator Morphological Matrix. Model names are attached to specific sizes and cannot be combined with model size exhaustively, i.e .there is no 3B parameter variant of Qwen2.5-Instruct.

MetaGator: a metadata aggregator

- Solving simulation metadata generation problems with LLMs



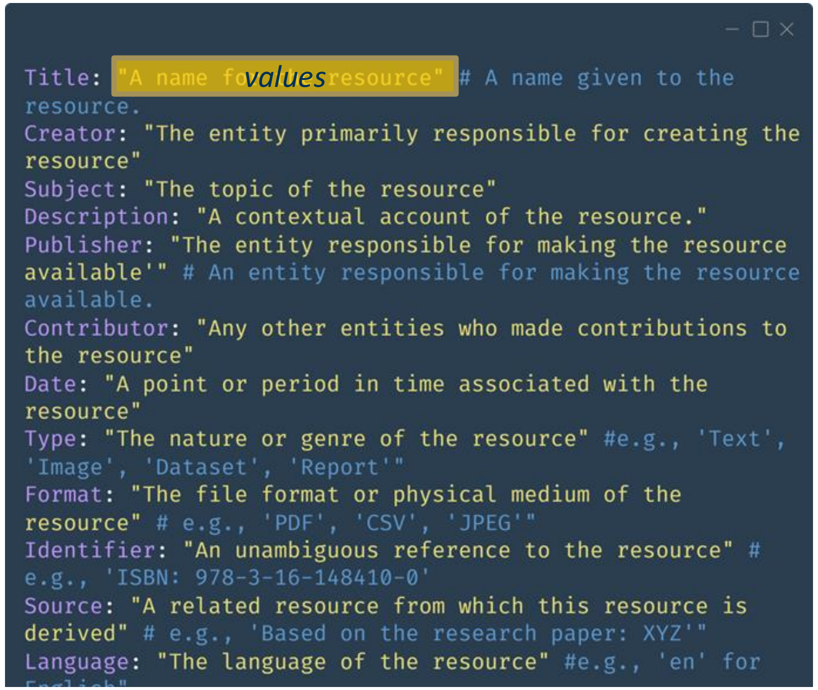


Generating useful text with large language models

- Two key goals:
  - characterize the performance of current open source LLMs on the metadata generation task
  - figure out how to generally improve that performance
- We’d like to know:
  - If there’s a relationship with model size
  - If prompting improves performance on this task
- Our dataset is small; we present some observational case-based results
- Today:
  - Talk about how to score model performance
  - Examine what the plots would look like

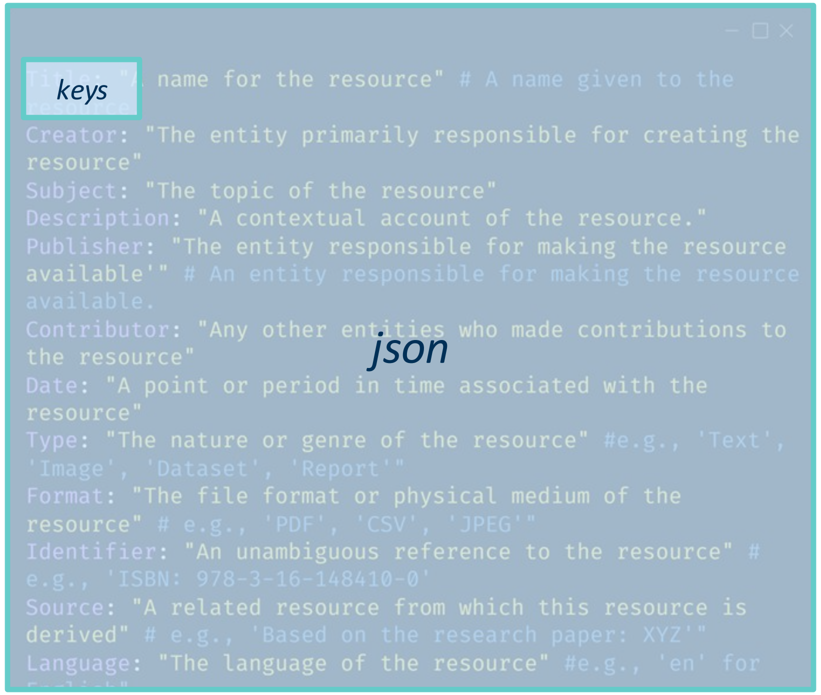
Format matters: making LLM outputs scoreable

- Metrics we care about depends on problem scope
  - How much do we want the model to do?
  - Atomicity: labeling, labeling + formatting,



```
Title: "A name for the resource" # A name given to the
resource.
Creator: "The entity primarily responsible for creating the
resource"
Subject: "The topic of the resource"
Description: "A contextual account of the resource."
Publisher: "The entity responsible for making the resource
available'" # An entity responsible for making the resource
available.
Contributor: "Any other entities who made contributions to
the resource"
Date: "A point or period in time associated with the
resource"
Type: "The nature or genre of the resource" #e.g., 'Text',
'Image', 'Dataset', 'Report'"
Format: "The file format or physical medium of the
resource" # e.g., 'PDF', 'CSV', 'JPEG'"
Identifier: "An unambiguous reference to the resource" #
e.g., 'ISBN: 978-3-16-148410-0'
Source: "A related resource from which this resource is
derived" # e.g., 'Based on the research paper: XYZ'"
Language: "The language of the resource" #e.g., 'en' for
English"
```

labeling



```
Title: "A name for the resource" # A name given to the
resource.
Creator: "The entity primarily responsible for creating the
resource"
Subject: "The topic of the resource"
Description: "A contextual account of the resource."
Publisher: "The entity responsible for making the resource
available'" # An entity responsible for making the resource
available.
Contributor: "Any other entities who made contributions to
the resource"
Date: "A point or period in time associated with the
resource"
Type: "The nature or genre of the resource" #e.g., 'Text',
'Image', 'Dataset', 'Report'"
Format: "The file format or physical medium of the
resource" # e.g., 'PDF', 'CSV', 'JPEG'"
Identifier: "An unambiguous reference to the resource" #
e.g., 'ISBN: 978-3-16-148410-0'
Source: "A related resource from which this resource is
derived" # e.g., 'Based on the research paper: XYZ'"
Language: "The language of the resource" #e.g., 'en' for
English"
```

formatting

Measuring LLM Performance

DCMI: Natural language generation

X	Y_true	Y_pred
<engineering artifact>	*“WR28 Waveguide Bandpass Filter Simulation”*	*“BandpassFilter”*

ESMS: Multi-label classification

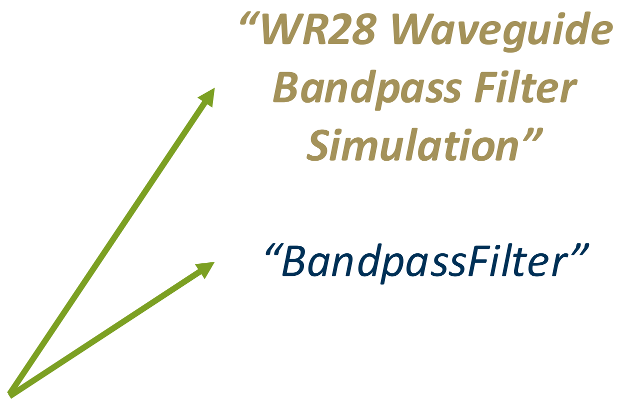
X	Y_true	Y_pred
<engineering artifact>	[“Electromagnetics (CEM) - high frequency”, “Multiphysics”]	[“Electromagnetics (CEM) - high frequency”, “Control”]

Measuring LLM performance

DCMI: Natural language generation

Semantic Similarity

- Cosine similarity of embedded tags
- Works for any sequence of text
- Want to see if we are getting close



ESMS: Multi-label classification

Macro-averaged F1

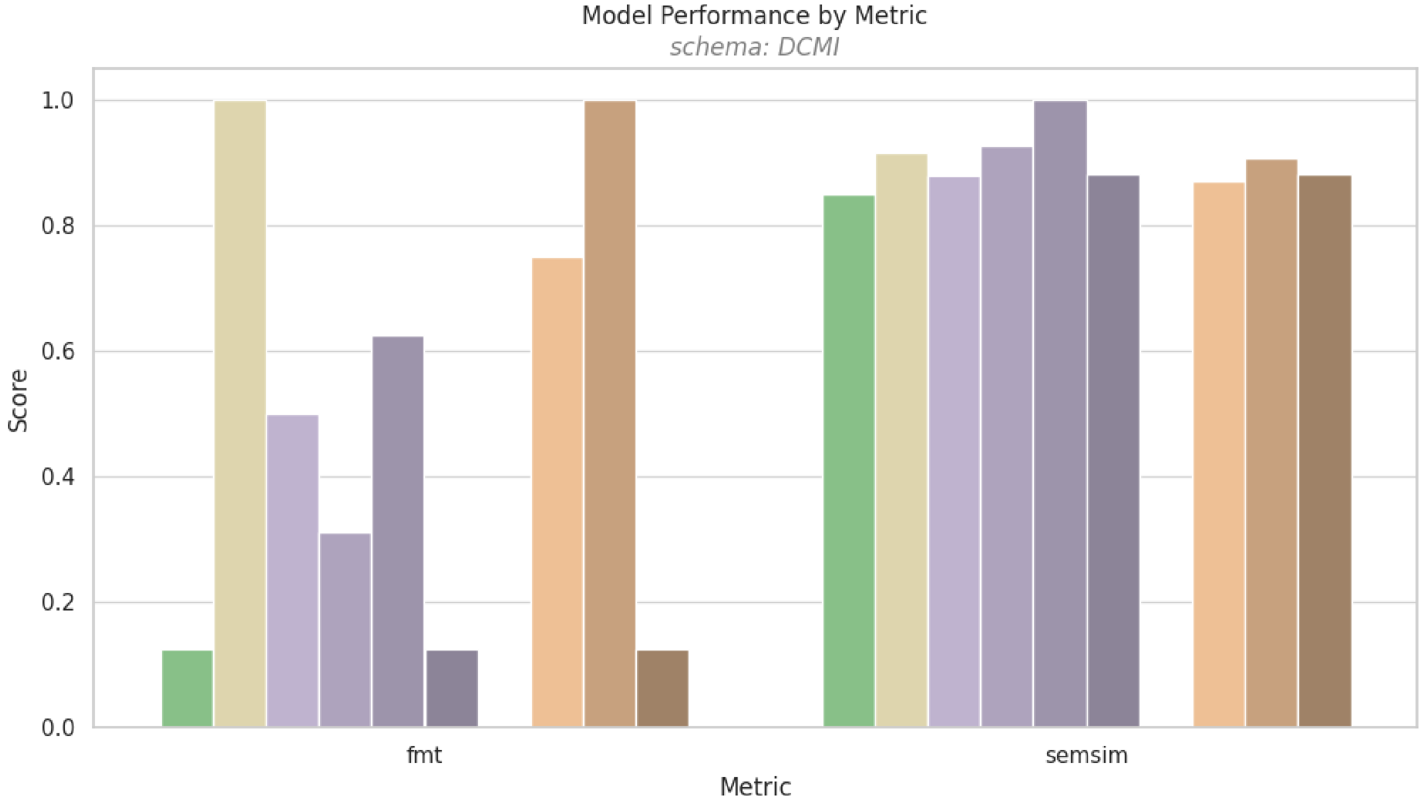
- Harmonic mean of precision and recall
- Averaged over all categories as equal weight
- Balanced dataset, first pass at performance

“High freq. EM” “Control”

“Multiphysics” “Multiphysics”

$$\mathrm{F}_1 = \frac{2TP}{2TP + FP + FN}$$

Model performance



Notes

- Metrics
  - Formatting is a major bottleneck to extracting useful outputs
  - Semantic Similarity to human-generated labels

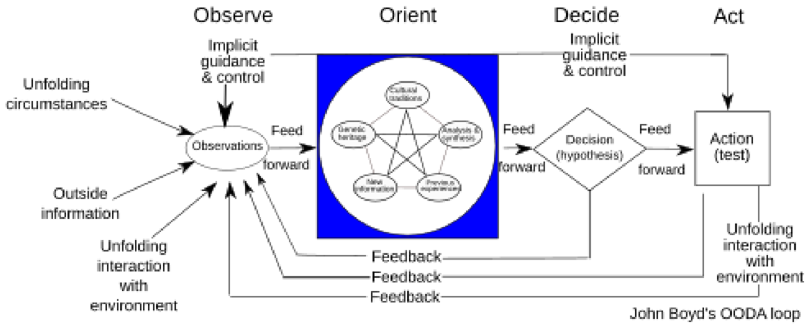
Preliminary results. Proof-of-concept showing potential outputs of study. Dataset is not yet large enough for rigorous statistical analysis.

Limitations and future work

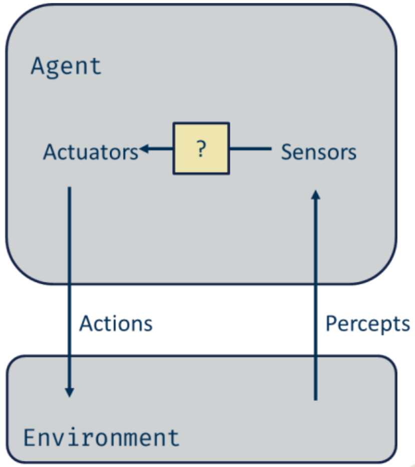
- Dataset is very limited compared to the variety of metadata generation problems in the real world
  - Takes a lot of expertise to understand what data tags are correct
  - Results show what we're trying to measure; limited dataset means drawing conclusions would be premature
  - This is a major open challenge; we invite collaborations and **expertise(!)** on what these benchmarks should look like
- Formatting issues in LLM outputs make results far lower yield than "close" results
  - Invalid JSON due to a missing closing bracket turns into a null score, even when the rest is mostly correct

Final thoughts

- System Engineering Environment
  - How do we represent systems to learn more about them?
- Modeling supports intelligence gathering
  - Can we shorten the modeler’s OODA loop?
  - Modeling is a process of developing infrastructure for analyzing the world
  - In turn this produces intelligence/understanding about a system and its environment, even if in simulation
- The way we represent knowledge drives how we can use it
  - Current efforts to build Model Context Protocol servers do exactly this for the language model/agentic AI ecosystem
  - Systems engineering emphasizes modeling and common abstraction formats to do this
    - Requirements engineering
    - SysML



Observe Orient Decide Act Loop [1]



Agent model of an AI system
Adapted from Russell and Norvig (2010) [1]

1. Wikipedia. “OODA Loop.” https://en.wikipedia.org/wiki/OODA_loop
2. Stuart Russell and Peter Norvig, (2010) Artificial Intelligence: A Modern Approach, 3e.

Discussion

Generating simulation metadata with large language models

J. Kambhampaty, O. Pinon Fischer, D. Mavris

jaka@gatech.edu | olivia.pinon@asdl.gatech.edu | dimitri.mavris@aerospace.gatech.edu

Aerospace Systems Design Laboratory

Backup