Models All the Way Down

Auditing the Invisible Dependencies of Modern LLMs

Ai2

Sanjay Adhikesaven*, Haoxiang Sun*, Sewon Min

BAIR
BERKELEY ARTIFICIAL INTELLIGENCE RESEARCH

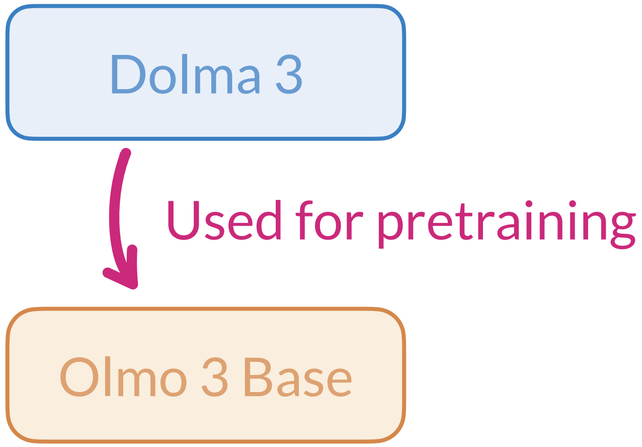
LLMs start helping build other LLMs

LLMs start helping build other LLMs

Olmo 3 Base

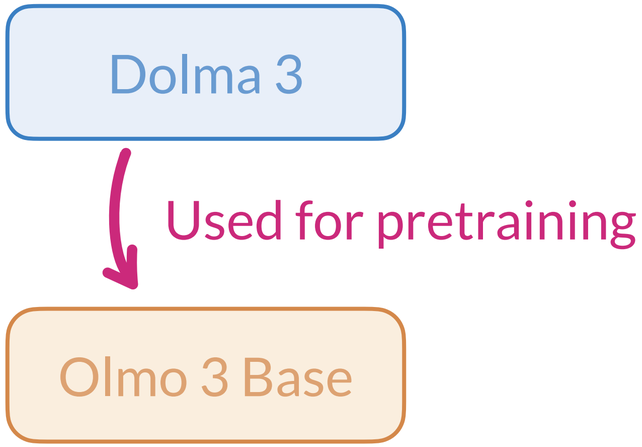
LLMs start helping build other LLMs

LLMs start helping build other LLMs



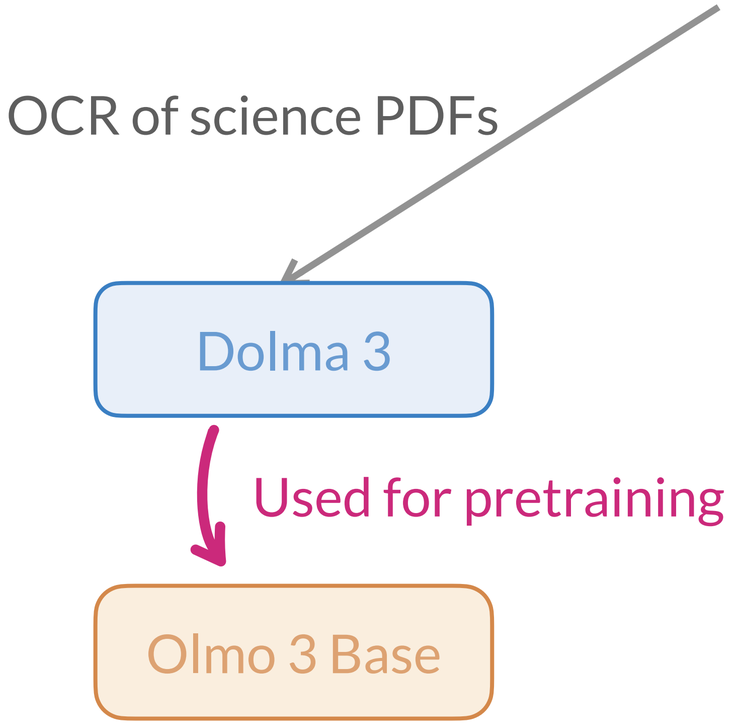
LLMs start helping build other LLMs

LLMs start helping build other LLMs



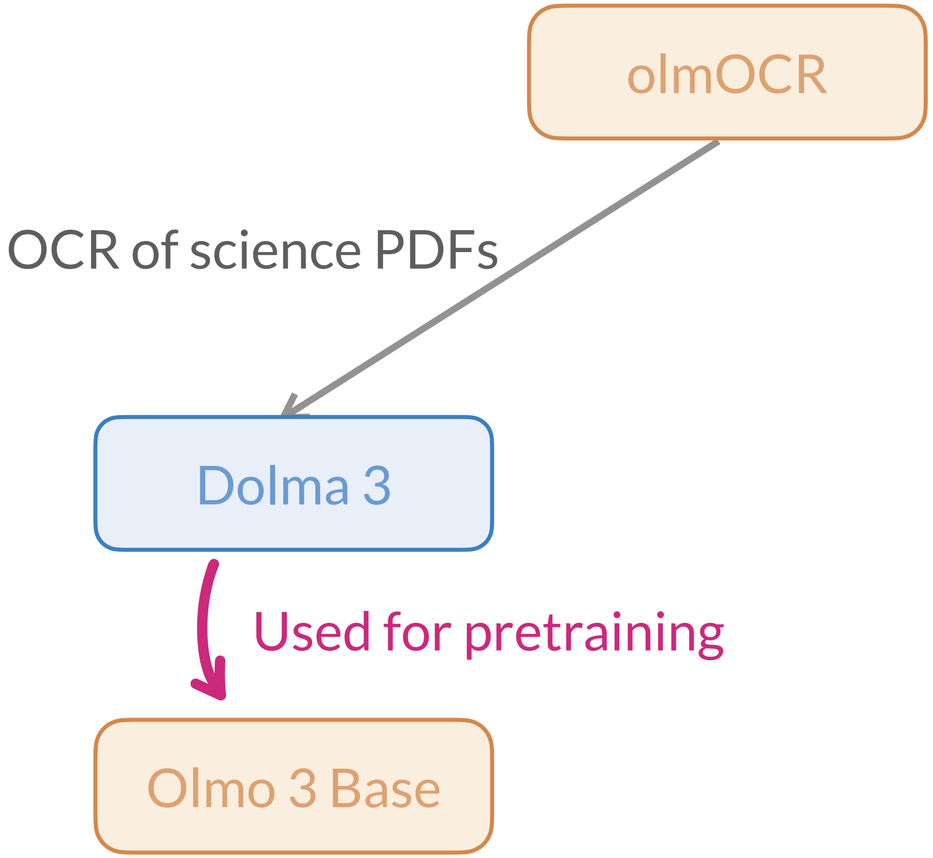
Models Datasets

LLMs start helping build other LLMs



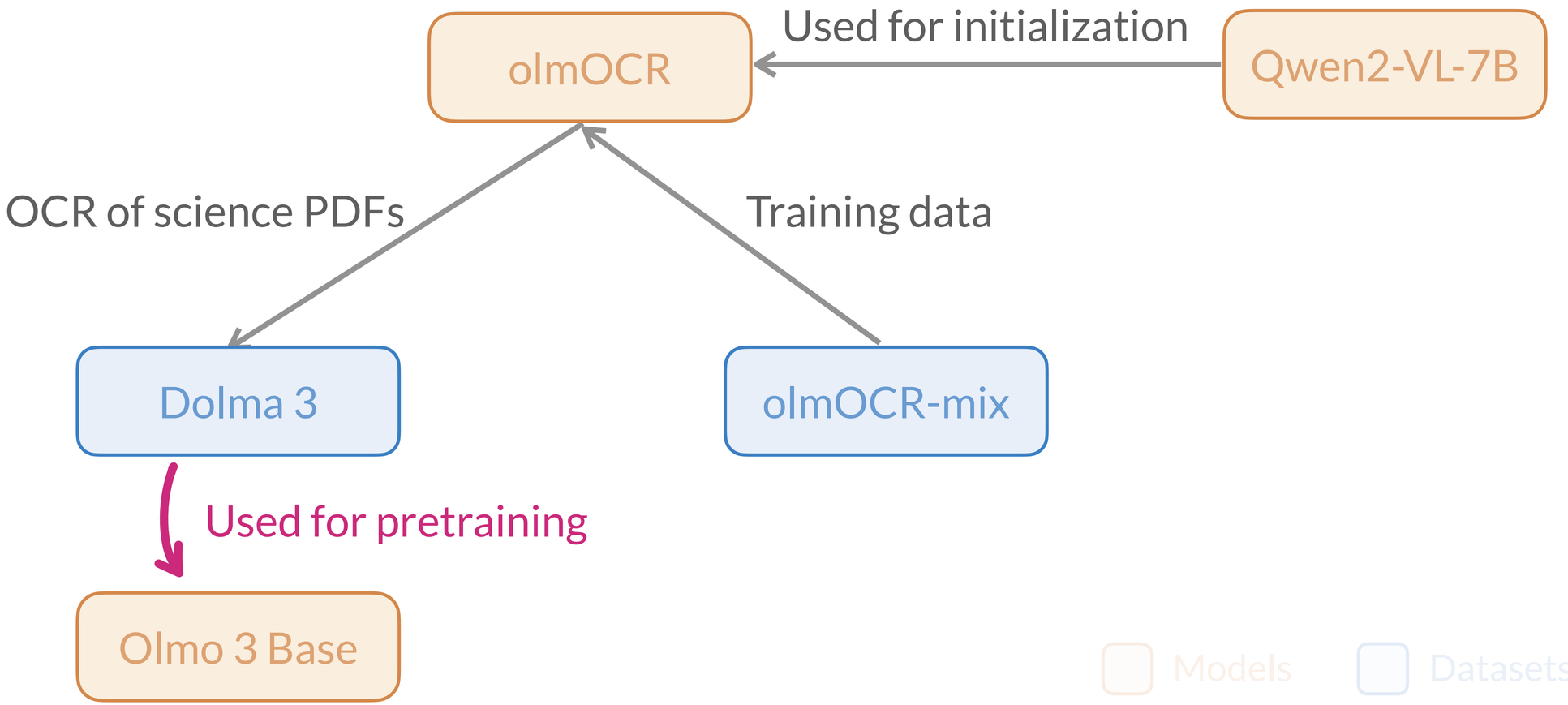
Models Datasets

LLMs start helping build other LLMs

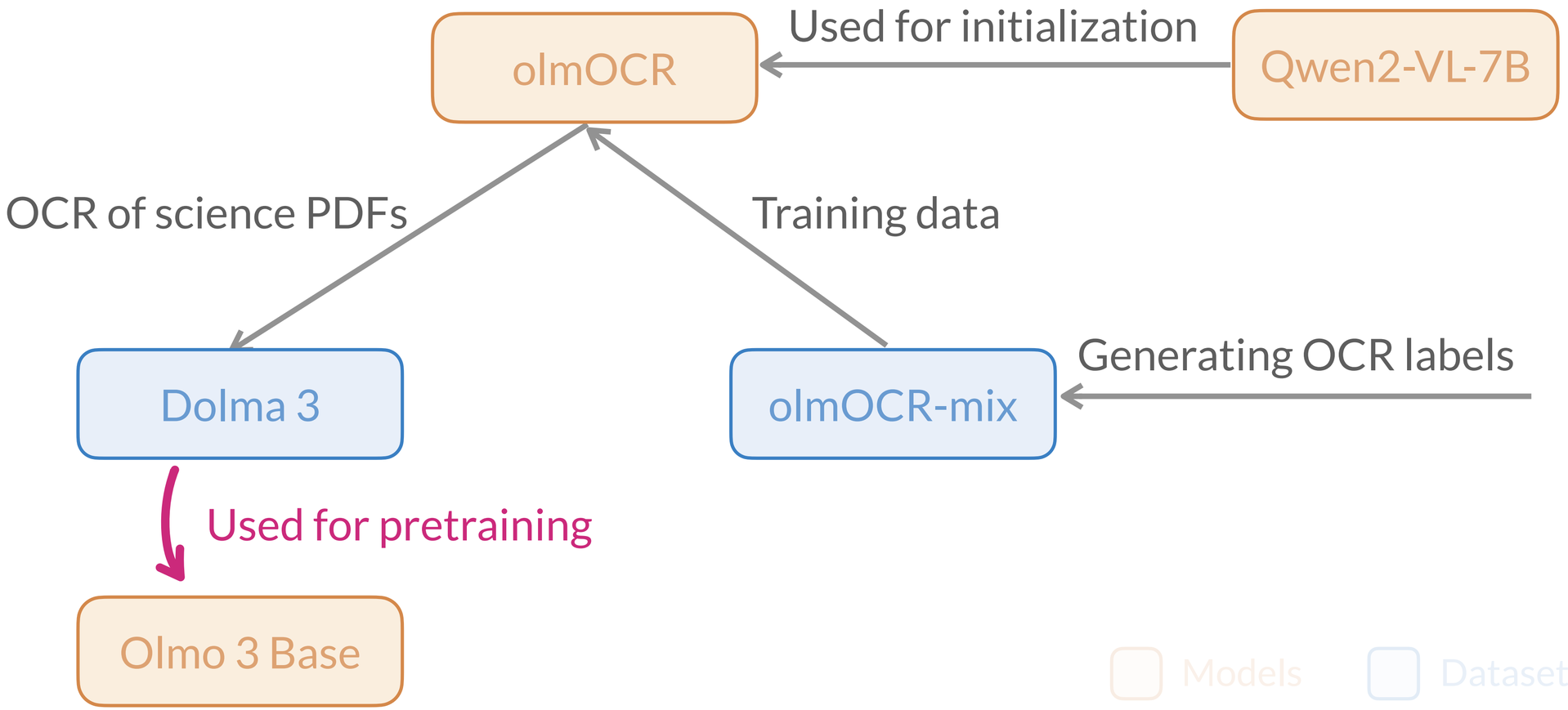


Models Datasets

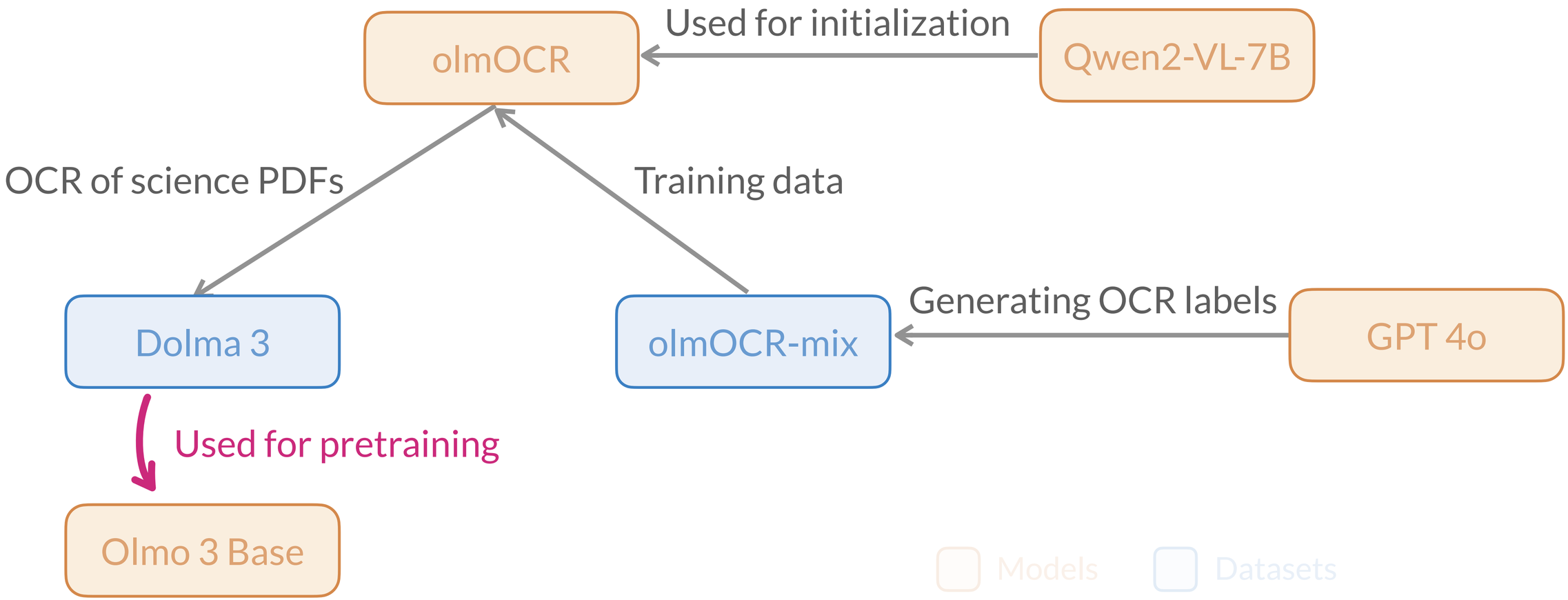
LLMs start helping build other LLMs



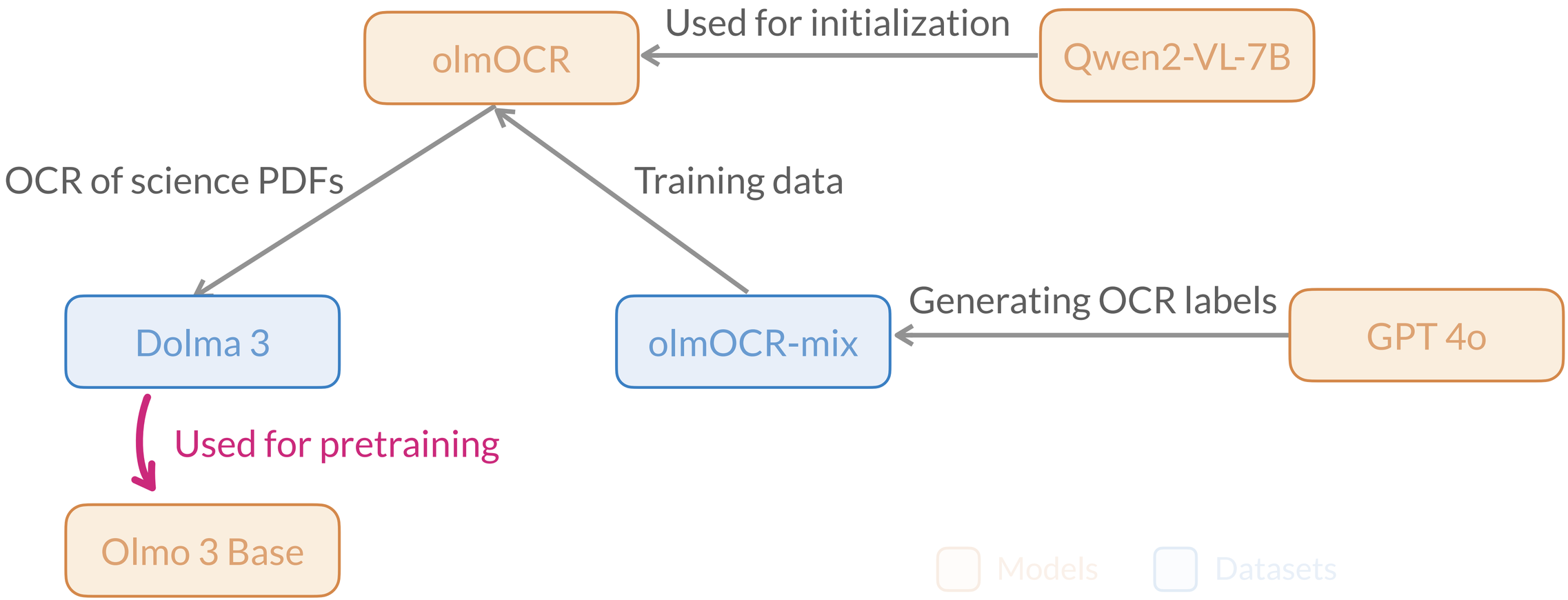
LLMs start helping build other LLMs



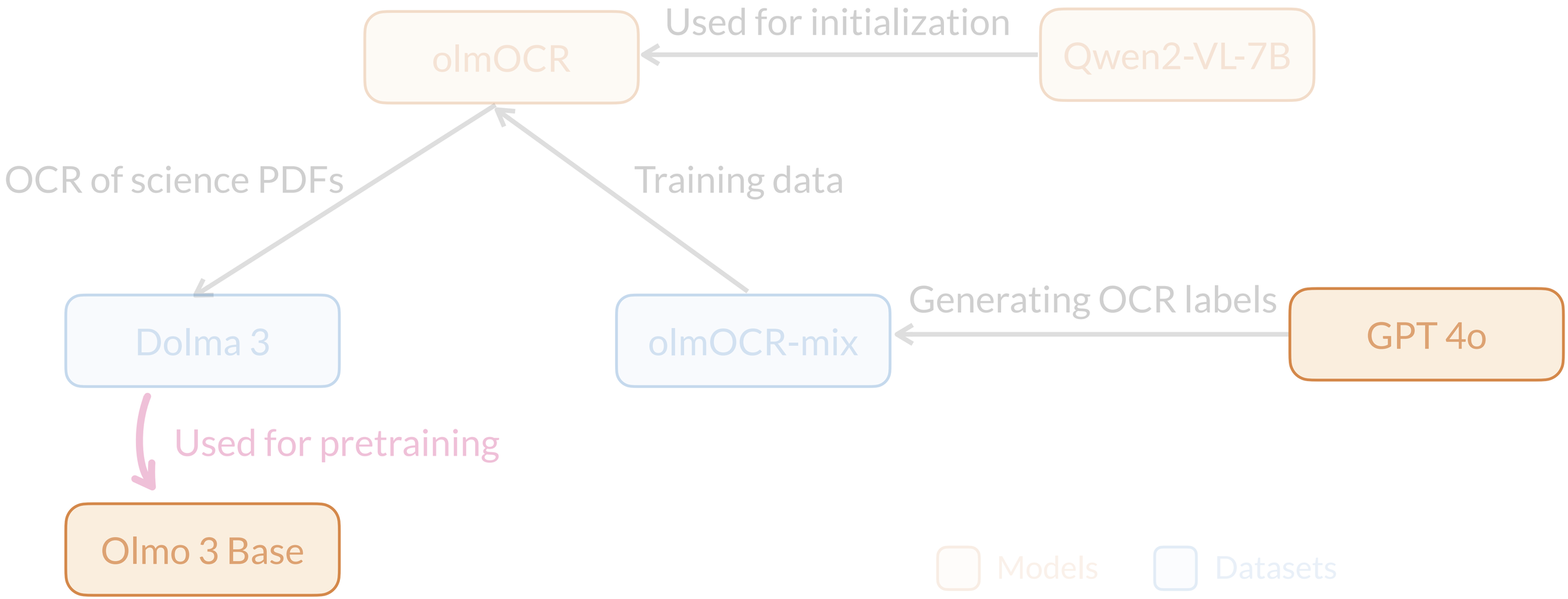
LLMs start helping build other LLMs



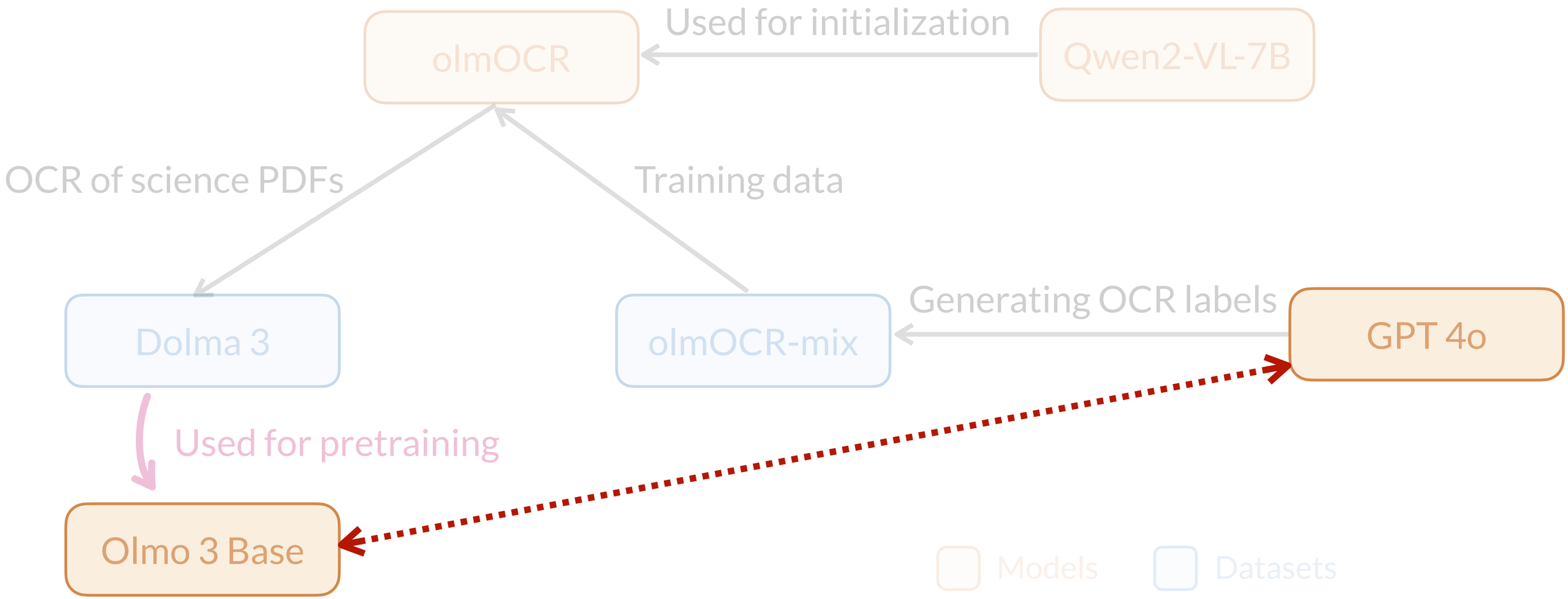
Complex dependencies of LLM development



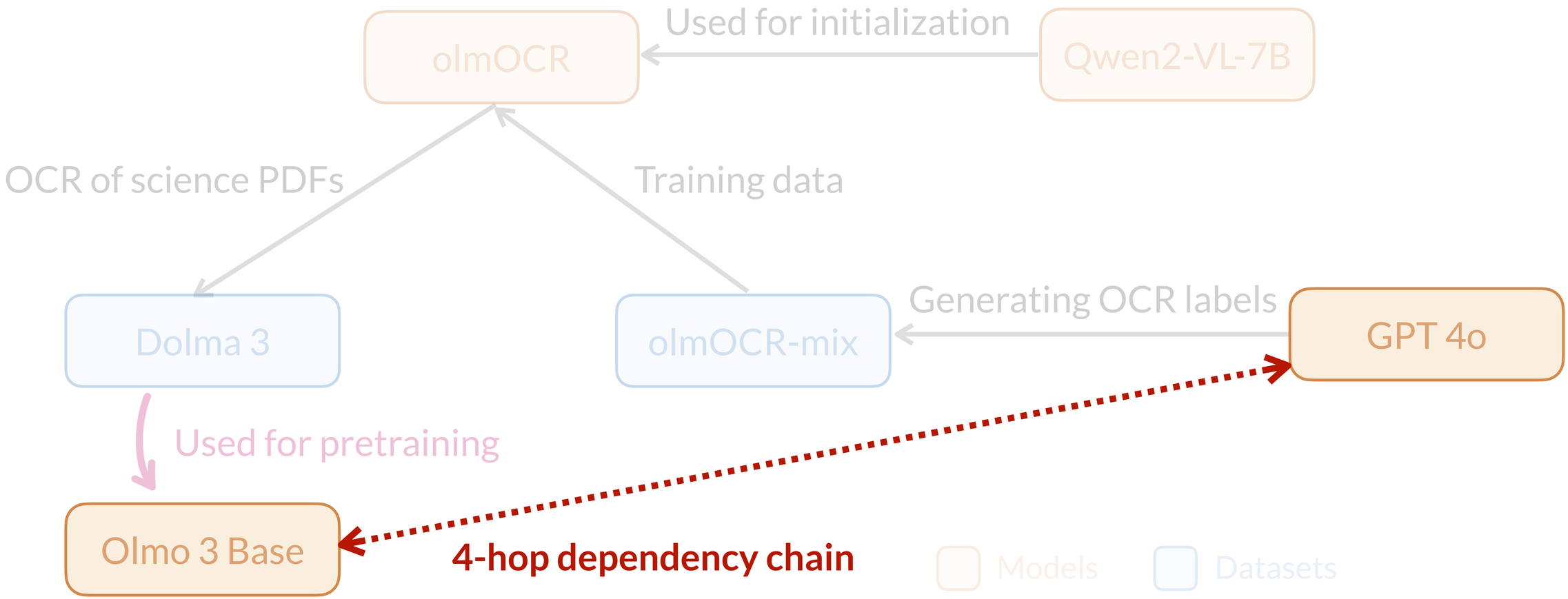
Complex dependencies of LLM development



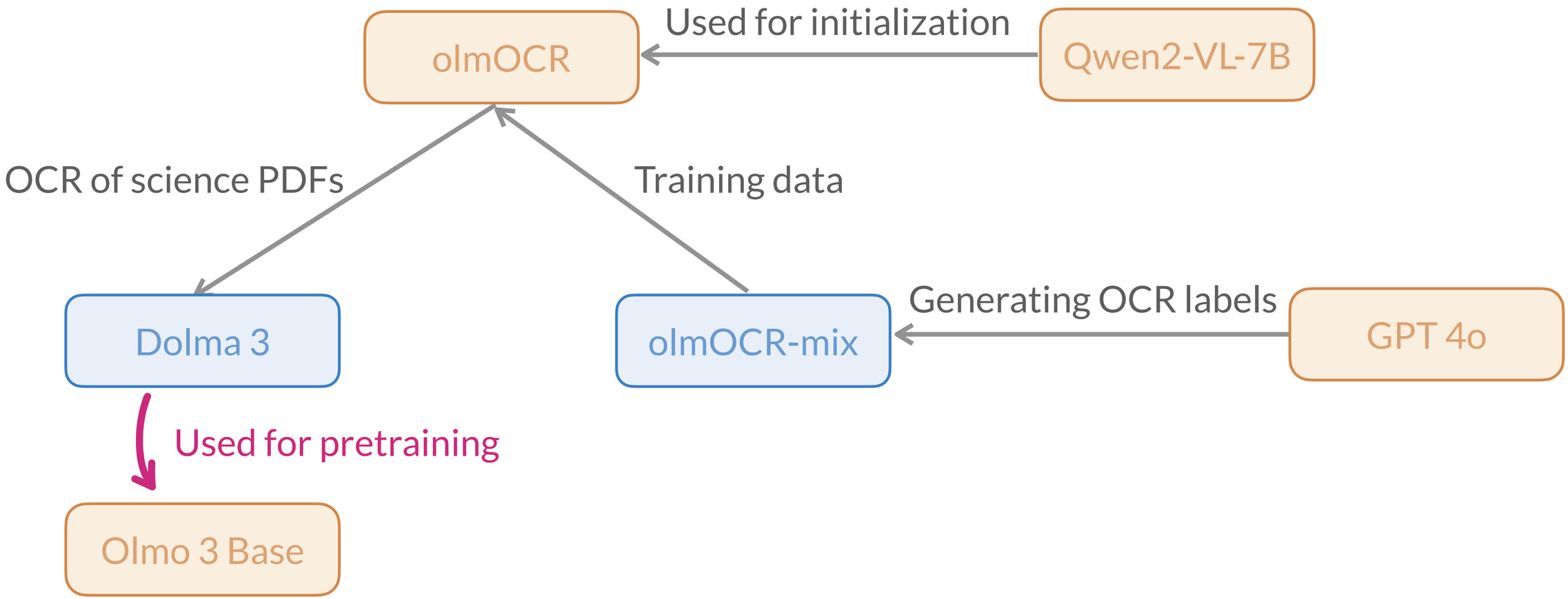
Complex dependencies of LLM development



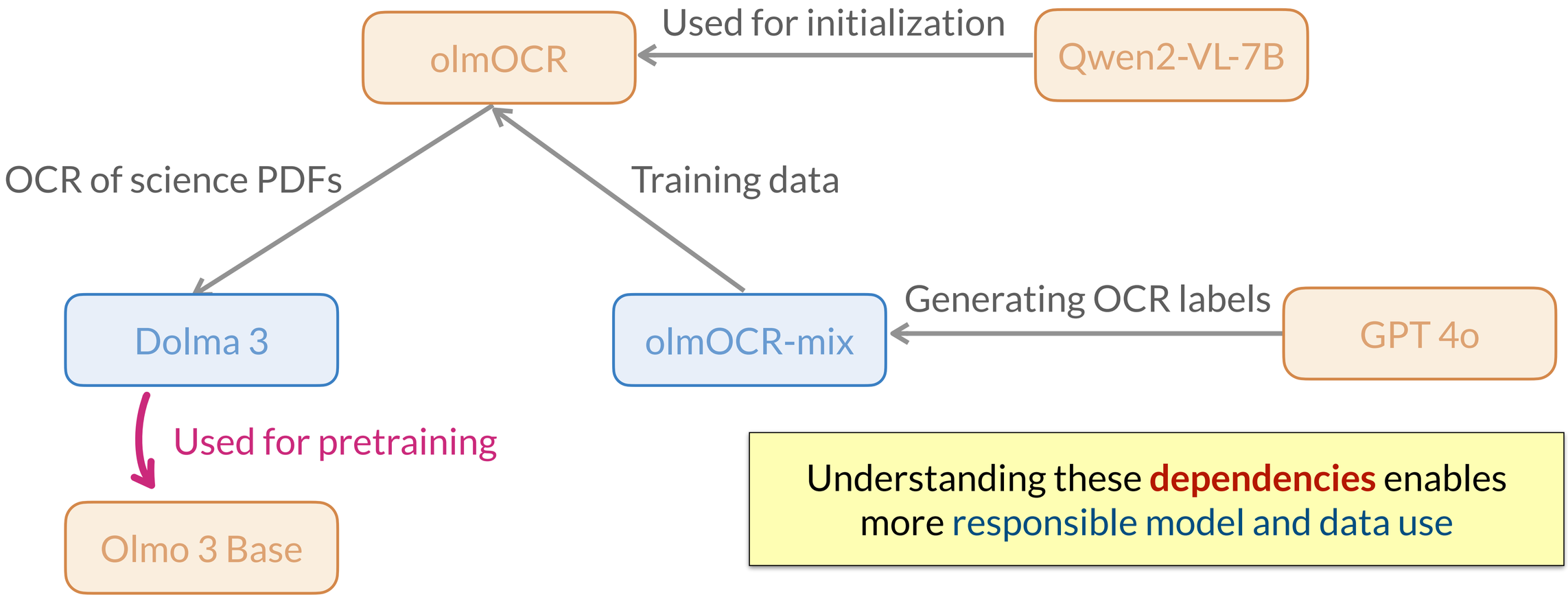
Complex dependencies of LLM development



Complex dependencies of LLM development

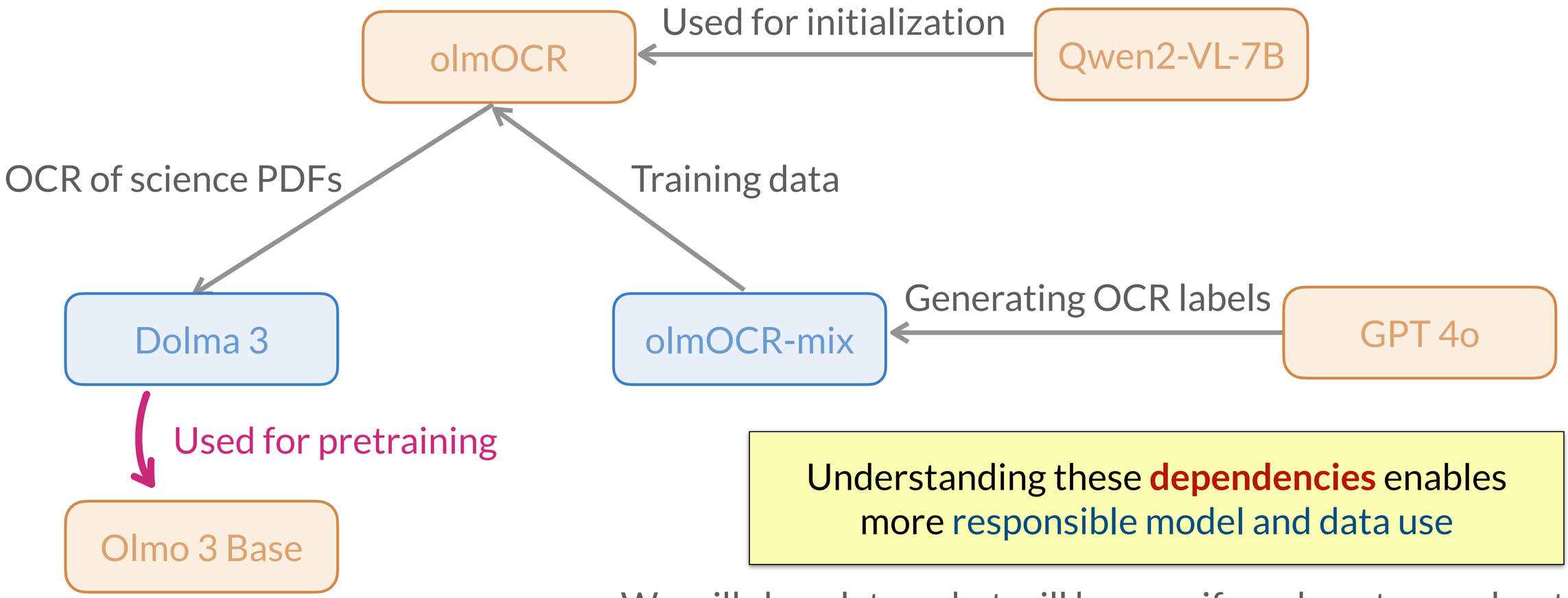


Complex dependencies of LLM development



Understanding these **dependencies** enables more responsible model and data use

Complex dependencies of LLM development



Understanding these **dependencies** enables more responsible model and data use

We will show later what will happen if we do not care about dependencies - license obligations, data contamination, etc.

Complex dependencies of LLM development

olmOCR

Dolma 3

Olmo 3 Base

Complex dependencies of LLM development

olmOCR

Dolma 3

Olmo 3 Base

Olmo 3 technical report

The Olmo 3 family is the strongest collection of fully-open base models, outperforming Stanford Marin (Hall et al., 2025), Apertus (Apertus Team, 2025), and LLM360 K2-V2 (Team et al., 2025). To achieve these results, we construct new datasets for every stage of the model flow. This includes DOLMA **3**, our pretraining data mix encompassing carefully-sampled natural data from crawled sources, our midtraining mix of high-quality data designed to jump-start reasoning, and a large collection of science-focused PDF documents that unlock

Complex dependencies of LLM development

olmOCR

Dolma 3

Used for pretraining

Olmo 3 Base

Olmo 3 technical report

The Olmo 3 family is the strongest collection of fully-open base models, outperforming Stanford Marin (Hall et al., 2025), Apertus (Apertus Team, 2025), and LLM360 K2-V2 (Team et al., 2025). To achieve these results, we construct new datasets for every stage of the model flow. This includes DOLMA **3**, our pretraining data mix encompassing carefully-sampled natural data from crawled sources, our midtraining mix of high-quality data designed to jump-start reasoning, and a large collection of science-focused PDF documents that unlock

Complex dependencies of LLM development

olmOCR

Dolma 3

Used for pretraining

Olmo 3 Base

Dolma 3 dataset card

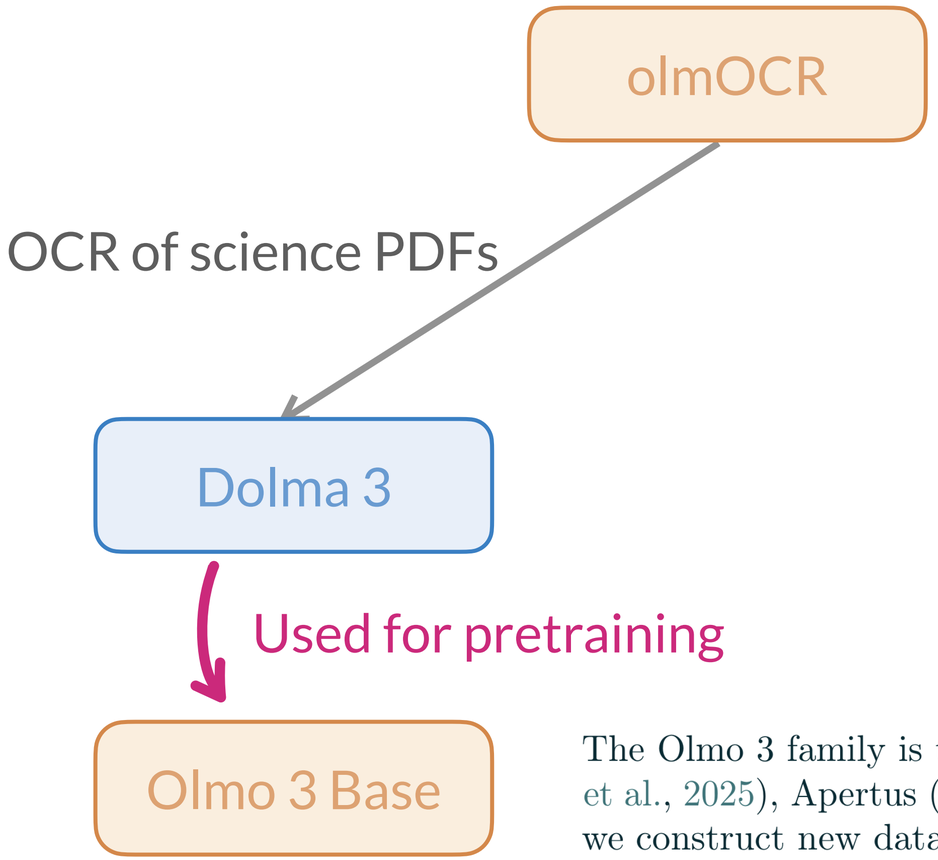
Datasets: allenai / **dolma3_mix-6T-1025-7B**

Source	Doc Type	Tokens	Bytes (uncompressed)	Documents	License
common_crawl	web pages	4.51T	18.0TB	3.15B	ODC-BY
olmocr_science_pdfs	academic papers	805B	3.22TB	83.8M	ODC-BY

Olmo 3 technical report

The Olmo 3 family is the strongest collection of fully-open base models, outperforming Stanford Marin (Hall et al., 2025), Apertus (Apertus Team, 2025), and LLM360 K2-V2 (Team et al., 2025). To achieve these results, we construct new datasets for every stage of the model flow. This includes DOLMA 3, our pretraining data mix encompassing carefully-sampled natural data from crawled sources, our midtraining mix of high-quality data designed to jump-start reasoning, and a large collection of science-focused PDF documents that unlock

Complex dependencies of LLM development



Dolma 3 dataset card

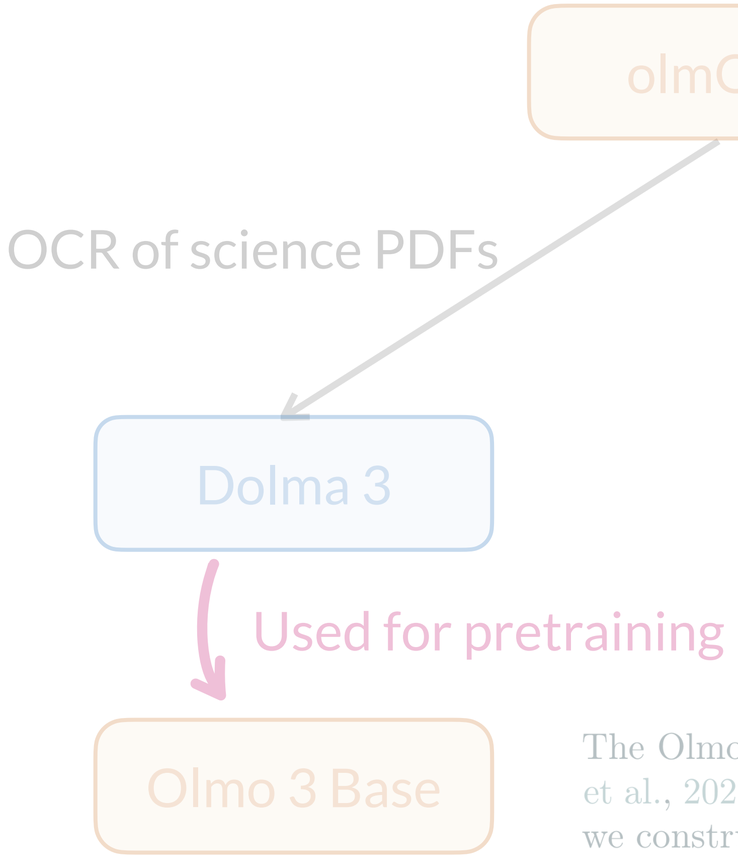
Datasets: allenai/**dolma3_mix-6T-1025-7B**

Source	Doc Type	Tokens	Bytes (uncompressed)	Documents	License
common_crawl	web pages	4.51T	18.0TB	3.15B	ODC-BY
olmocr_science_pdfs	academic papers	805B	3.22TB	83.8M	ODC-BY

Olmo 3 technical report

The Olmo 3 family is the strongest collection of fully-open base models, outperforming Stanford Marin (Hall et al., 2025), Apertus (Apertus Team, 2025), and LLM360 K2-V2 (Team et al., 2025). To achieve these results, we construct new datasets for every stage of the model flow. This includes DOLMA 3, our pretraining data mix encompassing carefully-sampled natural data from crawled sources, our midtraining mix of high-quality data designed to jump-start reasoning, and a large collection of science-focused PDF documents that unlock

Complex dependencies of LLM development



Dolma 3 dataset card

Datasets: allenai/dolma3_mix-6T-1025-7B

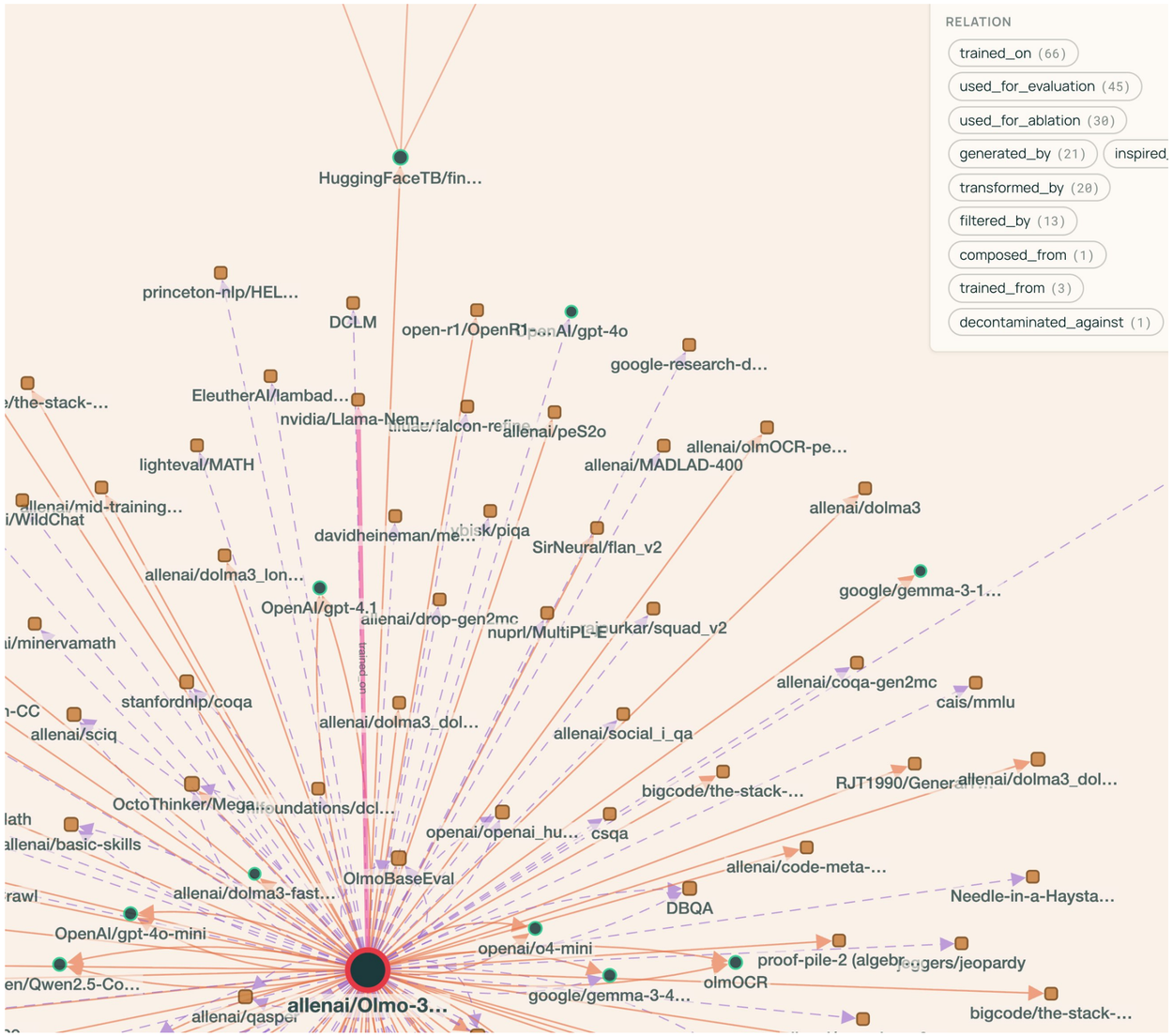
Source	Doc Type	Tokens	Bytes (uncompressed)	Documents	License
common_crawl	web pages	4.51T	18.0TB	3.15B	ODC-BY
olmocr_science_pdfs	academic papers	805B	3.22TB	83.8M	ODC-BY

Olmo 3 technical report

The Olmo 3 family is the strongest collection of fully-open base models, outperforming Stanford Marin (Hall et al., 2025), Apertus (Apertus Team, 2025), and LLM360 K2-V2 (Team et al., 2025). To achieve these results, we construct new datasets for every stage of the model flow. This includes DOLMA 3, our pretraining data mix encompassing carefully-sampled natural data from crawled sources, our midtraining mix of high-quality data designed to jump-start reasoning, and a large collection of science-focused PDF documents that unlock

Complex dependencies of LLM development

ModSleuth: Reconstructing dependency graphs

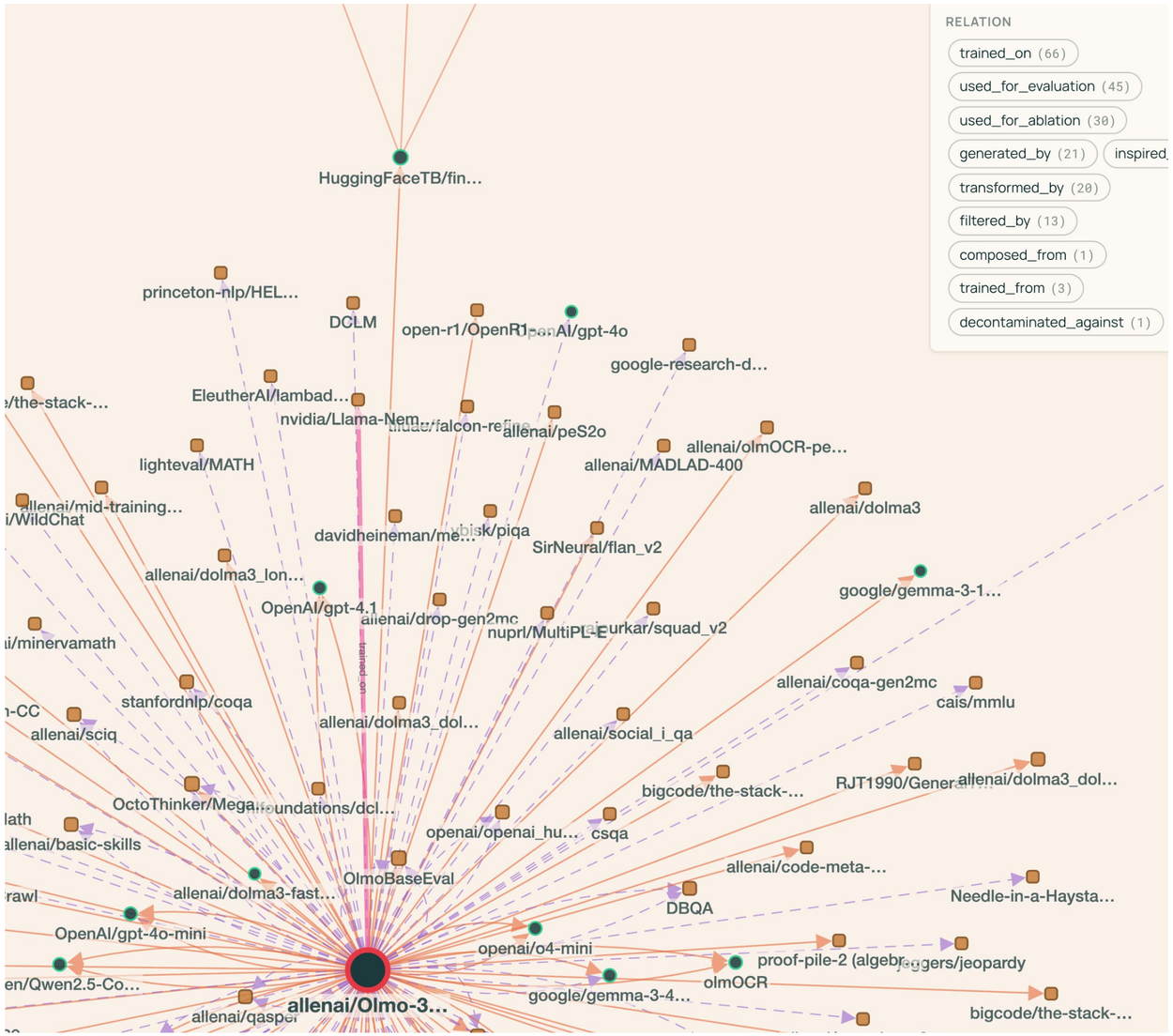


circle = model square = dataset

arrow = one documented dependency

Target Olmo 3 at the bottom centre; depth grows outward.

ModSleuth: Reconstructing dependency graphs



https://modsleuth.cal-data-audit.org

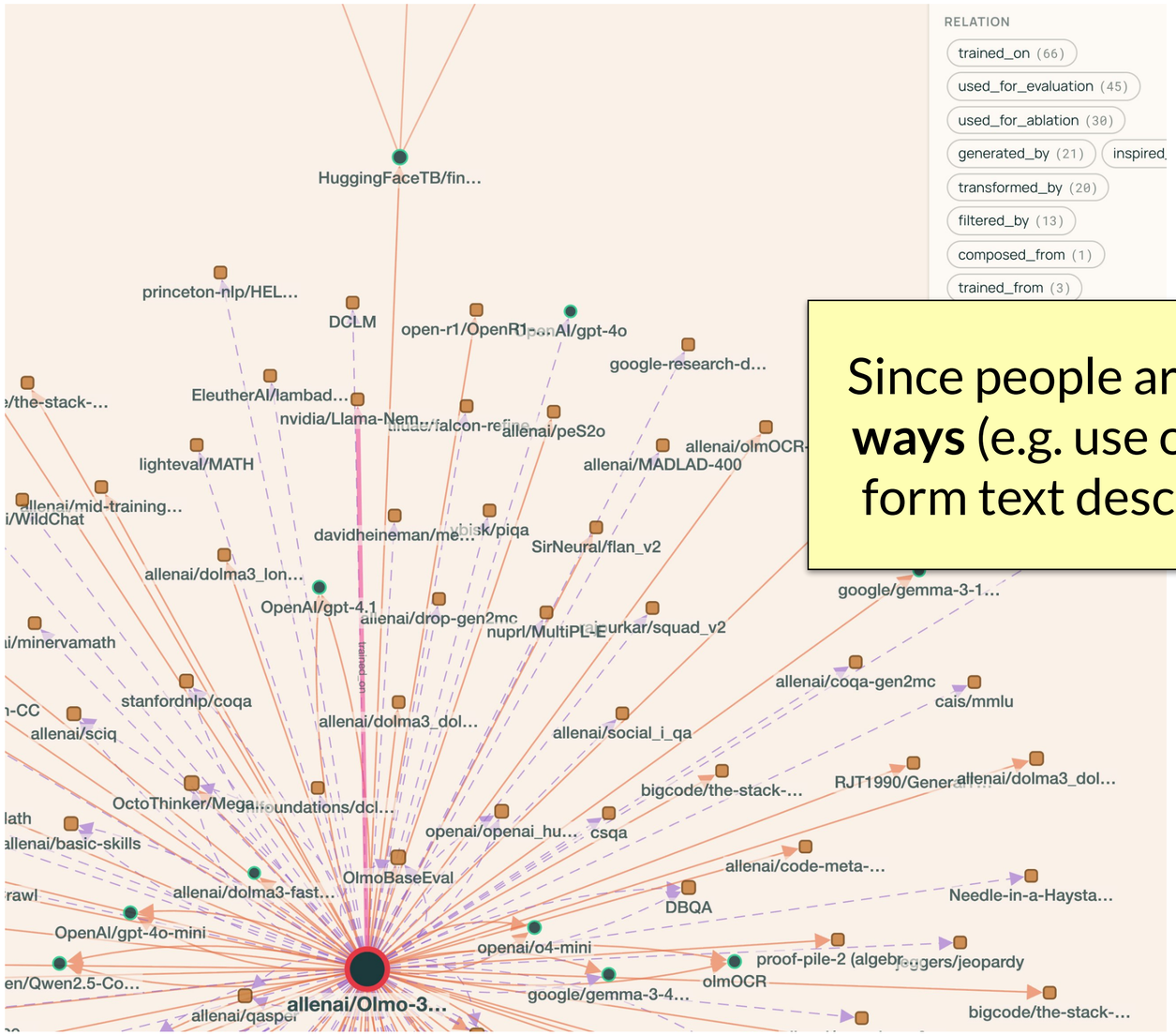
● circle = model ■ square = dataset

—— arrow = one documented dependency

Target Olmo 3 at the bottom centre; depth grows outward.

ModSleuth: Reconstructing dependency graphs

https://modsleuth.cal-data-audit.org



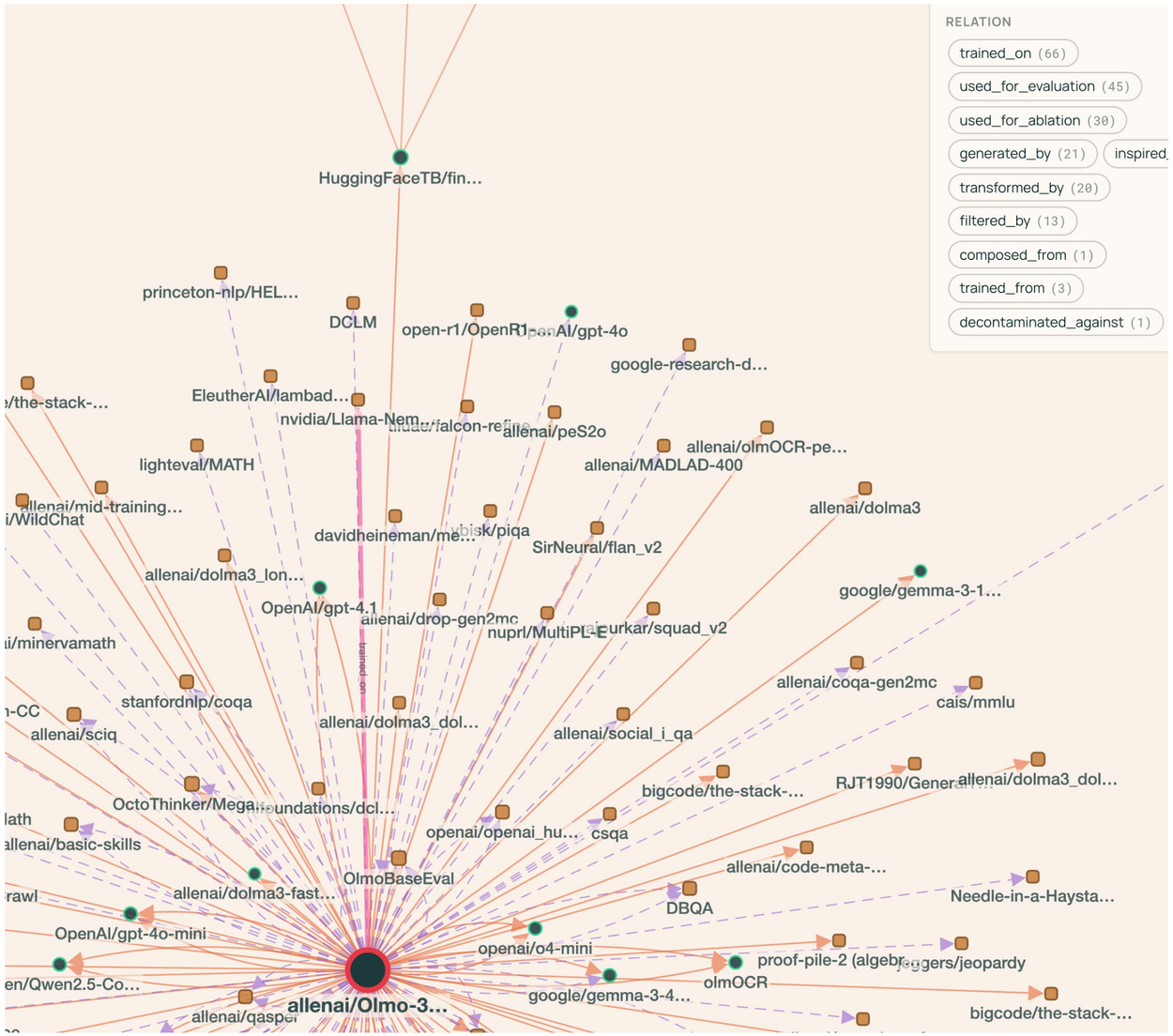
Since people are using models in **more and more creative ways** (e.g. use olmOCR to process PDFs), we ask for free-form text description for a dependency to preserve info.

circle = model square = dataset

arrow = one documented dependency

Target Olmo 3 at the bottom centre; depth grows outward.

ModSleuth: Reconstructing dependency graphs

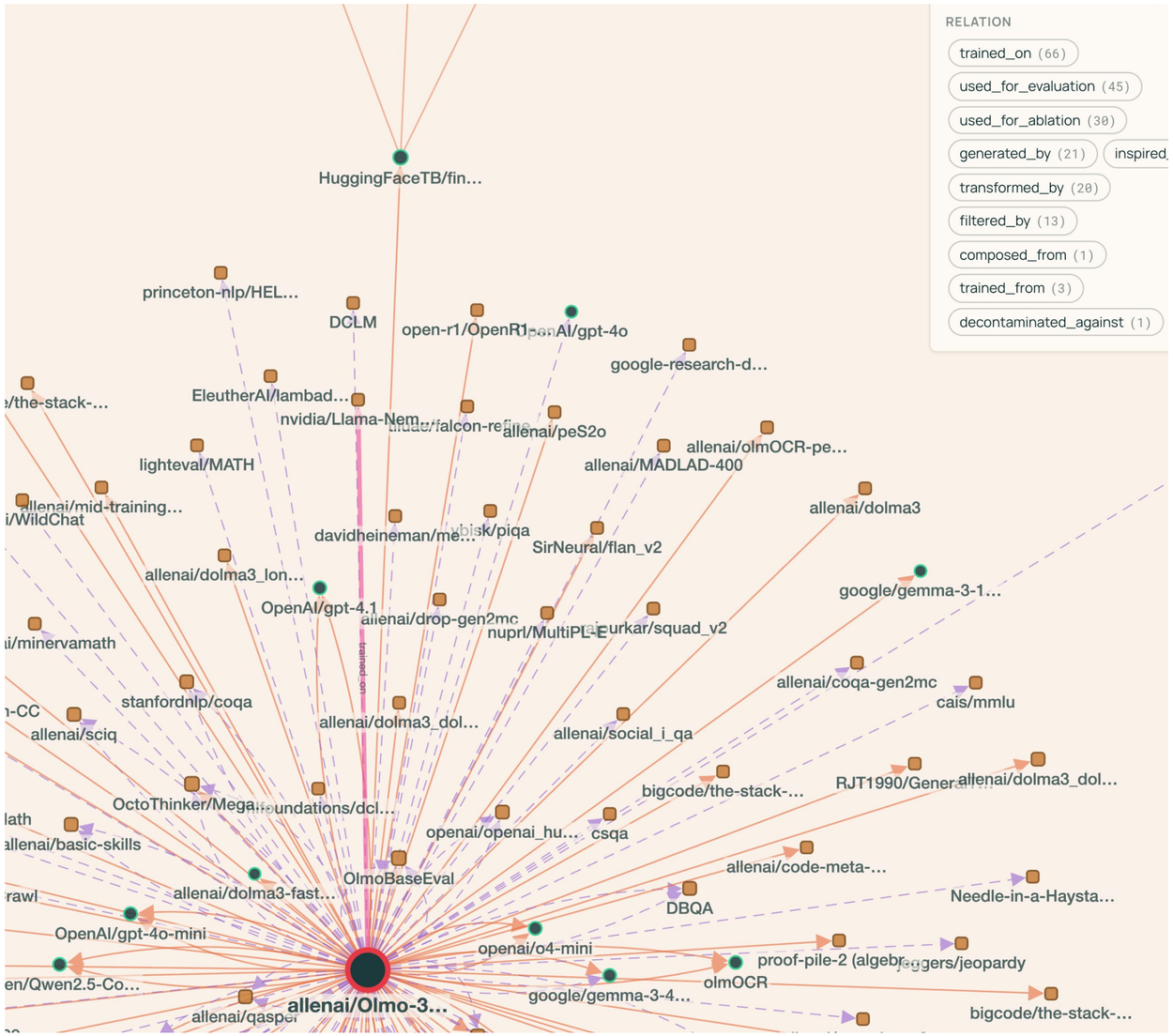


circle = model square = dataset

arrow = one documented dependency

Target Olmo 3 at the bottom centre; depth grows outward.

ModSleuth: Reconstructing dependency graphs



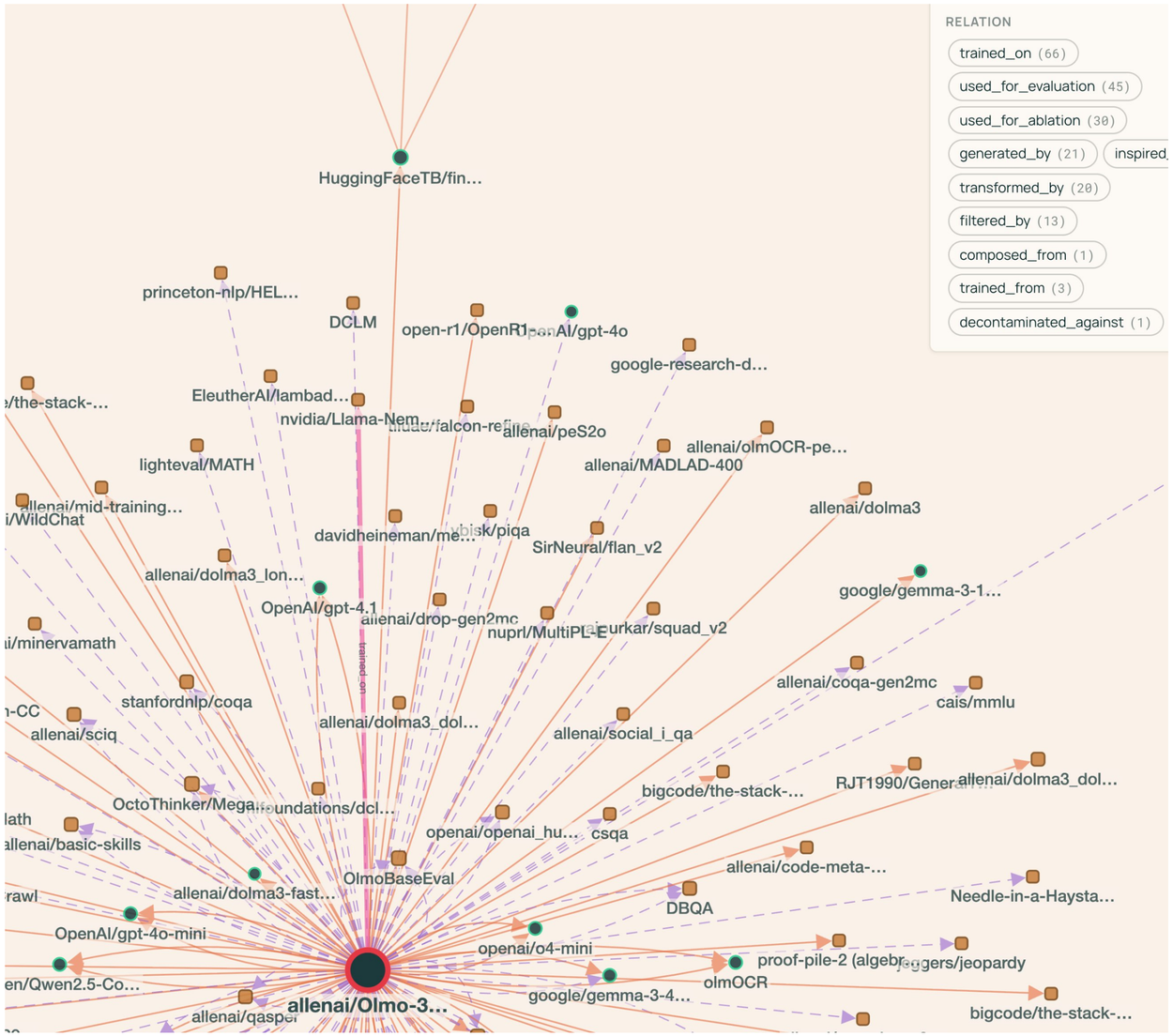
Our task: Recursively reconstructs LLM dependency graphs from public artifacts with source-grounded evidence

● circle = model ■ square = dataset

—— arrow = one documented dependency

Target Olmo 3 at the bottom centre; depth grows outward.

ModSleuth: Reconstructing dependency graphs



Our task: Recursively reconstructs LLM dependency graphs from public artifacts with source-grounded evidence

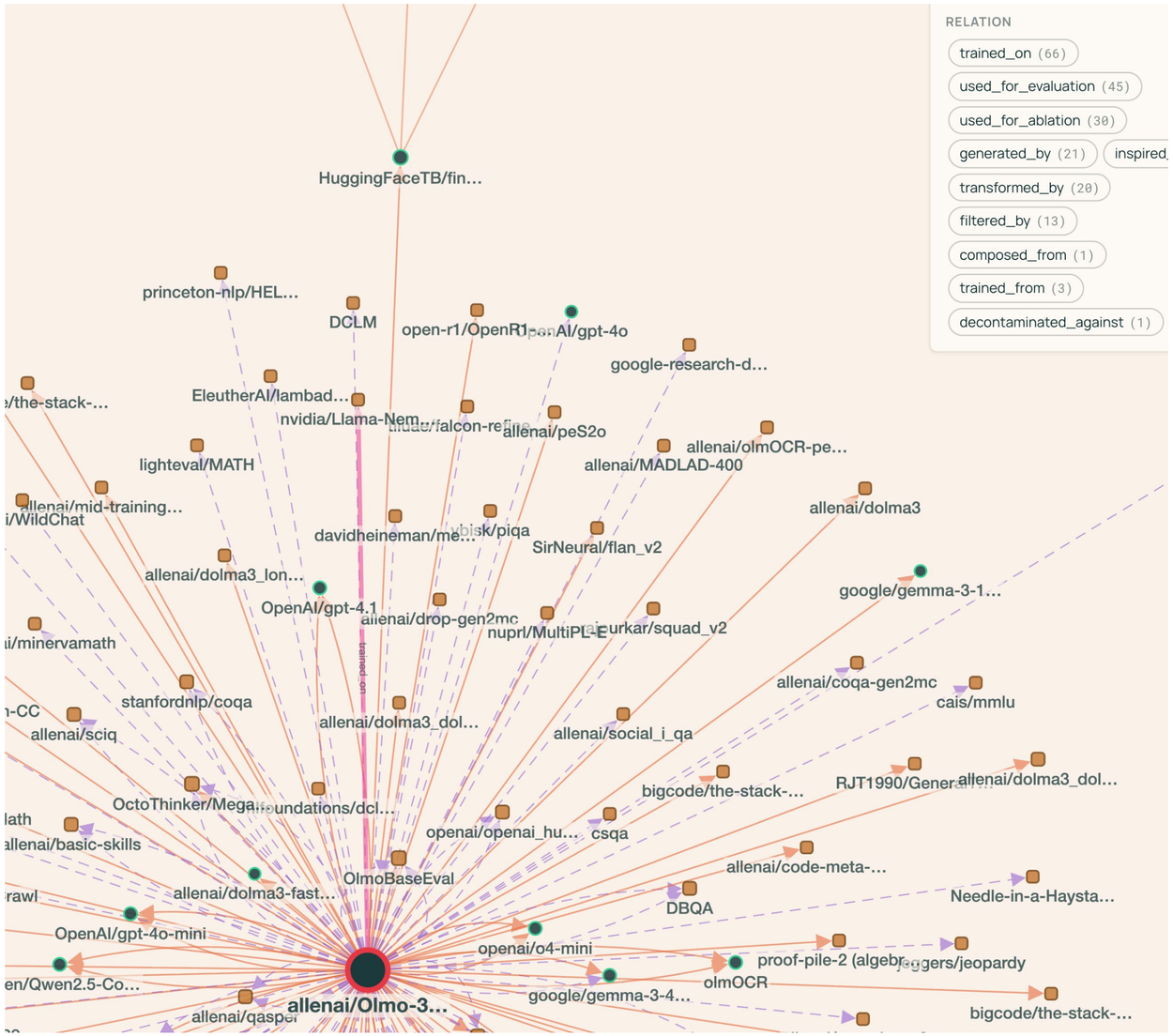
409 upstream artifacts behind Olmo 3 Inst.

● circle = model ■ square = dataset

—— arrow = one documented dependency

Target Olmo 3 at the bottom centre; depth grows outward.

ModSleuth: Reconstructing dependency graphs



Our task: Recursively reconstructs LLM dependency graphs from public artifacts with source-grounded evidence

409 upstream artifacts behind Olmo 3 Inst.

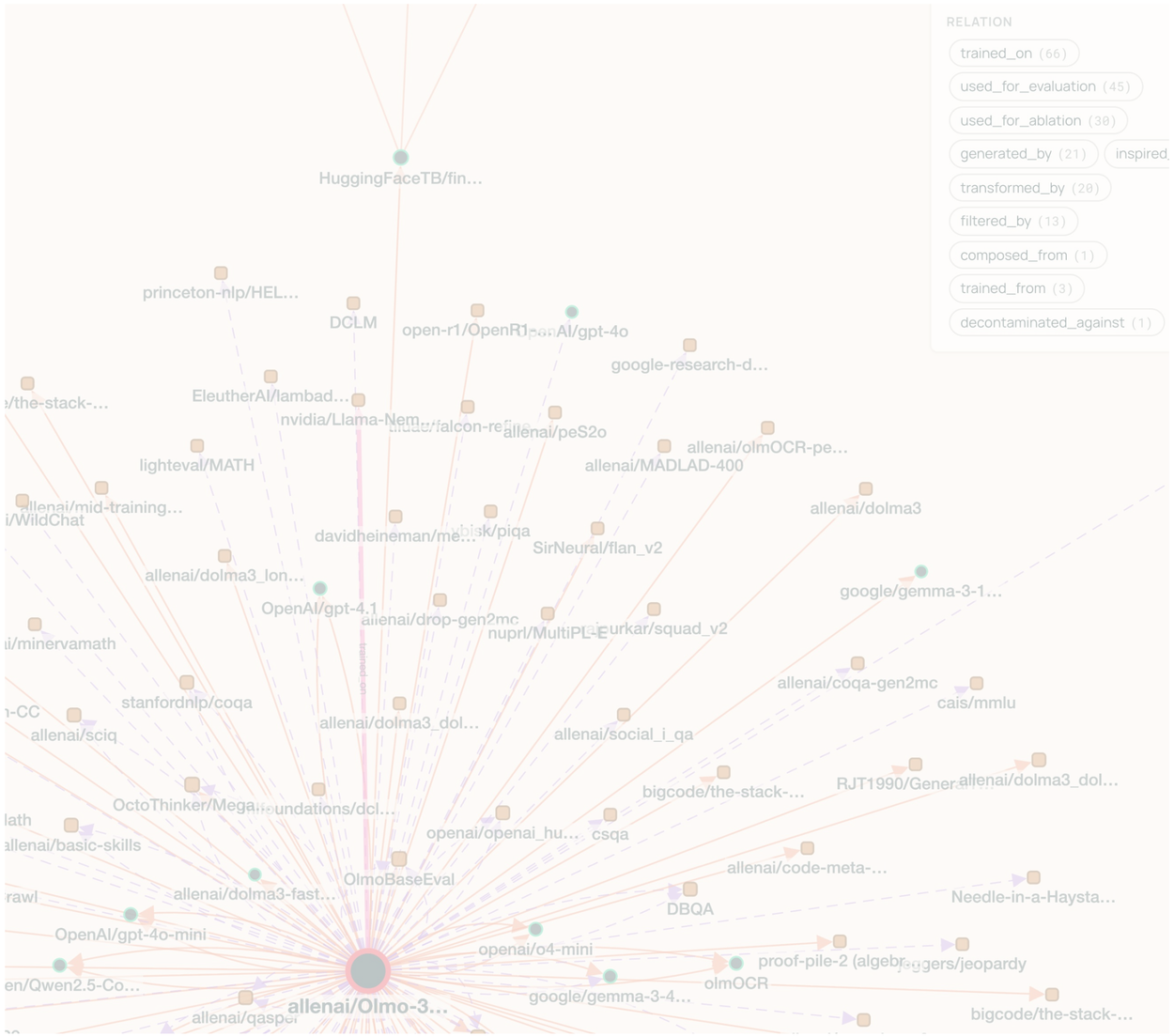
8 hops deepest dependency chain

● circle = model ■ square = dataset

—— arrow = one documented dependency

Target Olmo 3 at the bottom centre; depth grows outward.

ModSleuth: Reconstructing dependency graphs



Our task: Recursively reconstructs LLM dependency graphs from public artifacts with source-grounded evidence

409 upstream artifacts behind Olmo 3 Inst.

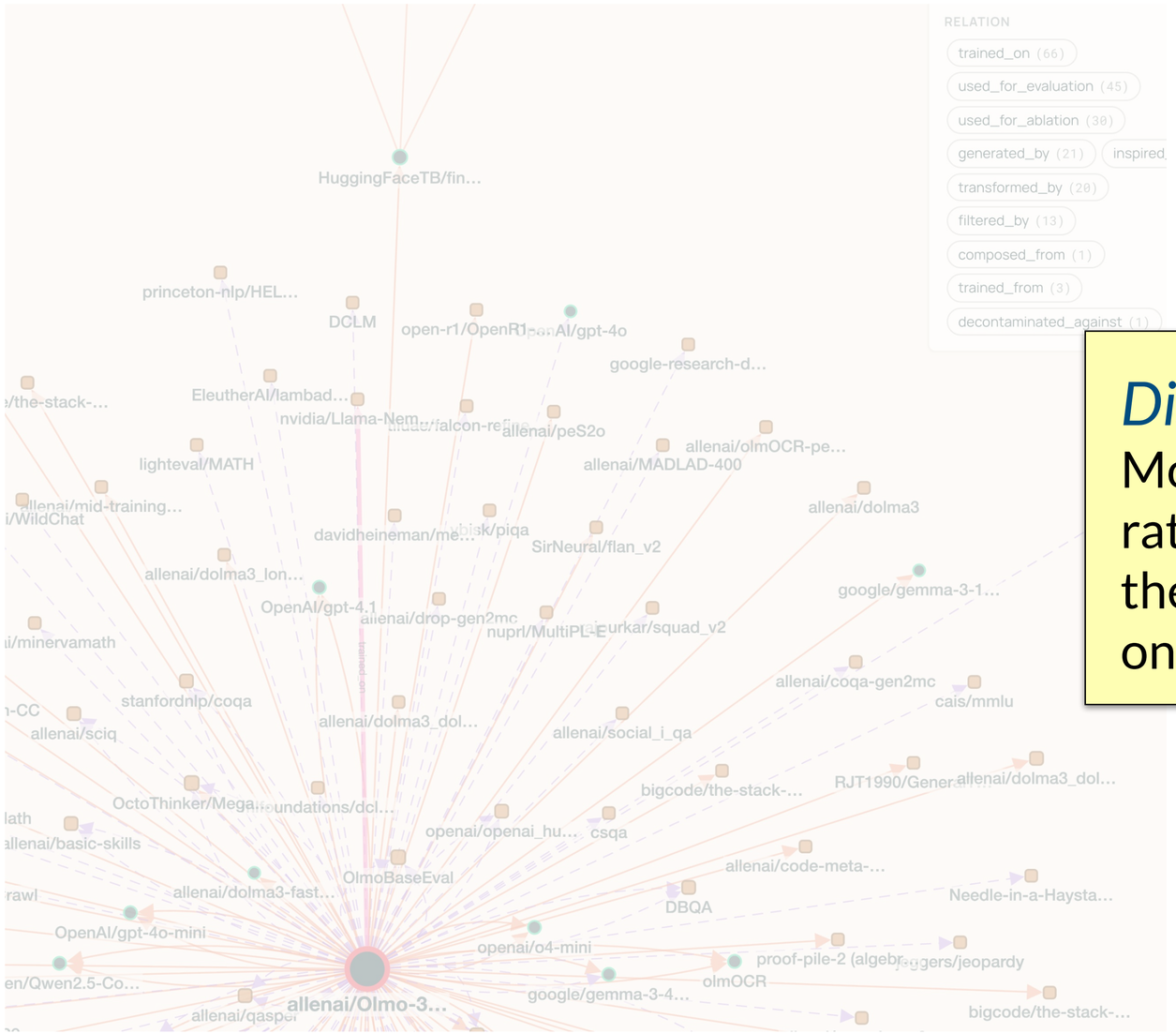
8 hops deepest dependency chain

circle = model square = dataset

arrow = one documented dependency

Target Olmo 3 at the bottom centre; depth grows outward.

ModSleuth: Reconstructing dependency graphs



Our task: Recursively reconstructs LLM dependency graphs from public artifacts with source-grounded evidence

Disclosure:
ModSleuth identifies ***declared*** dependencies rather than ***true*** dependencies, meaning that the dependencies we detect likely represent only a lower bound on the actual dependencies.

circle = model square = dataset

arrow = one documented dependency

Target Olmo 3 at the bottom centre; depth grows outward.

Talk Overview

1. What counts as a dependency, and why is tracing them hard?

   Much more than a list of names! Every dependency enters through some operations (generating, filtering), many hops upstream, documented in scattered and ambiguous sources.

2. Can we *automatically* and *reliably* recover the dependency graph from public artifacts?

   Yes! ModSleuth, an agentic system we developed, recursively expands the graph and grounds every edge in source evidence—recovering 3x more verified dependencies than baselines.

3. What do the recovered graphs reveal?

   Plenty that no single page shows: hidden upstream models, license terms riding along multiple hops, benchmarks on both sides of training and eval, etc.

Talk Overview

1. What counts as a dependency, and why is tracing them hard?

   Much more than a list of names! Every dependency enters through some operations (generating, filtering), many hops upstream, documented in scattered and ambiguous sources.

2. Can we *automatically* and *reliably* recover the dependency graph from public artifacts?

   Yes! ModSleuth, an agentic system we developed, recursively expands the graph and grounds every edge in source evidence—recovering 3x more verified dependencies than baselines.

3. What do the recovered graphs reveal?

   Plenty that no single page shows: hidden upstream models, license terms riding along multiple hops, benchmarks on both sides of training and eval, etc.

Talk Overview

1. What counts as a dependency, and why is tracing them hard?

   Much more than a list of names! Every dependency enters through some operations (generating, filtering), many hops upstream, documented in scattered and ambiguous sources.

LLM Agents are getting better at **information extraction**…

ependency graph

expands the graph and grounds
ependencies than baselines.

ense terms riding along multiple

Talk Overview

1. What counts as a dependency, and why is tracing them hard?

   Much more than a list of names! Every dependency enters through some operations (generating, filtering), many hops upstream, documented in scattered and ambiguous sources.

LLM Agents are getting better at **information extraction**…

ependency graph

expands the graph and grounds
ependencies than baselines.

ense terms riding along multiple

Talk Overview

1. What counts as a dependency, and why is tracing them hard?

   Much more than a list of names! Every dependency enters through some operations (generating, filtering), many hops upstream, documented in scattered and ambiguous sources.

LLM Agents are getting better at **information extraction**…

Information extraction (from a specific document to facts) is **no longer the primary challenge**

ependency graph

expands the graph and grounds
ependencies than baselines.

ense terms riding along multiple

Talk Overview

1. What counts as a dependency, and why is tracing them hard?

   Much more than a list of names! Every dependency enters through some operations (generating, filtering), many hops upstream, documented in scattered and ambiguous sources.

LLM Agents are getting better at **information extraction**…

Information extraction (from a specific document to facts) is **no longer the primary challenge**

ependency graph

expands the graph and grounds
ependencies than baselines.

ense terms riding along multiple

Talk Overview

1. What counts as a dependency, and why is tracing them hard?

   Much more than a list of names! Every dependency enters through some operations (generating, filtering), many hops upstream, documented in scattered and ambiguous sources.

LLM Agents are getting better at **information extraction**…

Information extraction (from a specific document to facts) is **no longer the primary challenge**

Key obstacles:

ependency graph

expands the graph and grounds
ependencies than baselines.

ense terms riding along multiple

Talk Overview

1. What counts as a dependency, and why is tracing them hard?

   Much more than a list of names! Every dependency enters through some operations (generating, filtering), many hops upstream, documented in scattered and ambiguous sources.

LLM Agents are getting better at **information extraction**…

Information extraction (from a specific document to facts) is **no longer the primary challenge**

Key obstacles:
(1) What constitutes a dependency

ependency graph

expands the graph and grounds
ependencies than baselines.

ense terms riding along multiple

Talk Overview

1. What counts as a dependency, and why is tracing them hard?

   Much more than a list of names! Every dependency enters through some operations (generating, filtering), many hops upstream, documented in scattered and ambiguous sources.

LLM Agents are getting better at **information extraction**…

Information extraction (from a specific document to facts) is **no longer the primary challenge**

Key obstacles:
(1) What constitutes a dependency
(2) Resolving artifact references across inconsistent names, versions, models families, development stages, and repos

ependency graph

expands the graph and grounds
ependencies than baselines.

ense terms riding along multiple

Talk Overview

1. What counts as a dependency, and why is tracing them hard?

   Much more than a list of names! Every dependency enters through some operations (generating, filtering), many hops upstream, documented in scattered and ambiguous sources.

LLM Agents are getting better at **information extraction**…

Information extraction (from a specific document to facts) is **no longer the primary challenge**

Key obstacles:
(1) What constitutes a dependency
(2) Resolving artifact references across inconsistent names, versions, models families, development stages, and repos

Let's trace Olmo 3 by hand **to see the challenges**!

expands the graph and grounds
ependencies than baselines.

ense terms riding along multiple

Let's trace Olmo 3 by hand!

Let's trace Olmo 3 by hand!

Starting Point

Pre-training

Post-training

Evaluation

Let's trace Olmo 3 by hand!

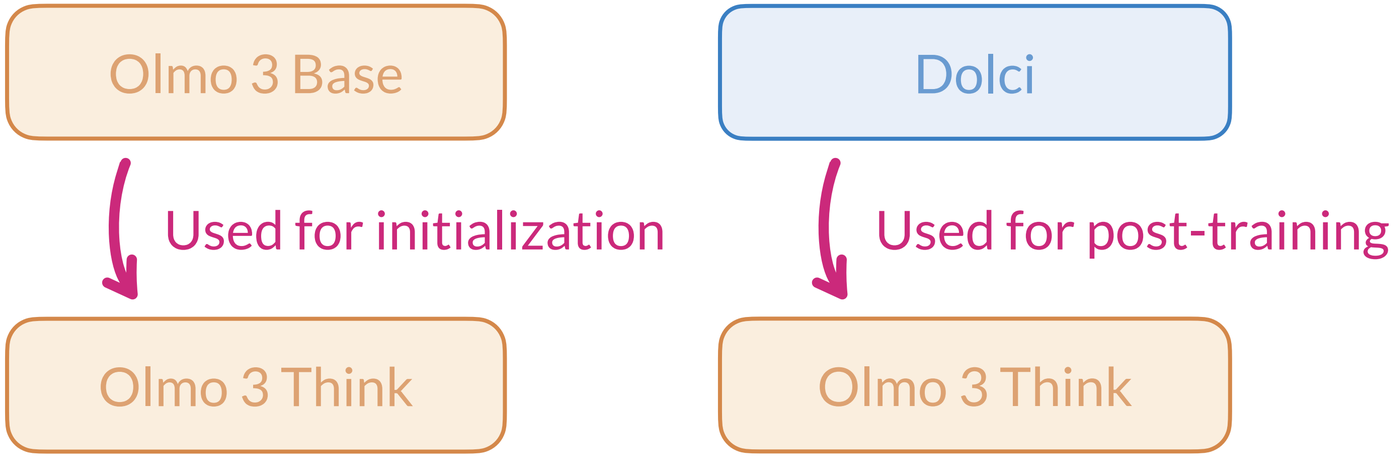
Starting Point | Pre-training | Post-training | Evaluation

Let's trace Olmo 3 by hand!

Starting Point | Pre-training | Post-training | Evaluation

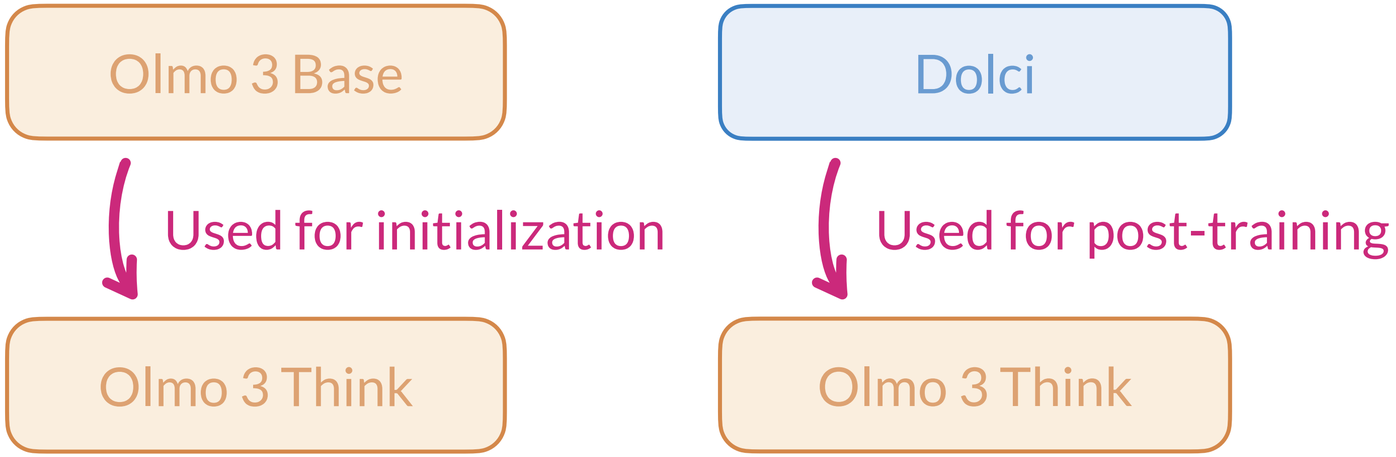
Let's trace Olmo 3 by hand!

Starting Point | Pre-training | Post-training | Evaluation



Let's trace Olmo 3 by hand!

Starting Point | Pre-training | Post-training | Evaluation



These are the most obvious and widely-discussed forms of dependencies. However …

Let's trace Olmo 3 by hand!

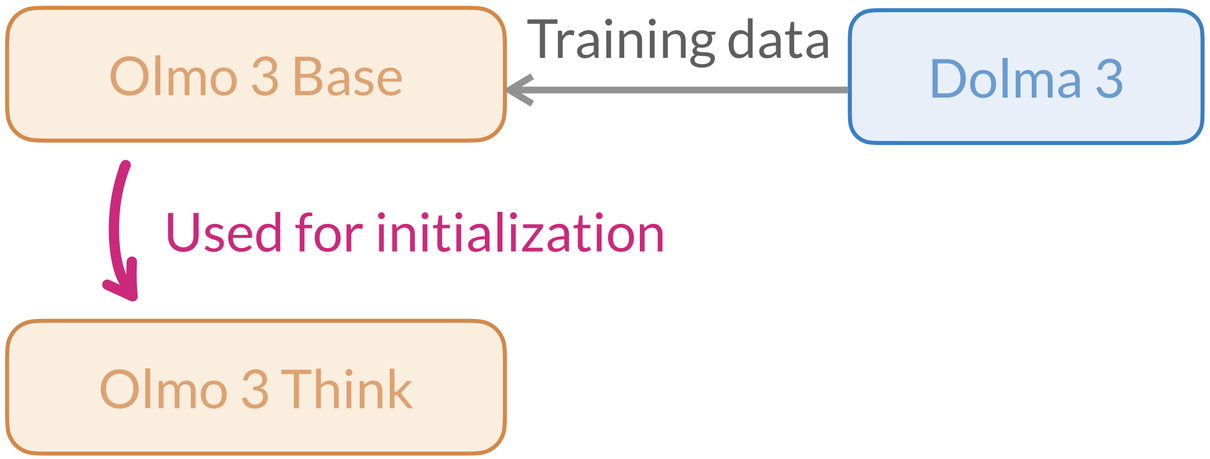
Starting Point | **Pre-training** | Post-training | Evaluation

Let's trace Olmo 3 by hand!

Starting Point | **Pre-training** | Post-training | Evaluation

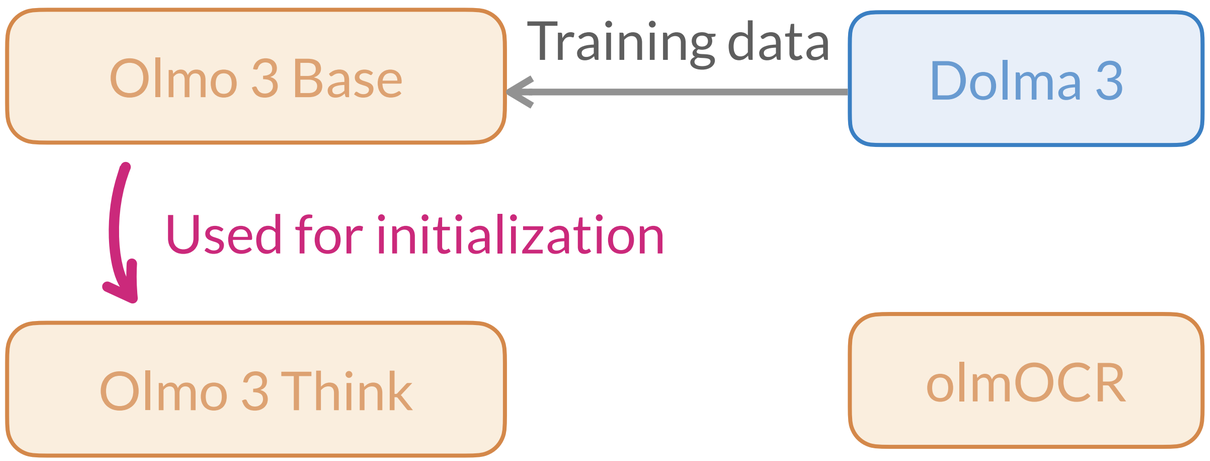
Let's trace Olmo 3 by hand!

Starting Point | **Pre-training** | Post-training | Evaluation



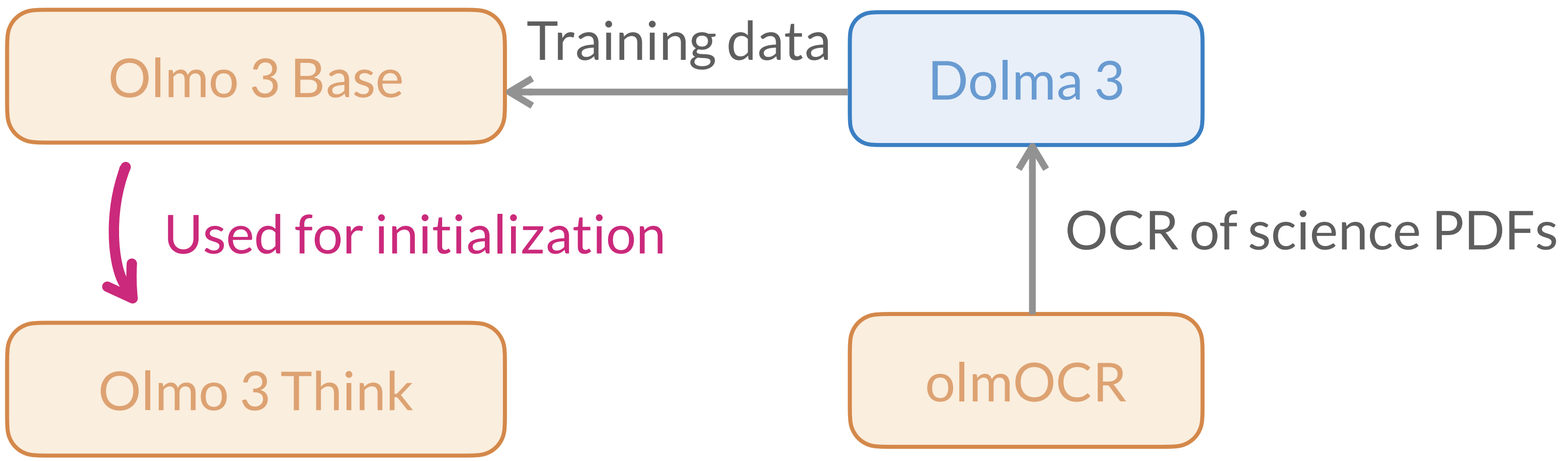
Let's trace Olmo 3 by hand!

Starting Point | **Pre-training** | Post-training | Evaluation



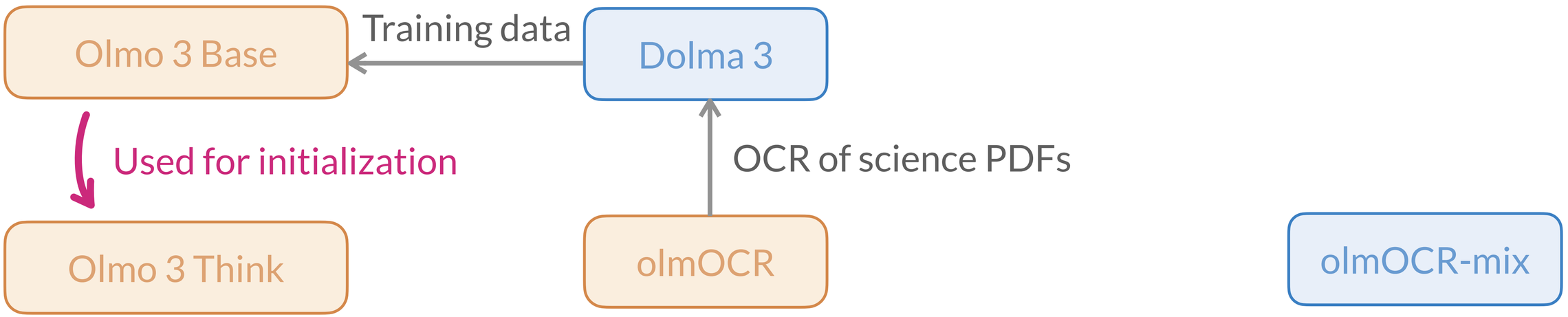
Let's trace Olmo 3 by hand!

Starting Point | **Pre-training** | Post-training | Evaluation



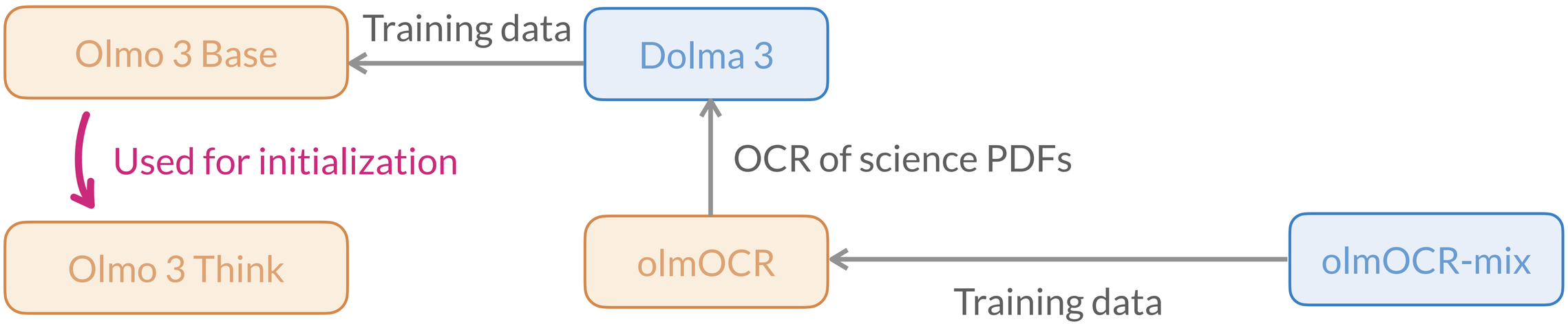
Let's trace Olmo 3 by hand!

Starting Point | **Pre-training** | Post-training | Evaluation



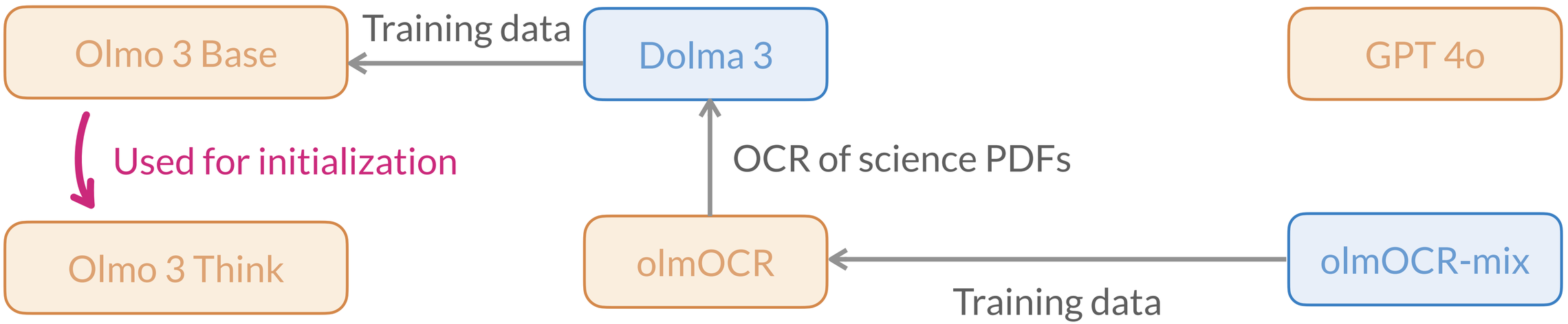
Let's trace Olmo 3 by hand!

Starting Point | **Pre-training** | Post-training | Evaluation



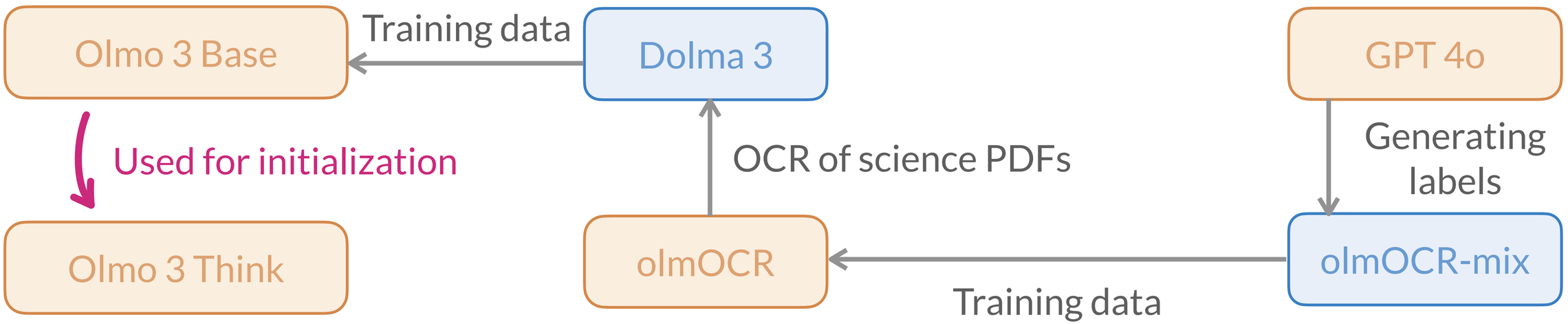
Let's trace Olmo 3 by hand!

Starting Point | **Pre-training** | Post-training | Evaluation



Let's trace Olmo 3 by hand!

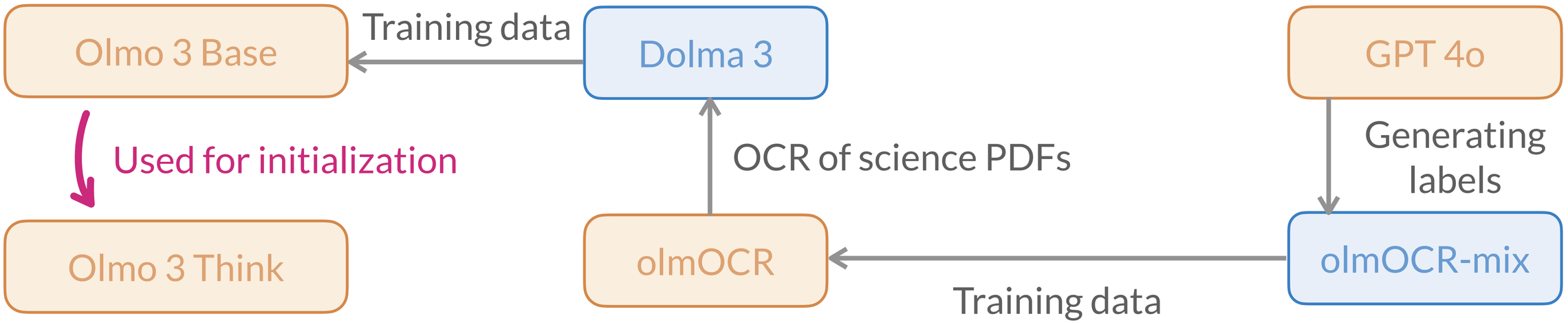
Starting Point | **Pre-training** | Post-training | Evaluation



Let's trace Olmo 3 by hand!

Starting Point | **Pre-training** | Post-training | Evaluation

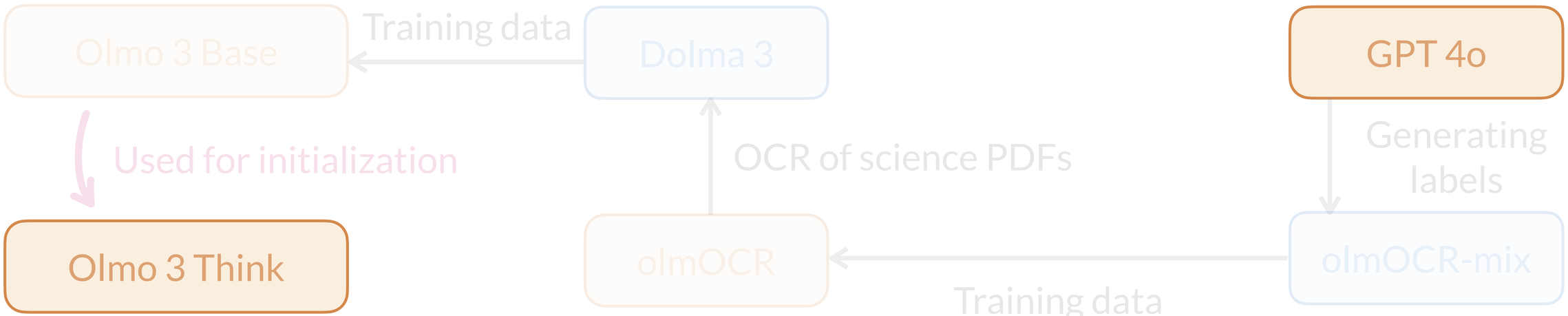
Dependencies are (1) **multi-hop** (2) **multi-type**



Let's trace Olmo 3 by hand!

Starting Point | **Pre-training** | Post-training | Evaluation

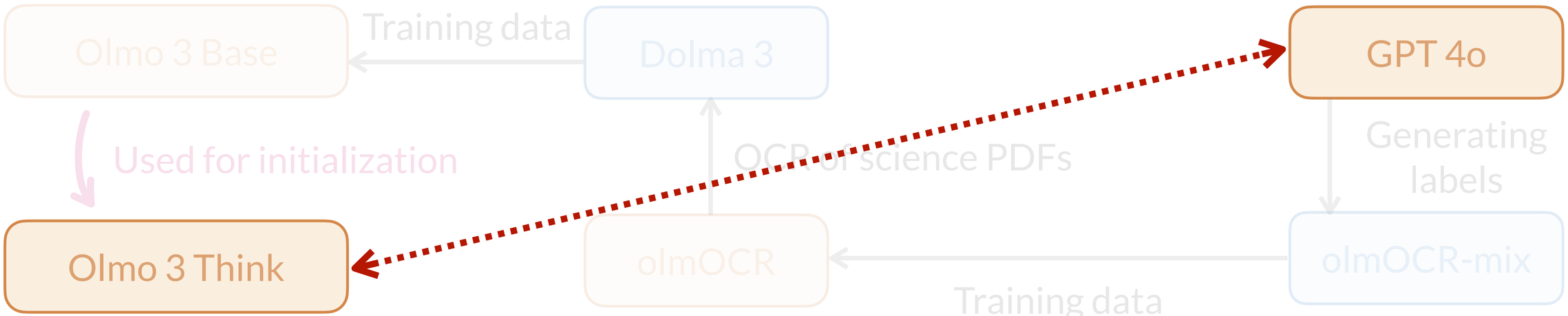
Dependencies are (1) **multi-hop** (2) **multi-type**



Let's trace Olmo 3 by hand!

Starting Point | **Pre-training** | Post-training | Evaluation

Dependencies are (1) **multi-hop** (2) **multi-type**



Let's trace Olmo 3 by hand!

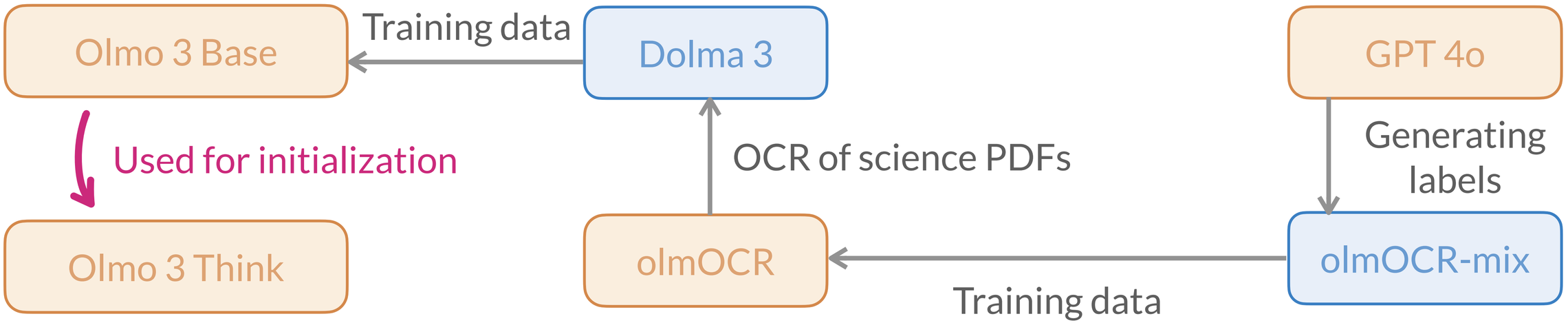
Starting Point | Pre-training | Post-training | Evaluation

Dependencies are (1) **multi-hop** (2) **multi-type**

Let's trace Olmo 3 by hand!

Starting Point | **Pre-training** | Post-training | Evaluation

Dependencies are (1) **multi-hop** (2) **multi-type**



Let's trace Olmo 3 by hand!

Starting Point | Pre-training | **Post-training** | Evaluation

Let's trace Olmo 3 by hand!

Starting Point | Pre-training | **Post-training** | Evaluation

To construct DOLCI THINK DPO, we compile a large pool of prompts covering a wide range of datasets and skills (see Table 19) and synthesize chosen and rejected responses to exhibit capability deltas. Following the delta-learning heuristic (Geng et al., 2025), for each prompt x, we simply decode a chosen completion y_c from one model (Qwen 3 32B, thinking) and a rejected completion y_r from an overall weaker model (Qwen 3 0.6B, thinking) to construct a consistent contrast.[32]

Let's trace Olmo 3 by hand!

Starting Point | Pre-training | **Post-training** | Evaluation

Olmo 3 32B SFT

↓ DPO

Olmo 3 32B DPO

To construct Dolci Think DPO, we compile a large pool of prompts covering a wide range of datasets and skills (see Table 19) and synthesize chosen and rejected responses to exhibit capability deltas. Following the delta-learning heuristic (Geng et al., 2025), for each prompt x, we simply decode a chosen completion y_c from one model (Qwen 3 32B, thinking) and a rejected completion y_r from an overall weaker model (Qwen 3 0.6B, thinking) to construct a consistent contrast.[32]

`github.com/allenai/open-instruct/blob/main/scripts/train/olmo3/32b_instruct_dpo.sh`

```
--use_slow_tokenizer False \
--mixer_list allenai/olmo-3-preference-mix-deltas_reasoning-scottmix-DECON-keyword-ftd-topic-ftd-dedup5_take2 1.0 \
--max_train_samples 200000 \
--skip_cache \
--zero_stage 3 \
--max_seq_length 8192 \
--per_device_train_batch_size 1 \
--gradient_accumulation_steps 1 \
```

Let's trace Olmo 3 by hand!

Starting Point | Pre-training | **Post-training** | Evaluation

To construct DOLCI THINK DPO, we compile a large pool of prompts covering a wide range of datasets and skills (see Table 19) and synthesize chosen and rejected responses to exhibit capability deltas. Following the delta-learning heuristic (Geng et al., 2025), for each prompt x, we simply decode a chosen completion y_c from one model (Qwen 3 32B, thinking) and a rejected completion y_r from an overall weaker model (Qwen 3 0.6B,

Dependencies are (1) **multi-hop** (2) **multi-type**

DPO

github.com/allenai/open-instruct/blob/main/scripts/train/olmo3/32b_instruct_dpo.sh

Olmo 3 32B DPO

```
--use_slow_tokenizer False \
--mixer_list allenai/olmo-3-preference-mix-deltas_reasoning-scottmix-DECON-keyword-ftd-topic-ftd-dedup5_take2 1.0 \
--max_train_samples 200000 \
--skip_cache \
--zero_stage 3 \
--max_seq_length 8192 \
--per_device_train_batch_size 1 \
--gradient_accumulation_steps 1 \
```

Let's trace Olmo 3 by hand!

Starting Point | Pre-training | **Post-training** | Evaluation

To construct DOLCI THINK DPO, we compile a large pool of prompts covering a wide range of datasets and skills (see Table 19) and synthesize chosen and rejected responses to exhibit capability deltas. Following the delta-learning heuristic (Geng et al., 2025), for each prompt x, we simply decode a chosen completion y_c from one model (Qwen 3 32B, thinking) and a rejected completion y_r from an overall weaker model (Qwen 3 0.6B,

Dependencies are (1) **multi-hop** (2) **multi-type** (3) **scattered across sources**

DPO

Olmo 3 32B DPO

github.com/allenai/open-instruct/blob/
main/scripts/train/olmo3/32b_instruct_dpo.sh

```
--use_slow_tokenizer False \
--mixer_list allenai/olmo-3-preference-mix-deltas_reasoning-scottmix-DECON-keyword-ftd-topic-ftd-dedup5_take2 1.0 \
--max_train_samples 200000 \
--skip_cache \
--zero_stage 3 \
--max_seq_length 8192 \
--per_device_train_batch_size 1 \
--gradient_accumulation_steps 1 \
```

Let's trace Olmo 3 by hand!

Starting Point | Pre-training | **Post-training** | Evaluation

Let's trace Olmo 3 by hand!

Starting Point | Pre-training | **Post-training** | Evaluation

`github.com/allenai/open-instruct/blob/main/scripts/train/olmo3/32b_instruct_dpo.sh`

Let's trace Olmo 3 by hand!

Starting Point | Pre-training | **Post-training** | Evaluation

`github.com/allenai/open-instruct/blob/main/scripts/train/olmo3/32b_instruct_dpo.sh`

```
11    DATASET_MIXER_LIST="
12    allenai/olmo-3-pref-mix-deltas-complement2-DECON-tpc-kwd-ch-dedup5-lbc100-grafmix-unbal 125000
13    allenai/dpo-yolo1-200k-gpt4.1-2w2s-maxdelta_reje-426124-rm-gemma3-kwd-ftd-ch-ftd-topic-ftd-dedup5-lbc100 125000
14    allenai/general_responses_dev_8maxturns_truncate-9fbef8-enrejected-kwd-ftd-cn-ftd-topic-filt-lb100 1250
```

Let's trace Olmo 3 by hand!

Starting Point | Pre-training | **Post-training** | Evaluation

`github.com/allenai/open-instruct/blob/main/scripts/train/olmo3/32b_instruct_dpo.sh`

```
11    DATASET_MIXER_LIST="
12    allenai/olmo-3-pref-mix-deltas-complement2-DECON-tpc-kwd-ch-dedup5-lbc100-grafmix-unbal 125000
13    allenai/dpo-yolo1-200k-gpt4.1-2w2s-maxdelta_reje-426124-rm-gemma3-kwd-ftd-ch-ftd-topic-ftd-dedup5-lbc100 125000
14    allenai/general_responses_dev_8maxturns_truncate-9fbef8-enrejected-kwd-ftd-cn-ftd-topic-filt-lb100 1250
```

Let's trace Olmo 3 by hand!

Starting Point | Pre-training | **Post-training** | Evaluation

`github.com/allenai/open-instruct/blob/main/scripts/train/olmo3/32b_instruct_dpo.sh`

```
11    DATASET_MIXER_LIST="
12    allenai/olmo-3-pref-mix-deltas-complement2-DECON-tpc-kwd-ch-dedup5-lbc100-grafmix-unbal 125000
13    allenai/dpo-yolo1-200k-gpt4.1-2w2s-maxdelta_reje-426124-rm-gemma3-kwd-ftd-ch-ftd-topic-ftd-dedup5-lbc100 125000
14    allenai/general_responses_dev_8maxturns_truncate-9fbef8-enrejected-kwd-ftd-cn-ftd-topic-filt-lb100 1250
```

GPT-4.1-2025-04-14?

Let's trace Olmo 3 by hand!

Starting Point | Pre-training | **Post-training** | Evaluation

`github.com/allenai/open-instruct/blob/main/scripts/train/olmo3/32b_instruct_dpo.sh`

```
11    DATASET_MIXER_LIST="
12    allenai/olmo-3-pref-mix-deltas-complement2-DECON-tpc-kwd-ch-dedup5-lbc100-grafmix-unbal 125000
13    allenai/dpo-yolo1-200k-gpt4.1-2w2s-maxdelta_reje-426124-rm-gemma3-kwd-ftd-ch-ftd-topic-ftd-dedup5-lbc100 125000
14    allenai/general_responses_dev_8maxturns_truncate-9fbef8-enrejected-kwd-ftd-cn-ftd-topic-filt-lb100 1250
```

GPT-4.1-2025-04-14?

Gemma3 1B? 4B? 12B?

Let's trace Olmo 3 by hand!

Starting Point | Pre-training | **Post-training** | Evaluation

`github.com/allenai/open-instruct/blob/main/scripts/train/olmo3/32b_instruct_dpo.sh`

```
11    DATASET_MIXER_LIST="
12    allenai/olmo-3-pref-mix-deltas-complement2-DECON-tpc-kwd-ch-dedup5-lbc100-grafmix-unbal 125000
13    allenai/dpo-yolo1-200k-gpt4.1-2w2s-maxdelta_reje-426124-rm-gemma3-kwd-ftd-ch-ftd-topic-ftd-dedup5-lbc100 125000
14    allenai/general_responses_dev_8maxturns_truncate-9fbef8-enrejected-kwd-ftd-cn-ftd-topic-filt-lb100 1250
```

What is 2w2s?

GPT-4.1-2025-04-14?

Gemma3 1B? 4B? 12B?

Let's trace Olmo 3 by hand!

Starting Point | Pre-training | **Post-training** | Evaluation

`github.com/allenai/open-instruct/blob/main/scripts/train/olmo3/32b_instruct_dpo.sh`

```
11    DATASET_MIXER_LIST="
12    allenai/olmo-3-pref-mix-deltas-complement2-DECON-tpc-kwd-ch-dedup5-lbc100-grafmix-unbal 125000
13    allenai/dpo-yolo1-200k-gpt4.1-2w2s-maxdelta_reje-426124-rm-gemma3-kwd-ftd-ch-ftd-topic-ftd-dedup5-lbc100 125000
14    allenai/general_responses_dev_8maxturns_truncate-9fbef8-enrejected-kwd-ftd-cn-ftd-topic-filt-lb100 1250
```

What is 2w2s?

GPT-4.1-2025-04-14?

Gemma3 1B? 4B? 12B?

1. Artifact identities are underspecified
2. Dependency roles are underspecified

Let's trace Olmo 3 by hand!

Starting Point | Pre-training | **Post-training** | Evaluation

`github.com/allenai/open-instruct/blob/main/scripts/train/olmo3/32b_instruct_dpo.sh`

```
11    DATASET_MIXER_LIST="
12    allenai/olmo-3-pref-mix-deltas-complement2-DECON-tpc-kwd-ch-dedup5-lbc100-grafmix-unbal 125000
13    allenai/dpo-yolo1-200k-gpt4.1-2w2s-maxdelta_reje-426124-rm-gemma3-kwd-ftd-ch-ftd-topic-ftd-dedup5-lbc100 125000
14    allenai/general_responses_dev_8maxturns_truncate-9fbef8-enrejected-kwd-ftd-cn-ftd-topic-filt-lb100 1250
```

What is 2w2s?

GPT-4.1-2025-04-14?

Gemma3 1B? 4B? 12B?

1. Artifact identities are underspecified
2. Dependency roles are underspecified

This will become a big issue when we are trying to merge the nodes and edges extracted from different sources.

Let's trace Olmo 3 by hand!

Starting Point | Pre-training | **Post-training** | Evaluation

github.com/allenai/open-instruct/blob/main/scripts/train/olmo3/32b_instruct_dpo.sh

What is 2w2s?

Dependencies are (1) **multi-hop** (2) **multi-type** (3) **scattered across sources**

GPT-4.1-2025-04-14?

Gemma3 1B? 4B? 12B?

1. Artifact identities are underspecified
2. Dependency roles are underspecified

This will become a big issue when we are trying to merge the nodes and edges extracted from different sources.

Let's trace Olmo 3 by hand!

Starting Point | Pre-training | **Post-training** | Evaluation

github.com/allenai/open-instruct/blob/main/scripts/train/olmo3/32b_instruct_dpo.sh

What is 2w2s?

Dependencies are (1) **multi-hop** (2) **multi-type** (3) **scattered across sources** (4) **underspecified**

GPT-4.1-2025-04-14?

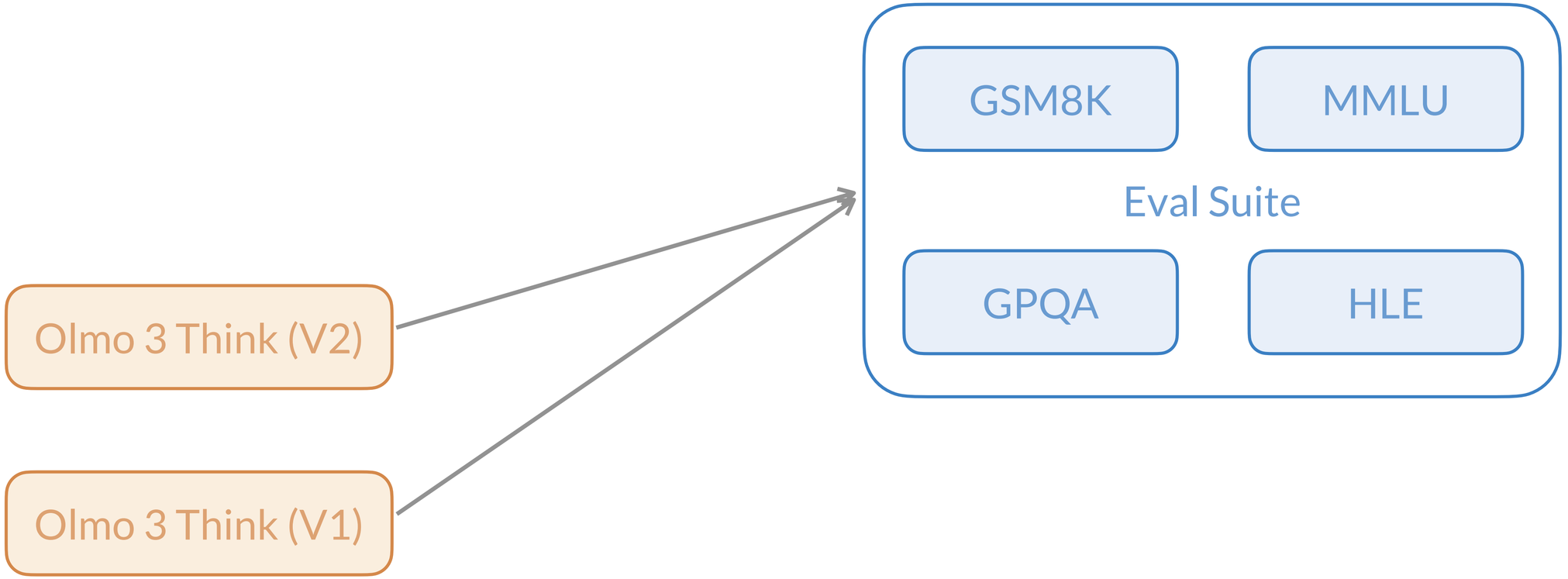
Gemma3 1B? 4B? 12B?

1. Artifact identities are underspecified
2. Dependency roles are underspecified

This will become a big issue when we are trying to merge the nodes and edges extracted from different sources.

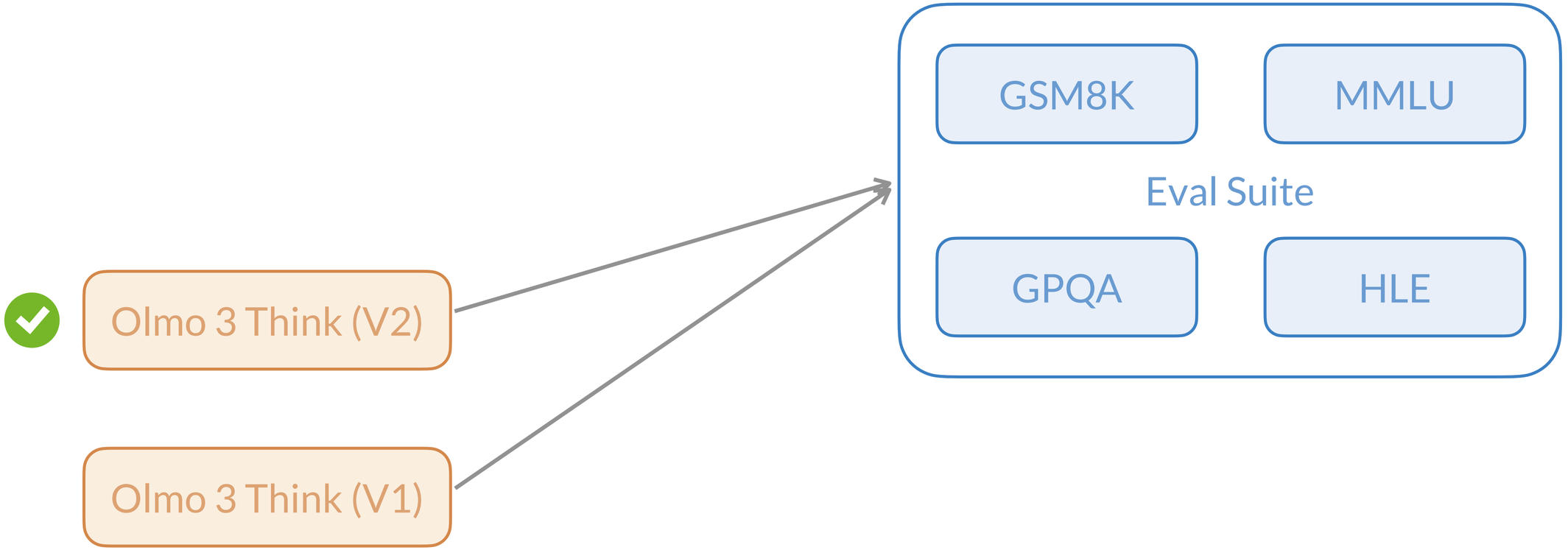
Let's trace Olmo 3 by hand!

Starting Point | Pre-training | Post-training | **Evaluation**



Let's trace Olmo 3 by hand!

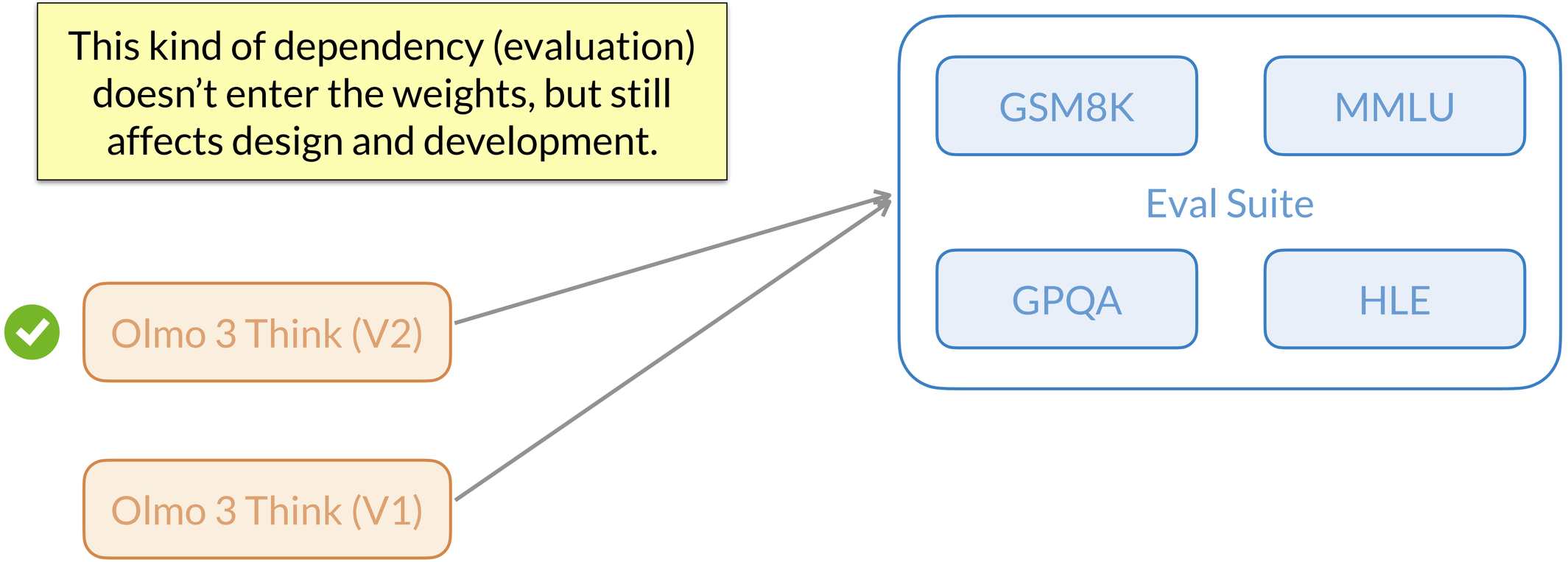
Starting Point | Pre-training | Post-training | **Evaluation**



Let's trace Olmo 3 by hand!

Starting Point | Pre-training | Post-training | **Evaluation**

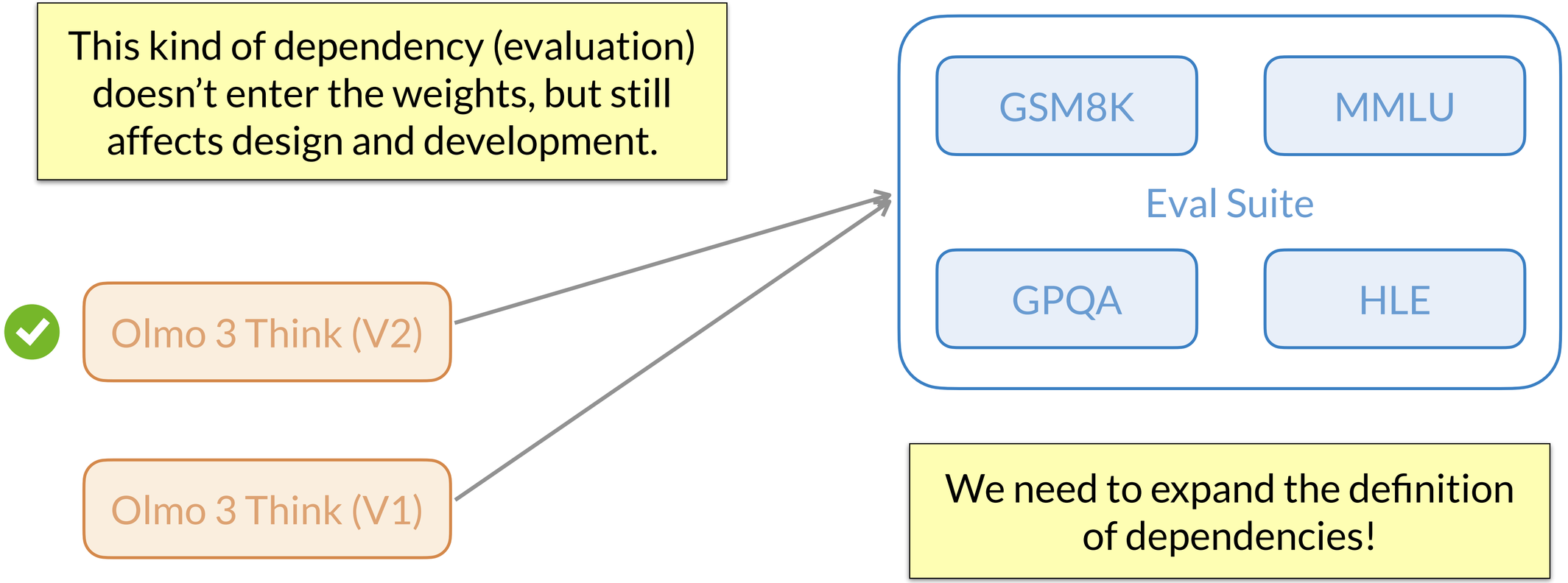
This kind of dependency (evaluation) doesn't enter the weights, but still affects design and development.



Let's trace Olmo 3 by hand!

Starting Point | Pre-training | Post-training | **Evaluation**

This kind of dependency (evaluation) doesn't enter the weights, but still affects design and development.



We need to expand the definition of dependencies!

Let's trace Olmo 3 by hand!

Starting Point | Pre-training | Post-training | **Evaluation**

This kind of dependency (evaluation) doesn't enter the weights, but still affects design and development

GSM8K

MMLU

Dependencies are (1) **multi-hop** (2) **multi-type** (3) **scattered across sources** (4) **underspecified**

Olmo 3 Think (V2)

Olmo 3 Think (V1)

We need to expand the definition of dependencies!

Let's trace Olmo 3 by hand!

Starting Point | Pre-training | Post-training | Evaluation

This kind of dependency (evaluation) doesn't enter the weights, but still affects design and development

GSM8K

MMLU

Dependencies are (1) **multi-hop** (2) **multi-type** (3) **scattered across sources** (4) **underspecified** (5) **inherently ambiguous to define**

Olmo 3 Think (V2)

Olmo 3 Think (V1)

We need to expand the definition of dependencies!

Talk Overview

1. What counts as a dependency, and why is tracing them hard?

   Much more than a list of names! Every dependency enters through some operations (generating, filtering), many hops upstream, documented in scattered and ambiguous sources.

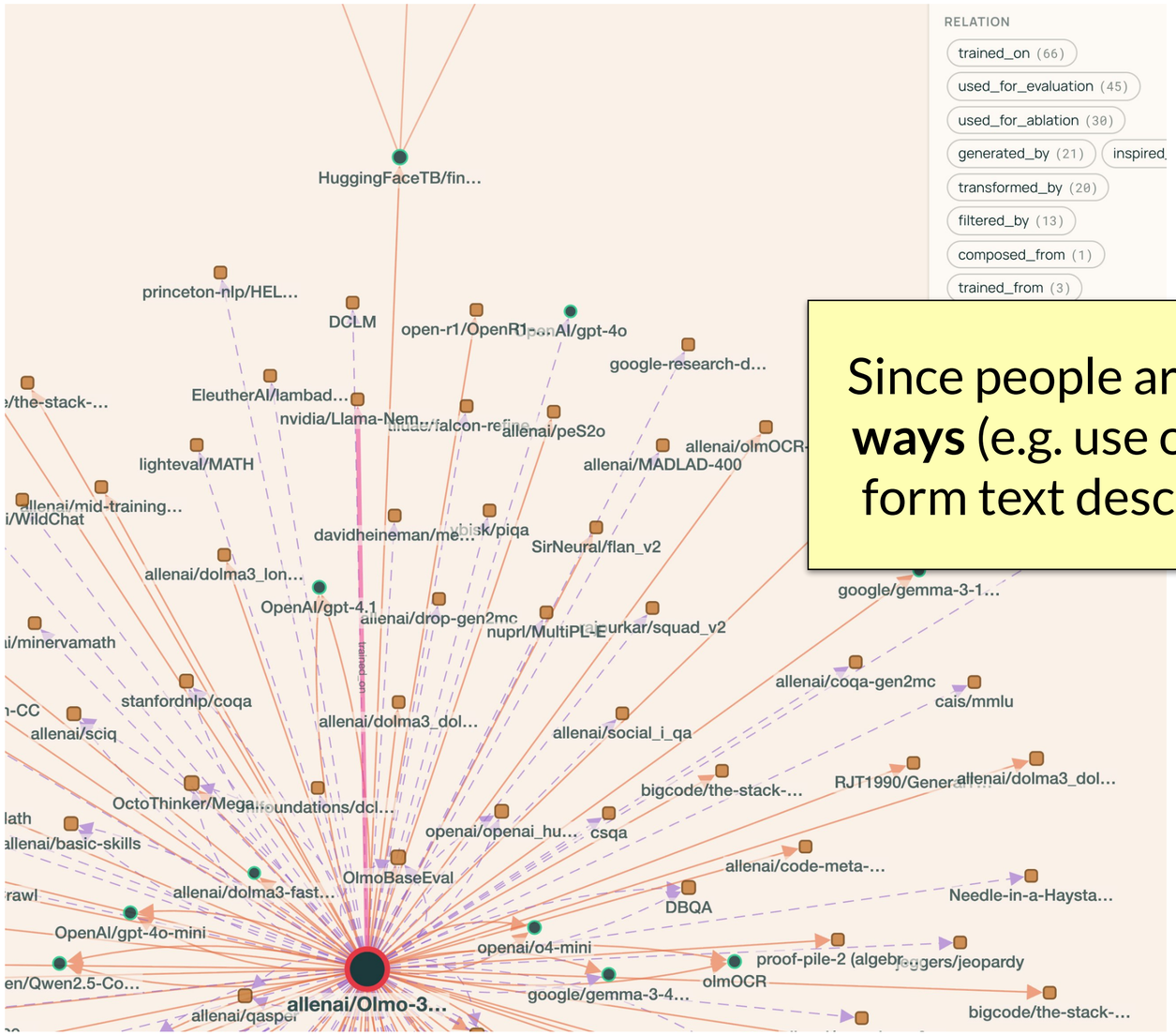
2. Can we *automatically* and *reliably* recover the dependency graph from public artifacts?

   Yes! ModSleuth, an agentic system we developed, recursively expands the graph and grounds every edge in source evidence—recovering 3x more verified dependencies than baselines.

3. What do the recovered graphs reveal?

   Plenty that no single page shows: hidden upstream models, license terms riding along multiple hops, benchmarks on both sides of training and eval, etc.

ModSleuth: Reconstructing dependency graphs



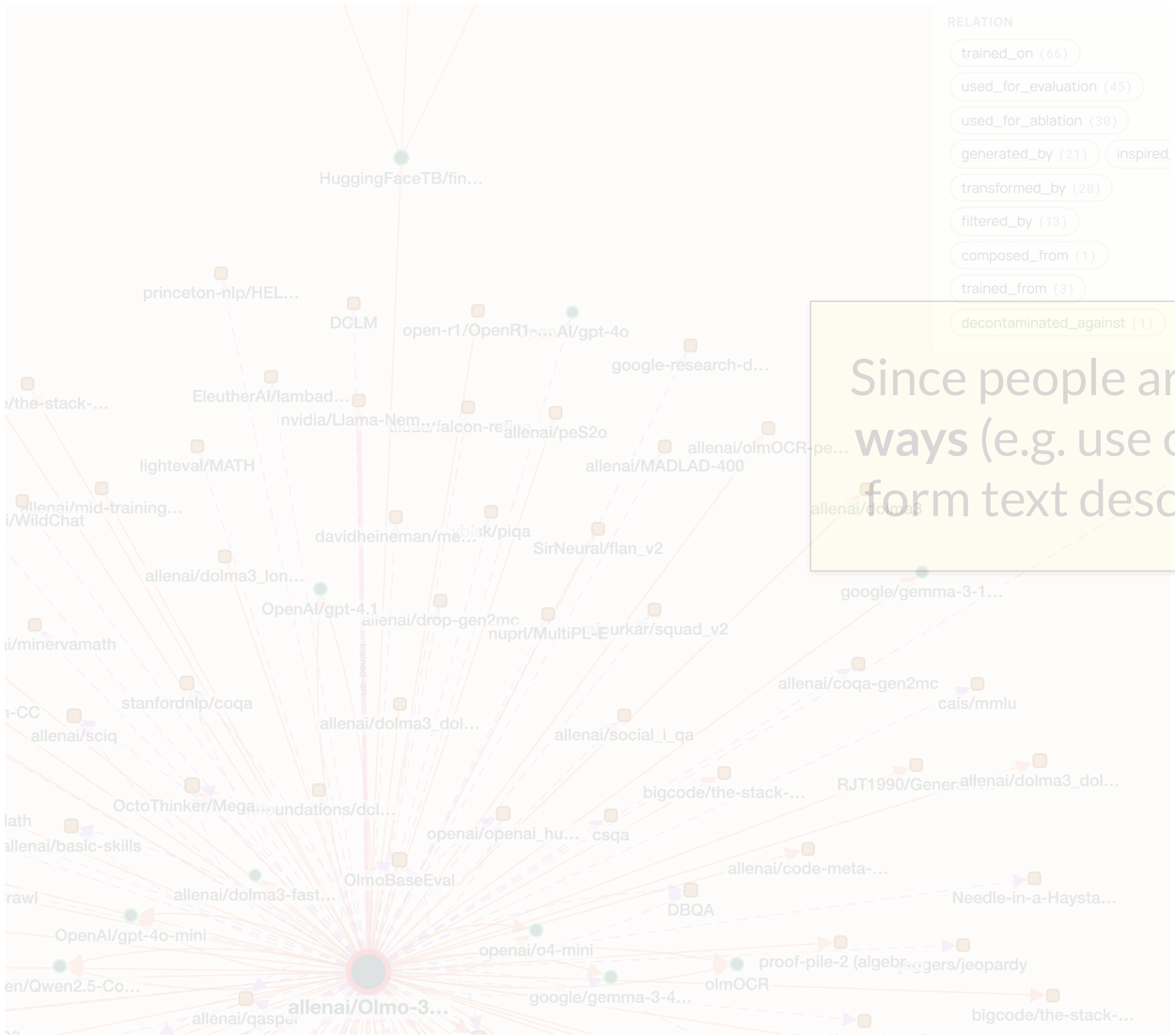
Since people are using models in **more and more creative ways** (e.g. use olmOCR to process PDFs), we ask for free-form text description for a dependency to preserve info.

circle = model square = dataset

arrow = one documented dependency

Target Olmo 3 at the bottom centre; depth grows outward.

ModSleuth: Reconstructing dependency graphs



Since people are using models in **more and more creative ways** (e.g. use olmOCR to process PDFs), we ask for free-form text description for a dependency to preserve info.

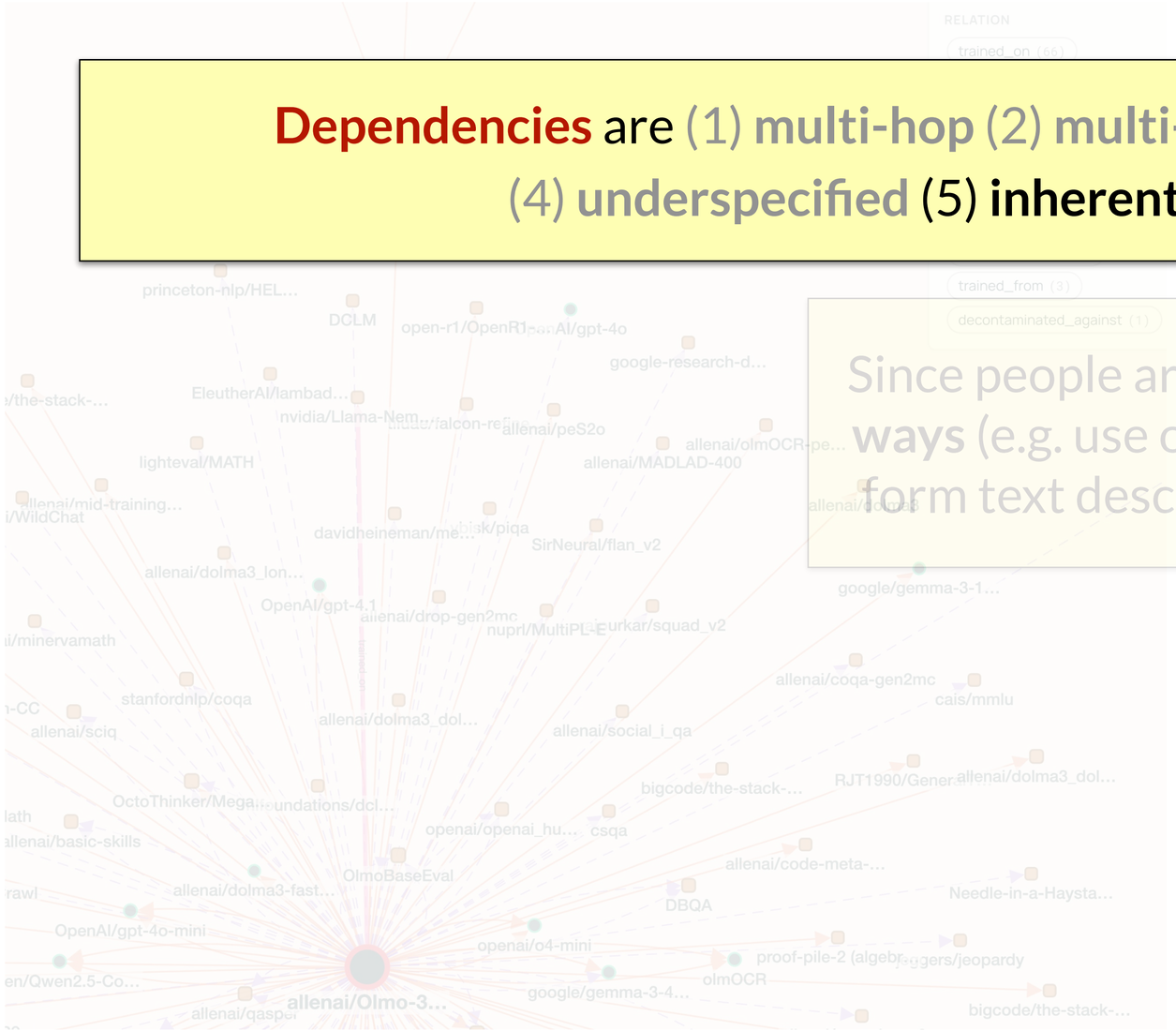
circle = model square = dataset

arrow = one documented dependency

Target Olmo 3 at the bottom centre; depth grows outward.

ModSleuth: Reconstructing dependency graphs

Dependencies are (1) **multi-hop** (2) **multi-type** (3) **scattered across sources**
(4) **underspecified** (5) **inherently ambiguous to define**



Since people are using models in **more and more creative ways** (e.g. use olmOCR to process PDFs), we ask for free-form text description for a dependency to preserve info.

circle = model square = dataset

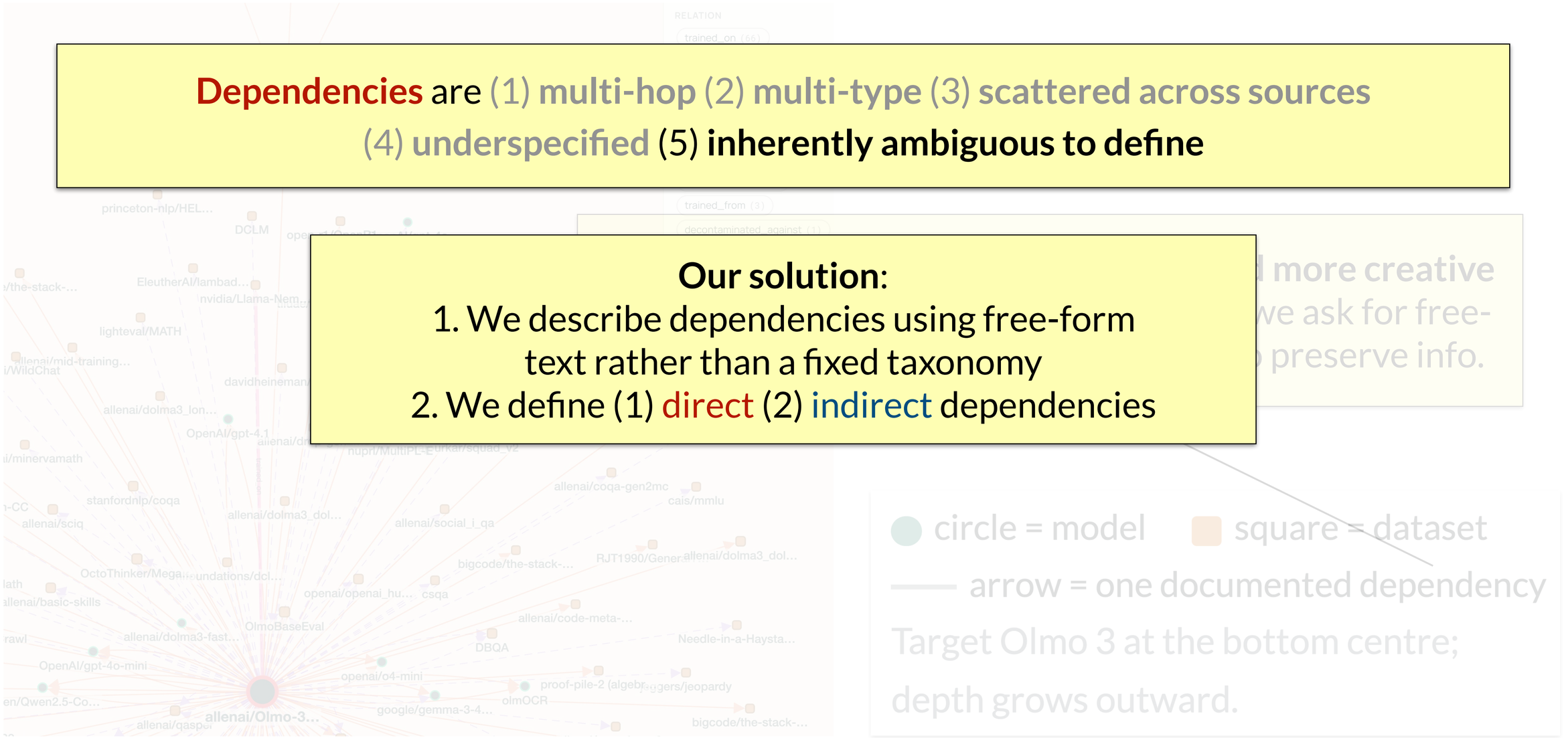
arrow = one documented dependency

Target Olmo 3 at the bottom centre; depth grows outward.

ModSleuth: Reconstructing dependency graphs

Dependencies are (1) **multi-hop** (2) **multi-type** (3) **scattered across sources**
(4) **underspecified** (5) **inherently ambiguous to define**

Our solution:
1. We describe dependencies using free-form text rather than a fixed taxonomy
2. We define (1) direct (2) indirect dependencies



ModSleuth: Reconstructing dependency graphs

Dependencies are (1) **multi-hop** (2) **multi-type** (3) **scattered across sources**
(4) **underspecified** (5) **inherently ambiguous to define**

Our solution:
1. We describe dependencies using free-form text rather than a fixed taxonomy
2. We define (1) direct (2) indirect dependencies

(1) Direct: upstream artifact that affects the target model's training or weights
(2) Indirect: upstream artifact that does not directly enter training, but nevertheless substantially influences development decisions

Design of ModSleuth

Design of ModSleuth

Let’s trace Olmo-3-7B with ModSleuth!

Dependencies are (1) **multi-hop** (2) **multi-type** (3) **scattered across sources**
(4) **underspecified** (5) ~~**inherently ambiguous to define**~~

Let’s trace Olmo-3-7B with ModSleuth!

Dependencies are (1) **multi-hop** (2) **multi-type** (3) **scattered across sources**
(4) **underspecified** (5) ~~inherently ambiguous to define~~

1 Source
Gathering

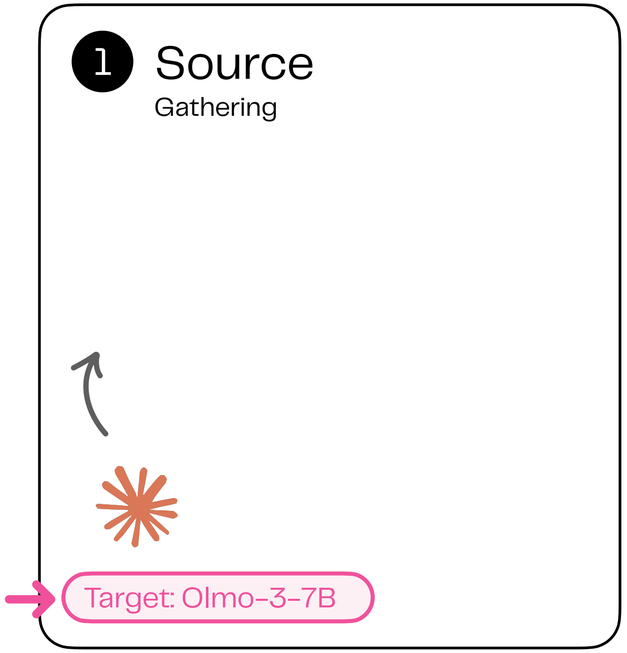
Let’s trace Olmo-3-7B with ModSleuth!

Dependencies are (1) **multi-hop** (2) **multi-type** (3) **scattered across sources**
(4) **underspecified** (5) ~~**inherently ambiguous to define**~~

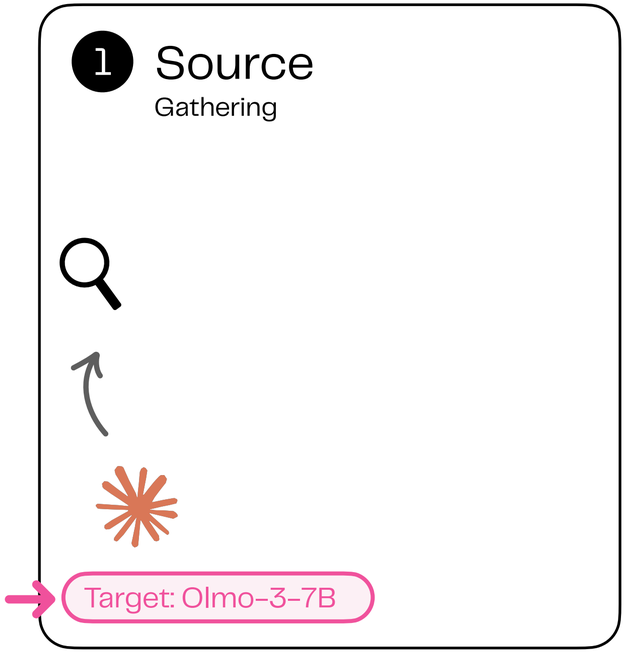
Let’s trace Olmo-3-7B with ModSleuth!

Dependencies are (1) **multi-hop** (2) **multi-type** (3) **scattered across sources**
(4) **underspecified** (5) ~~**inherently ambiguous to define**~~

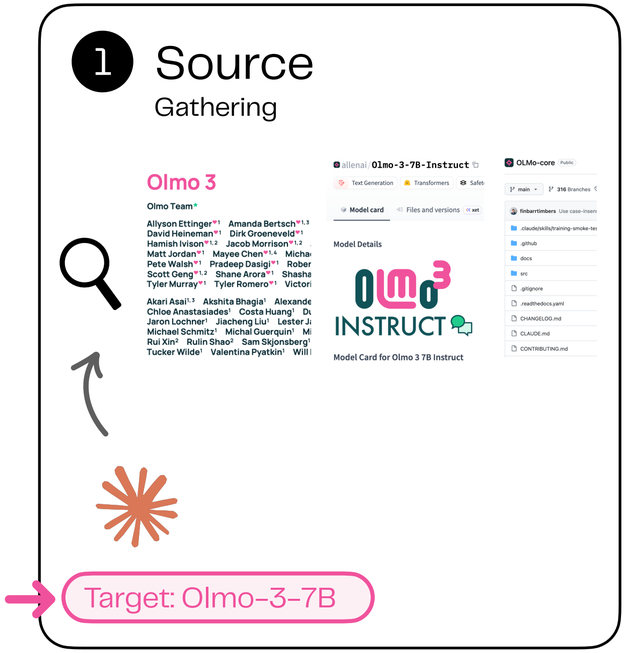
Dependencies are (1) **multi-hop** (2) **multi-type** (3) **scattered across sources**
(4) **underspecified** (5) ~~**inherently ambiguous to define**~~



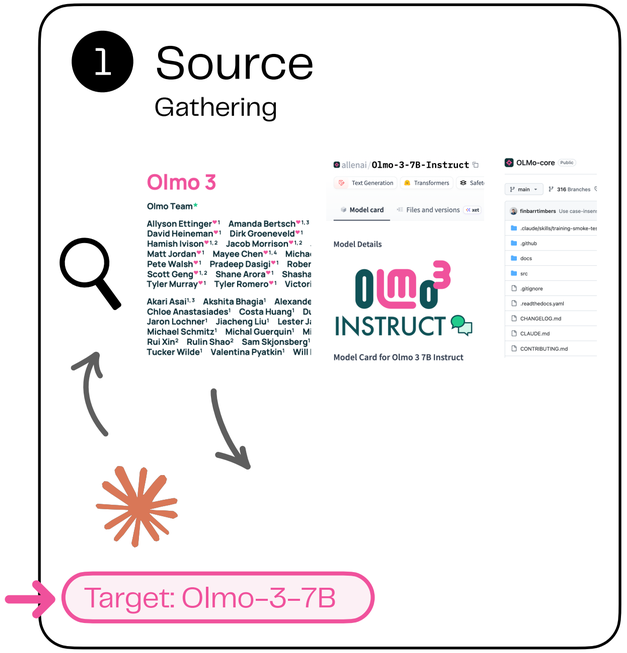
Dependencies are (1) **multi-hop** (2) **multi-type** (3) **scattered across sources** (4) **underspecified** (5) ~~**inherently ambiguous to define**~~



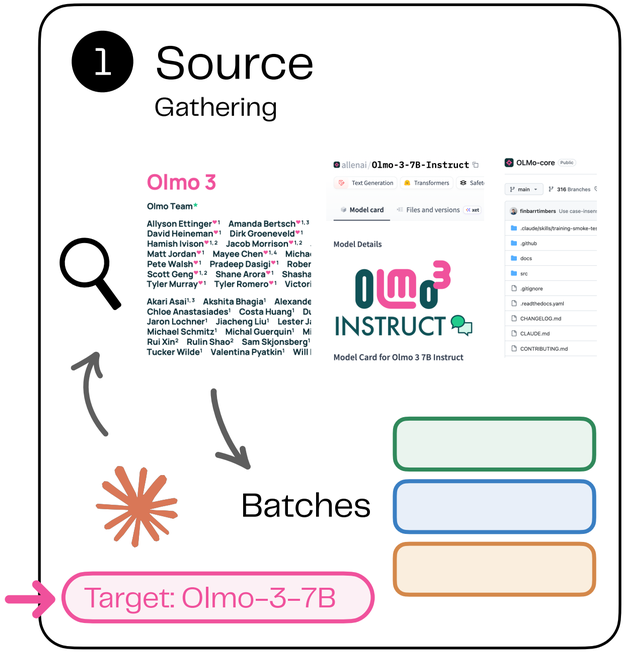
Dependencies are (1) **multi-hop** (2) **multi-type** (3) **scattered across sources**
(4) **underspecified** (5) ~~**inherently ambiguous to define**~~



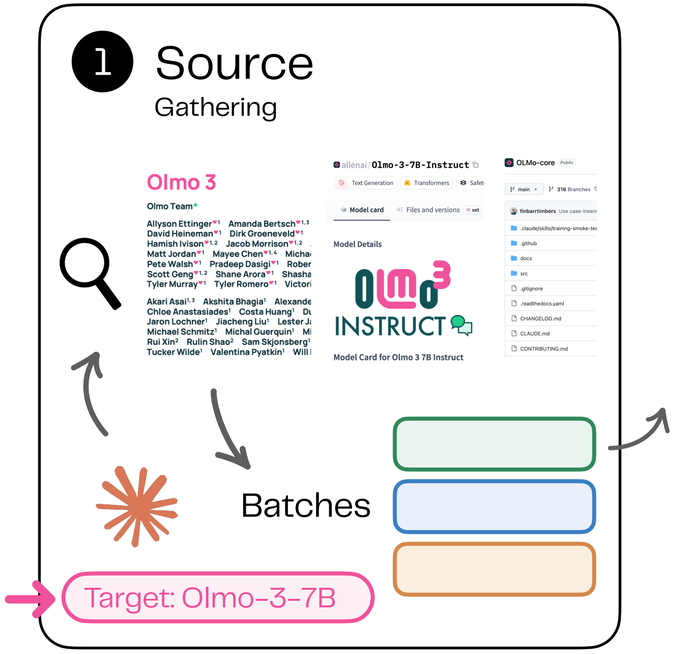
Dependencies are (1) **multi-hop** (2) **multi-type** (3) **scattered across sources**
(4) **underspecified** (5) ~~**inherently ambiguous to define**~~



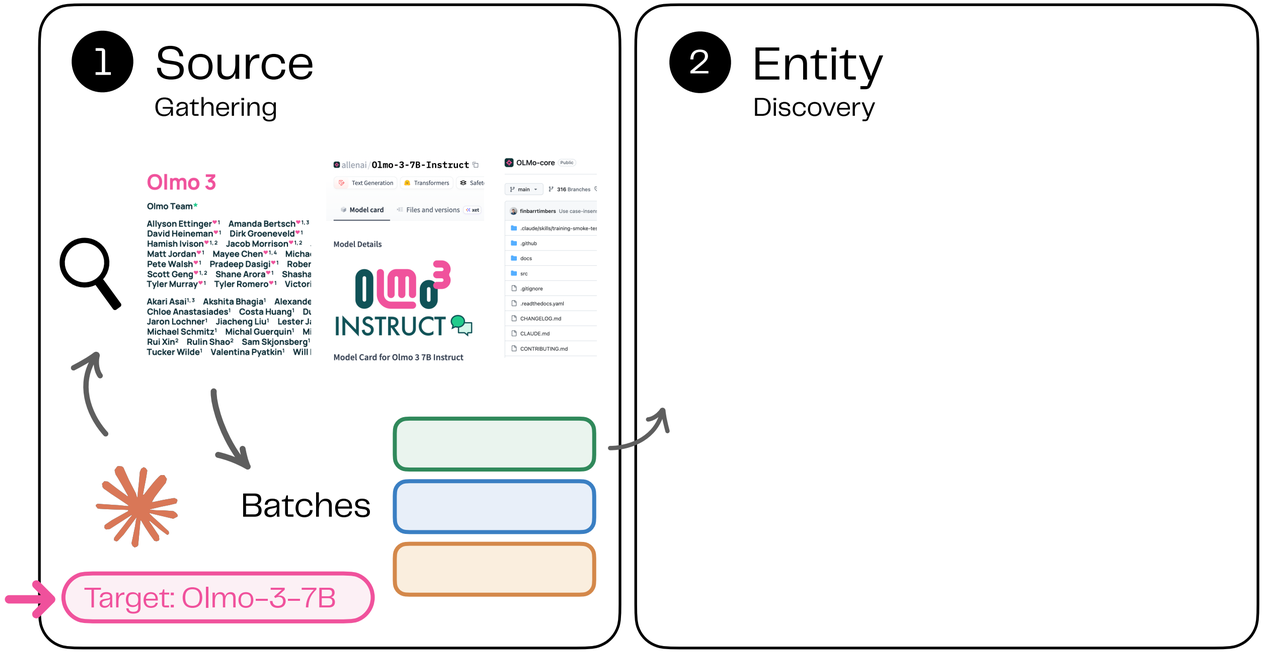
Dependencies are (1) **multi-hop** (2) **multi-type** (3) **scattered across sources**
(4) **underspecified** (5) ~~**inherently ambiguous to define**~~



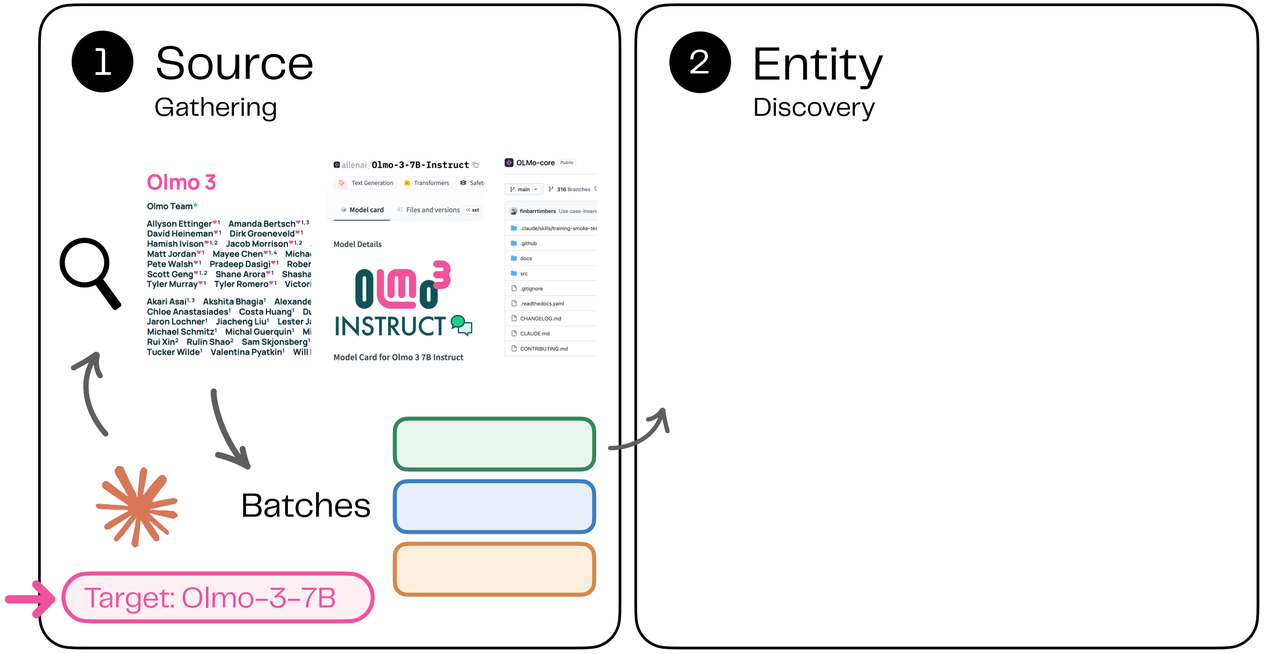
Dependencies are (1) **multi-hop** (2) **multi-type** (3) **scattered across sources**
(4) **underspecified** (5) ~~**inherently ambiguous to define**~~



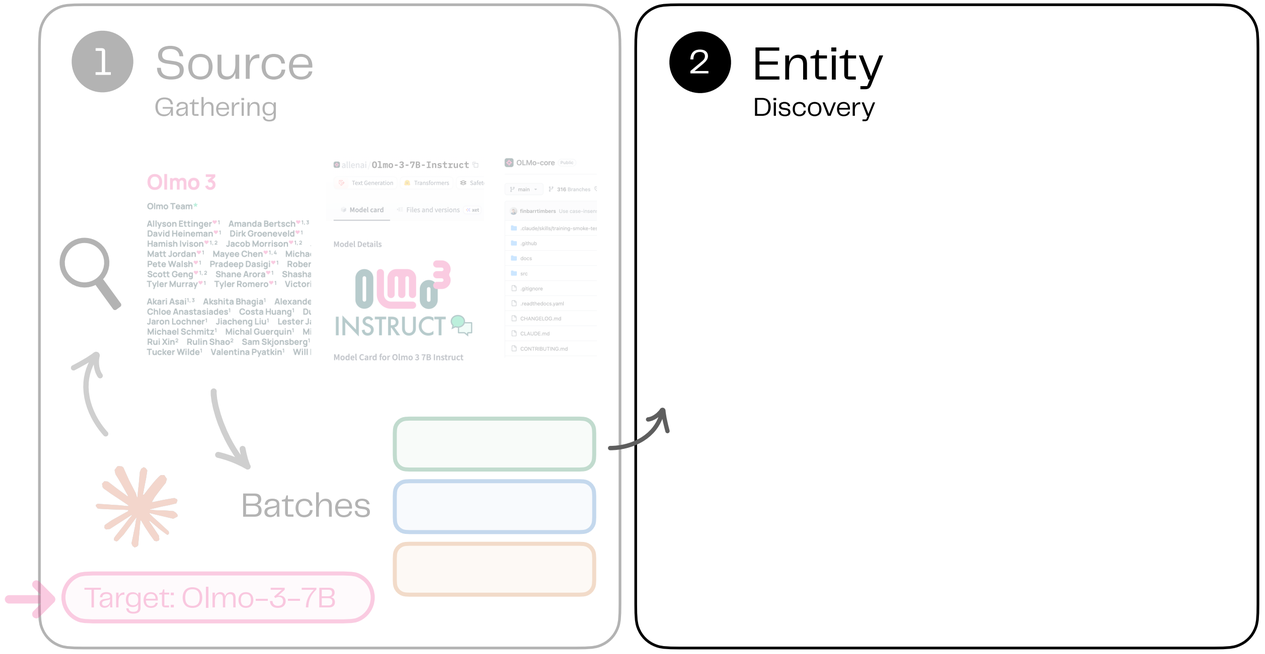
Dependencies are (1) **multi-hop** (2) **multi-type** (3) **scattered across sources** (4) **underspecified** (5) ~~**inherently ambiguous to define**~~



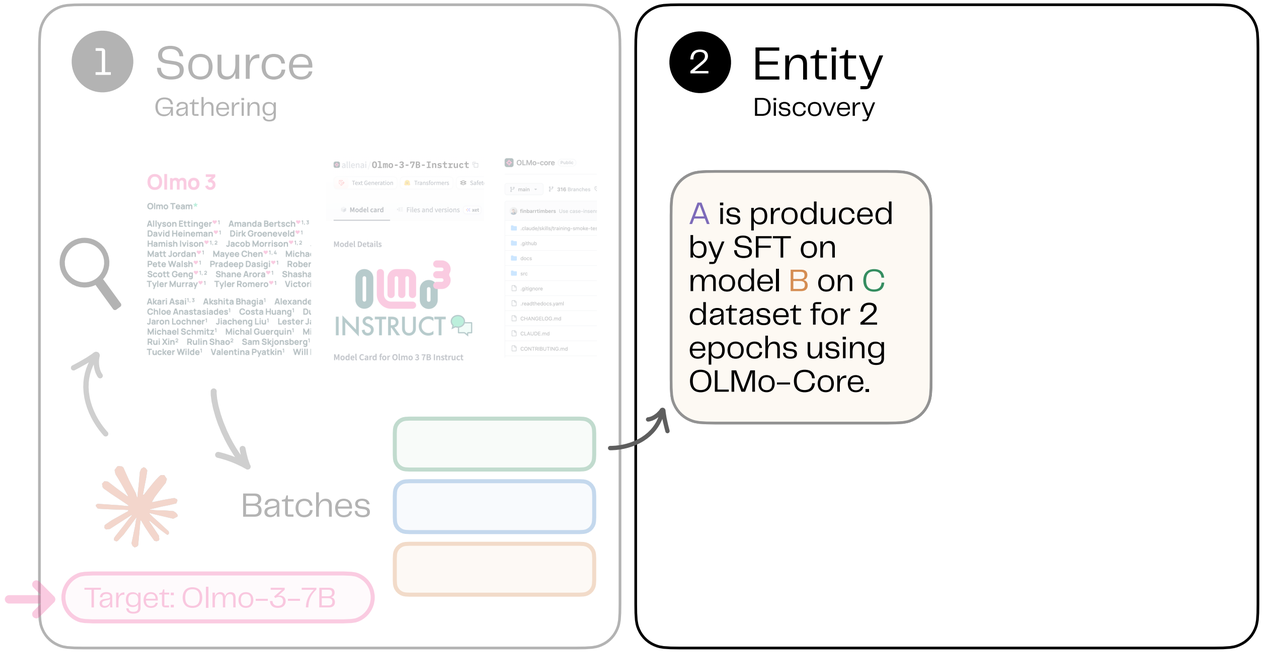
Dependencies are (1) **multi-hop** (2) **multi-type** (3) ~~**scattered across sources**~~ (4) **underspecified** (5) ~~inherently ambiguous to define~~



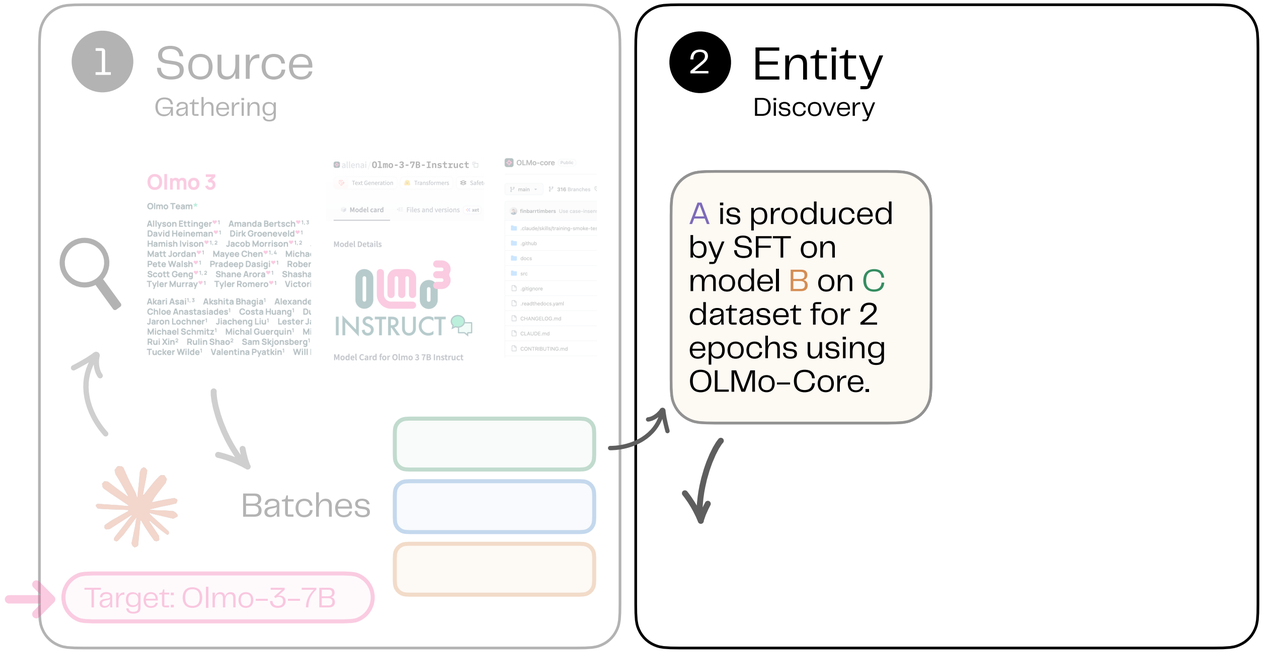
Dependencies are (1) **multi-hop** (2) **multi-type** (3) ~~scattered across sources~~ (4) **underspecified** (5) ~~inherently ambiguous to define~~



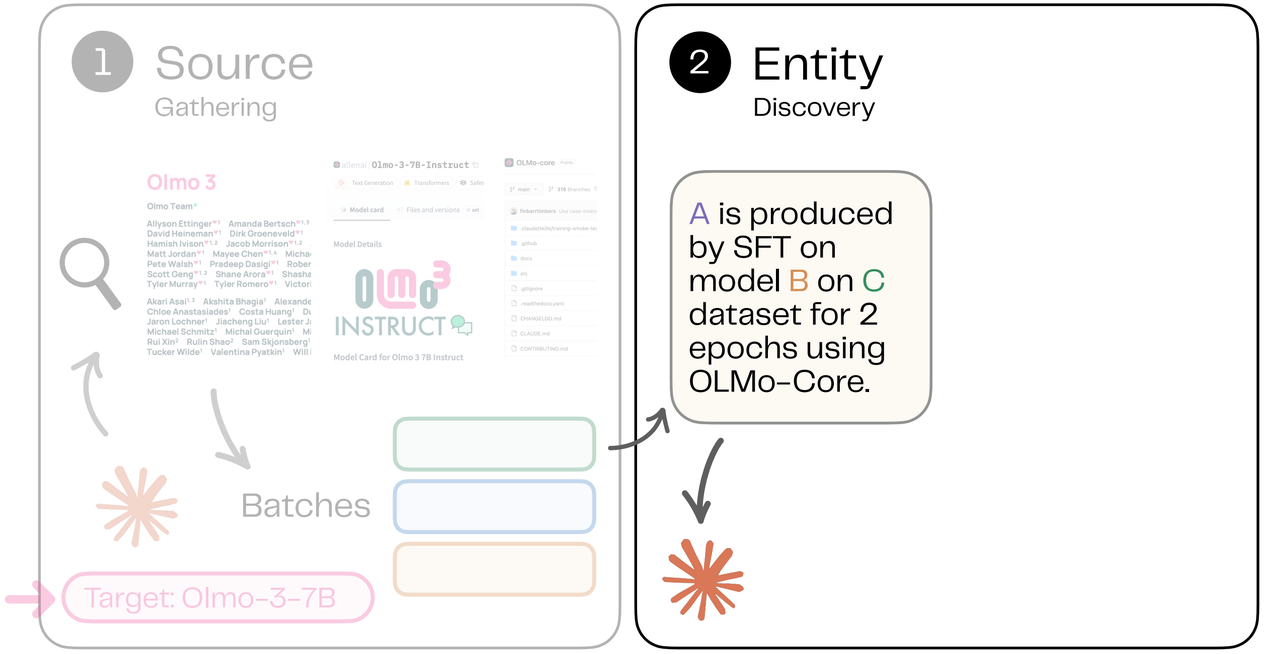
Dependencies are (1) **multi-hop** (2) **multi-type** (3) ~~scattered across sources~~ (4) **underspecified** (5) ~~inherently ambiguous to define~~



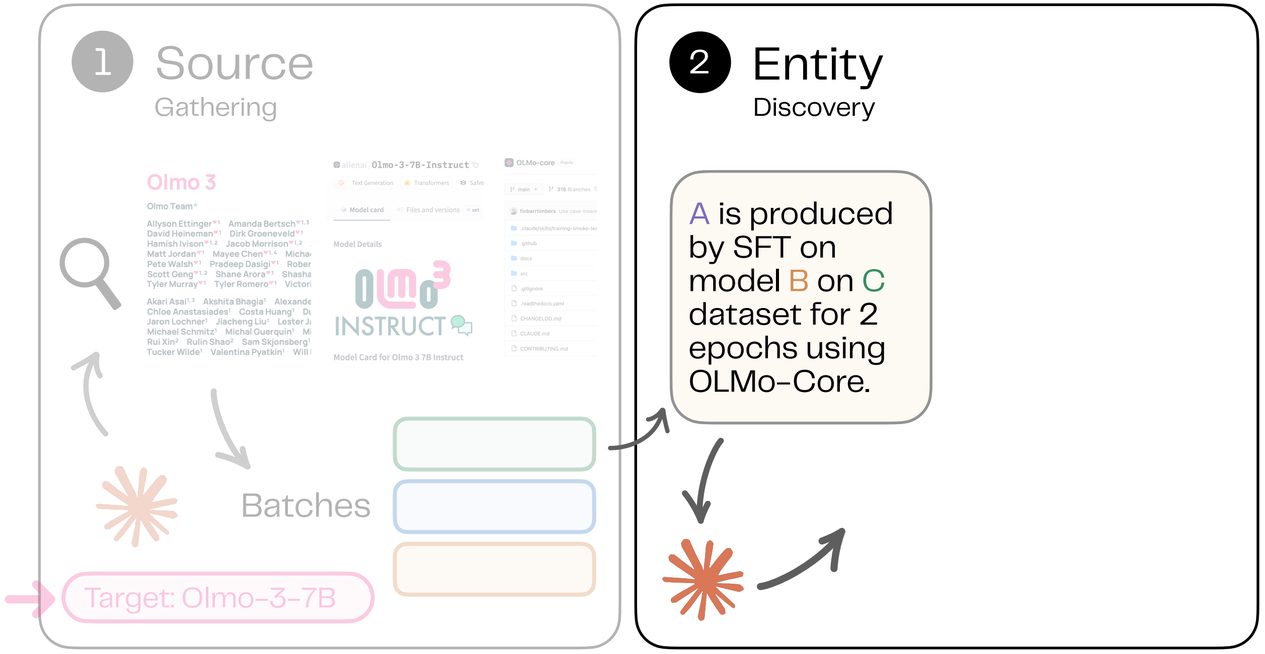
Dependencies are (1) **multi-hop** (2) **multi-type** (3) ~~scattered across sources~~ (4) **underspecified** (5) ~~inherently ambiguous to define~~



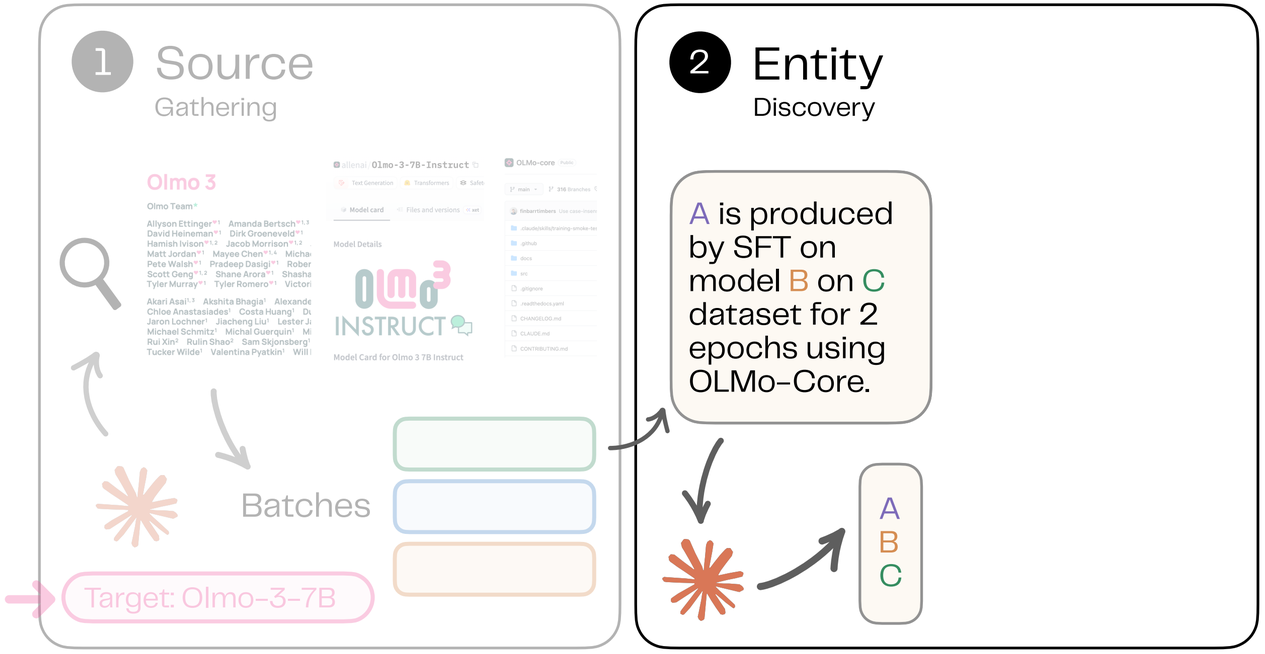
Dependencies are (1) **multi-hop** (2) **multi-type** (3) ~~scattered across sources~~ (4) **underspecified** (5) ~~inherently ambiguous to define~~



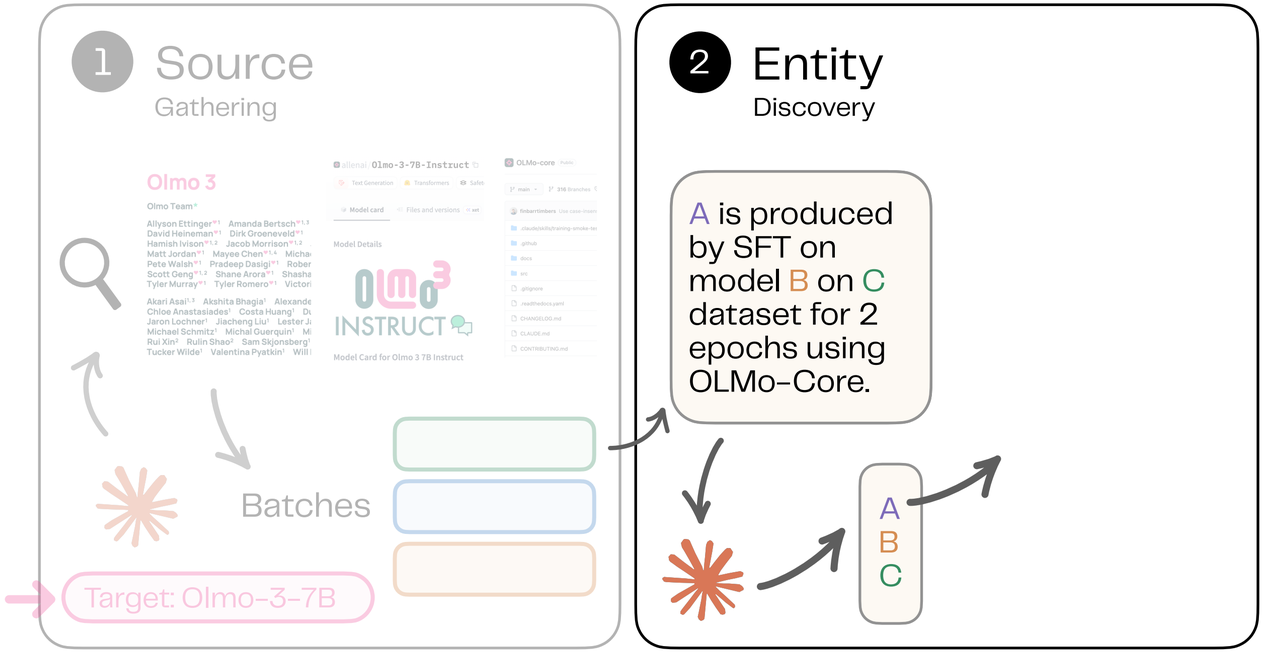
Dependencies are (1) **multi-hop** (2) **multi-type** (3) ~~scattered across sources~~ (4) **underspecified** (5) ~~inherently ambiguous to define~~



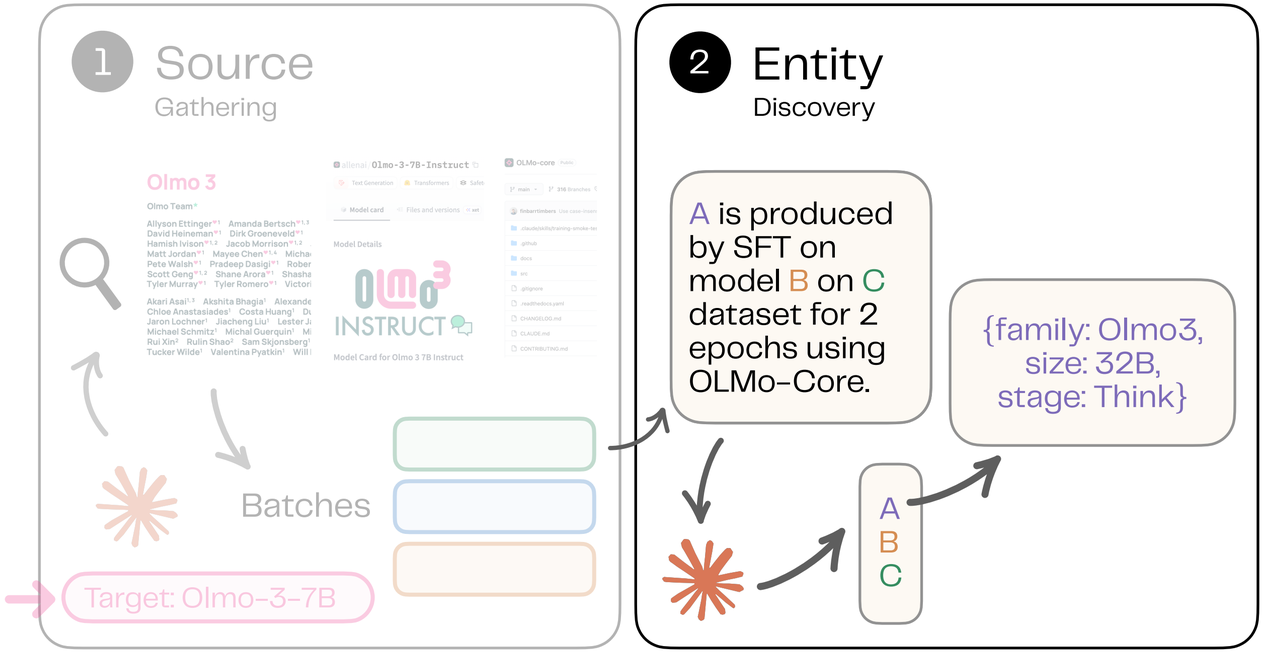
Dependencies are (1) **multi-hop** (2) **multi-type** (3) ~~scattered across sources~~ (4) **underspecified** (5) ~~inherently ambiguous to define~~



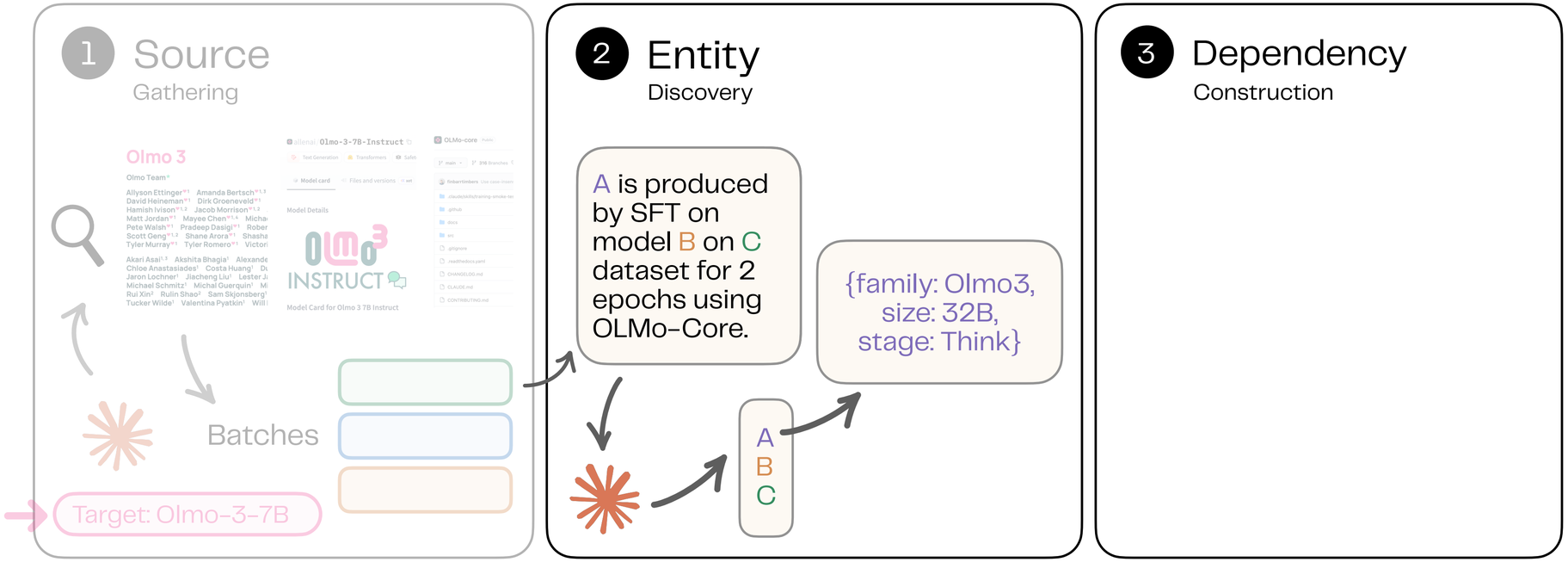
Dependencies are (1) **multi-hop** (2) **multi-type** (3) ~~scattered across sources~~ (4) **underspecified** (5) ~~inherently ambiguous to define~~



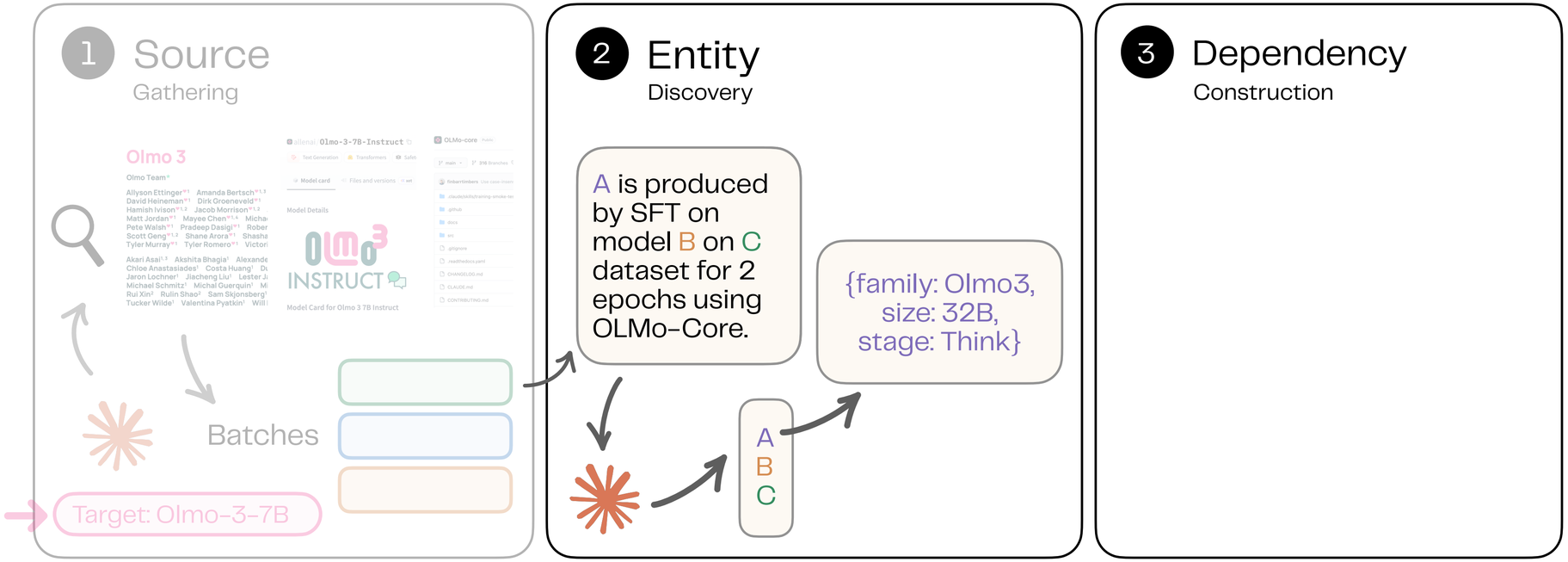
Dependencies are (1) **multi-hop** (2) **multi-type** (3) ~~scattered across sources~~ (4) **underspecified** (5) ~~inherently ambiguous to define~~



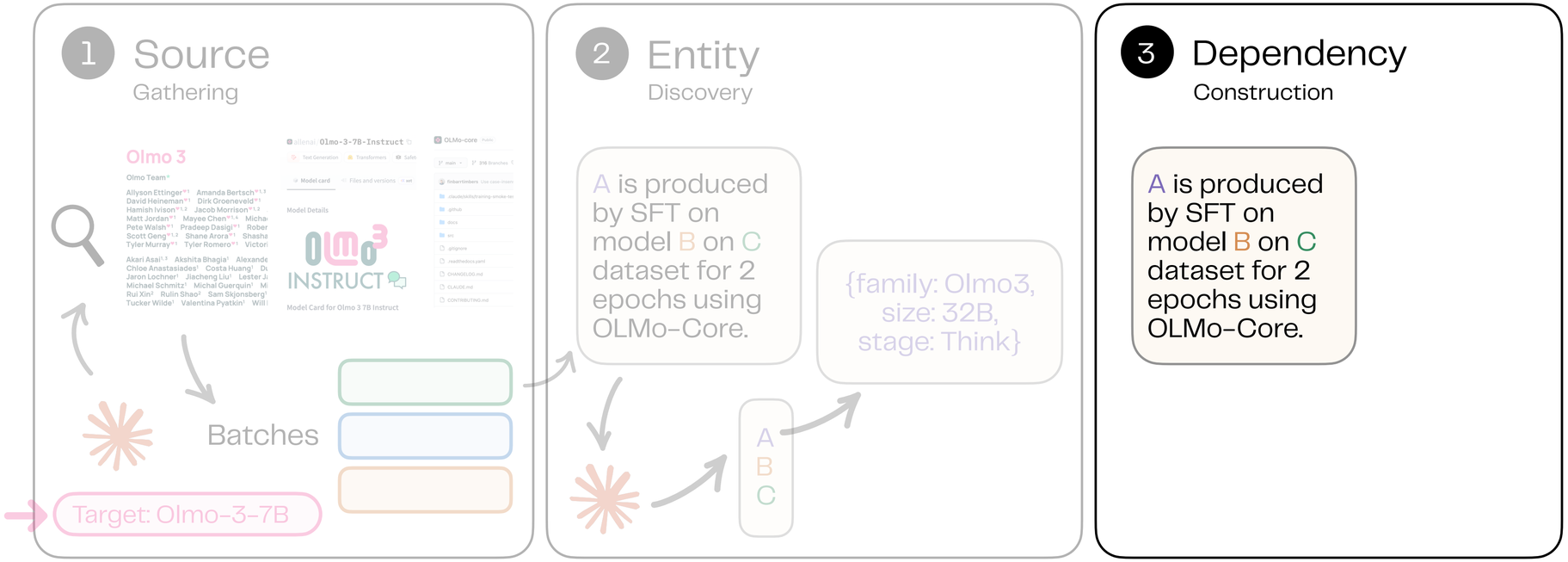
Dependencies are (1) **multi-hop** (2) **multi-type** (3) ~~scattered across sources~~ (4) **underspecified** (5) ~~inherently ambiguous to define~~



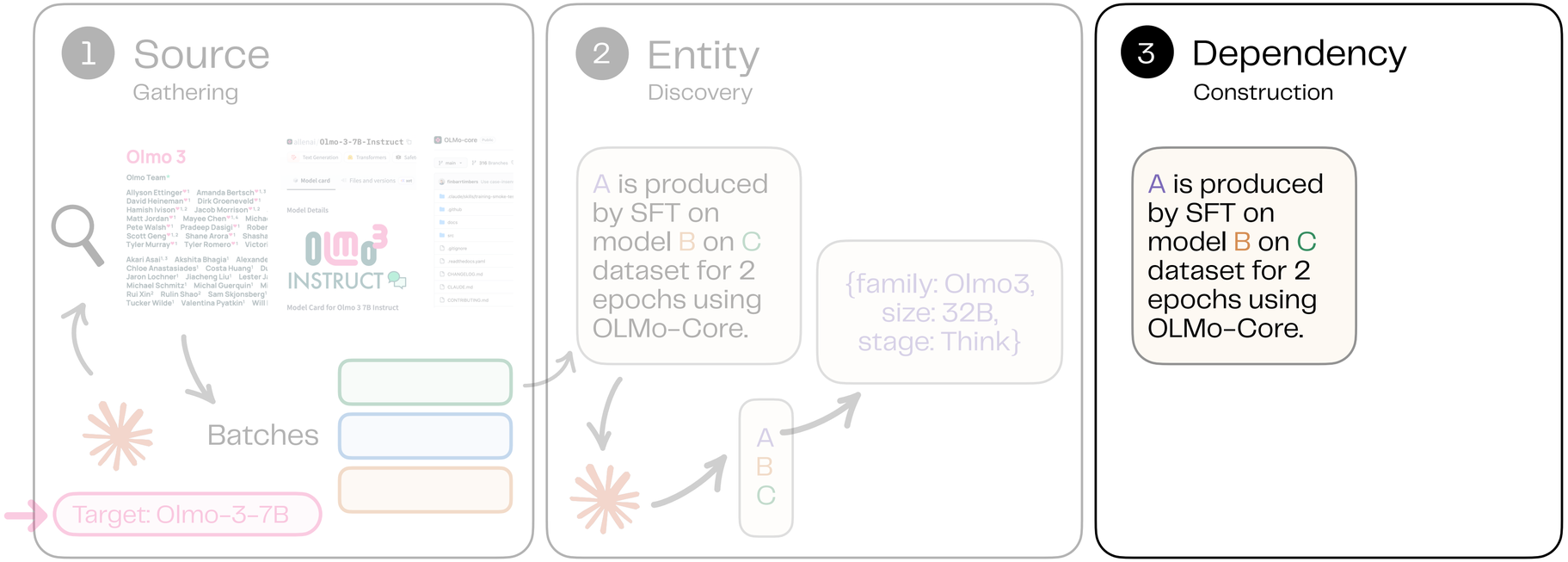
Dependencies are (1) **multi-hop** (2) **multi-type** (3) ~~scattered across sources~~ (4) ~~**underspecified**~~ (5) ~~inherently ambiguous to define~~



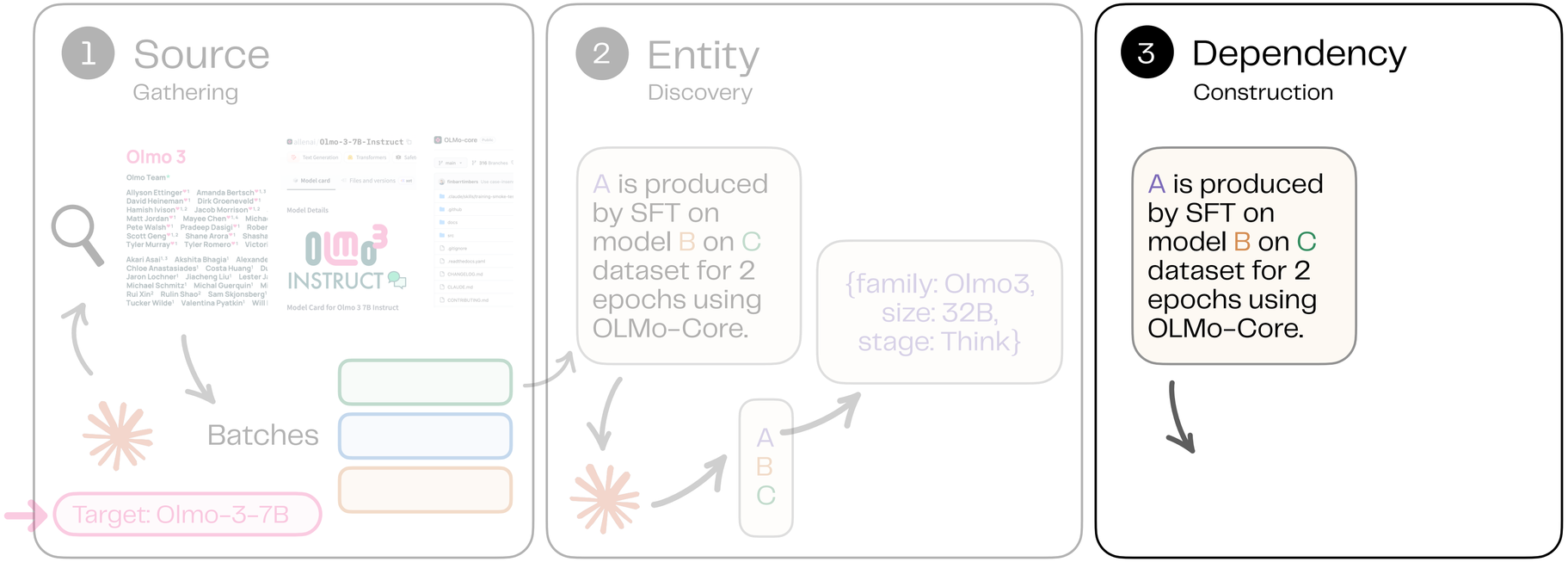
Dependencies are (1) **multi-hop** (2) **multi-type** (3) ~~scattered across sources~~ (4) ~~**underspecified**~~ (5) ~~inherently ambiguous to define~~



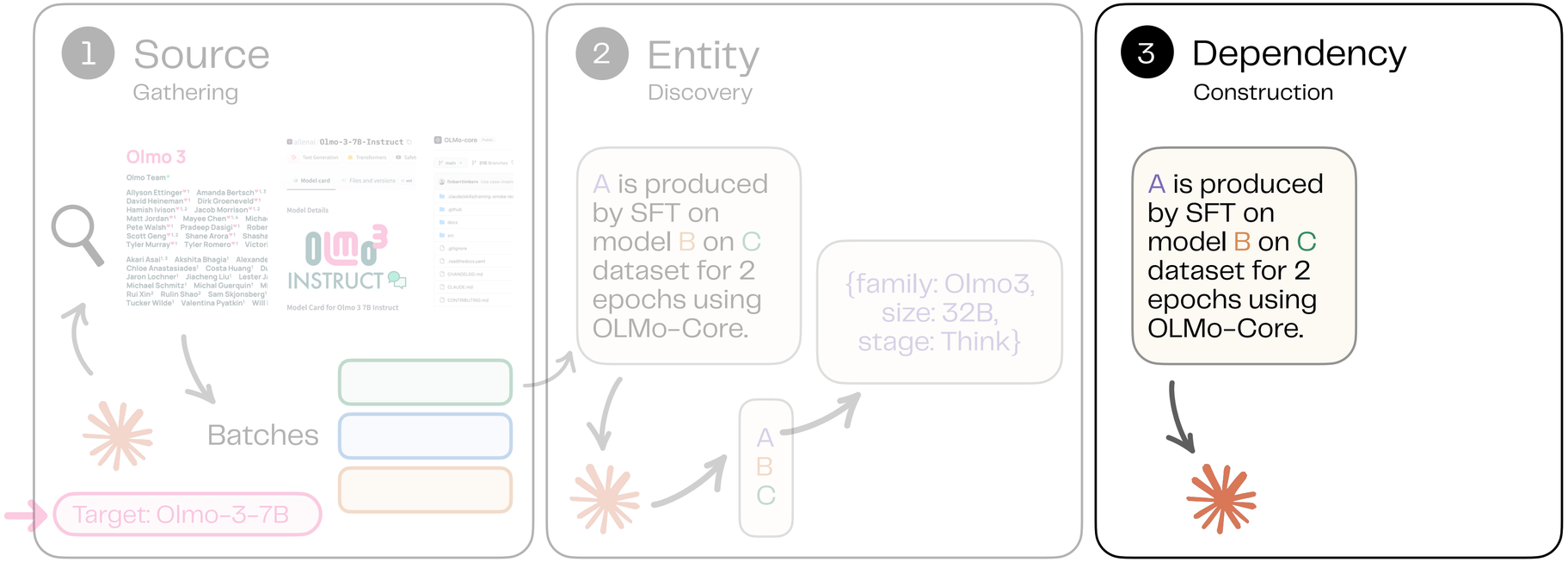
Dependencies are (1) **multi-hop** (2) **multi-type** (3) ~~scattered across sources~~ (4) ~~underspecified~~ (5) ~~inherently ambiguous to define~~



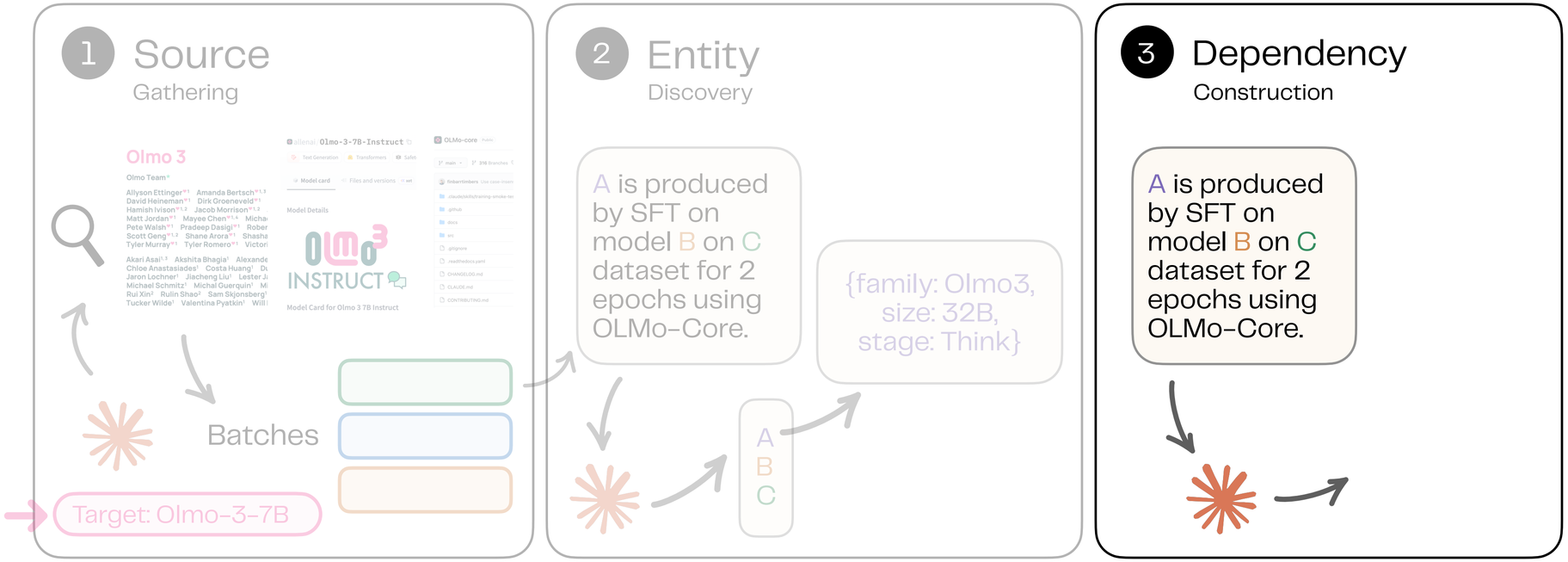
Dependencies are (1) **multi-hop** (2) **multi-type** (3) ~~scattered across sources~~ (4) ~~underspecified~~ (5) ~~inherently ambiguous to define~~



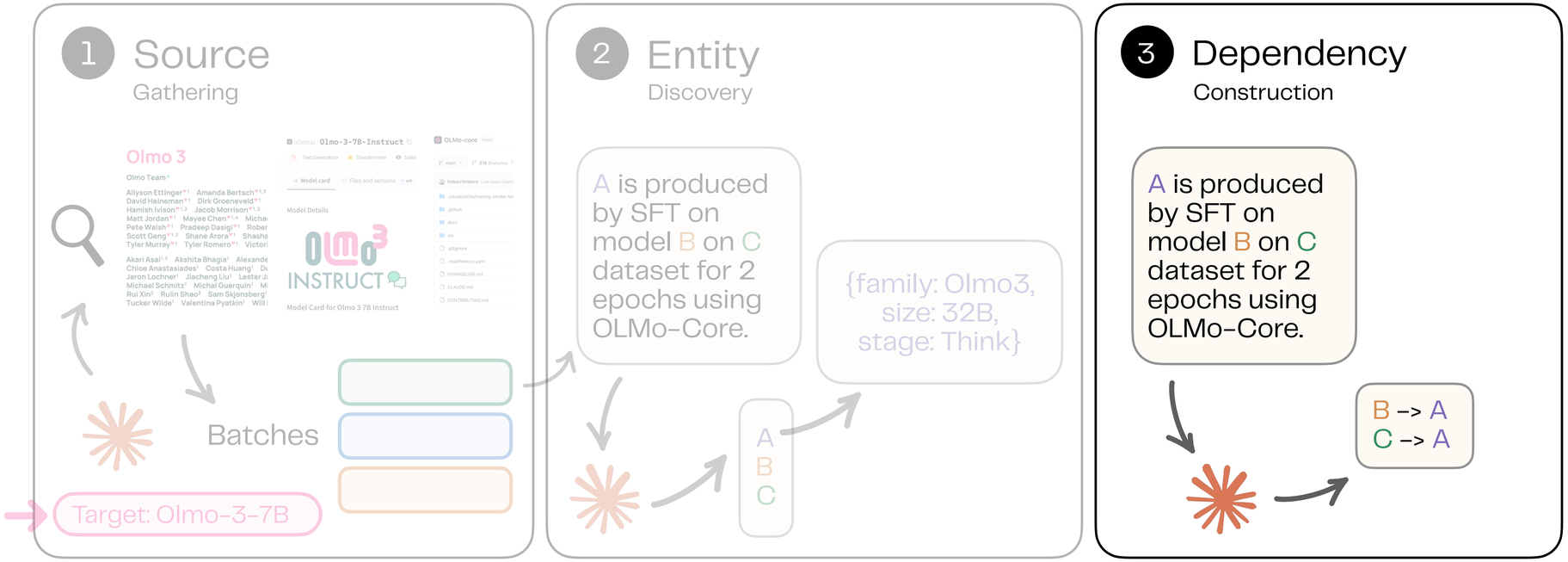
Dependencies are (1) **multi-hop** (2) **multi-type** (3) ~~scattered across sources~~ (4) ~~underspecified~~ (5) ~~inherently ambiguous to define~~



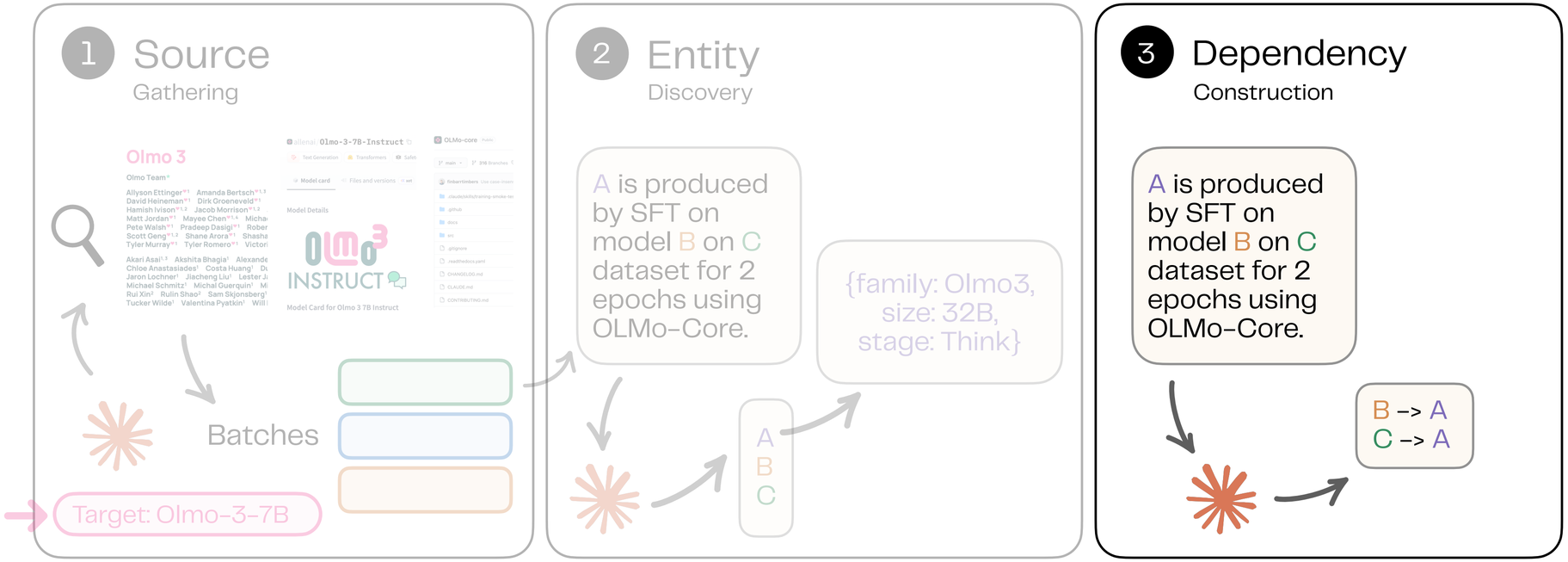
Dependencies are (1) **multi-hop** (2) **multi-type** (3) ~~scattered across sources~~ (4) ~~underspecified~~ (5) ~~inherently ambiguous to define~~



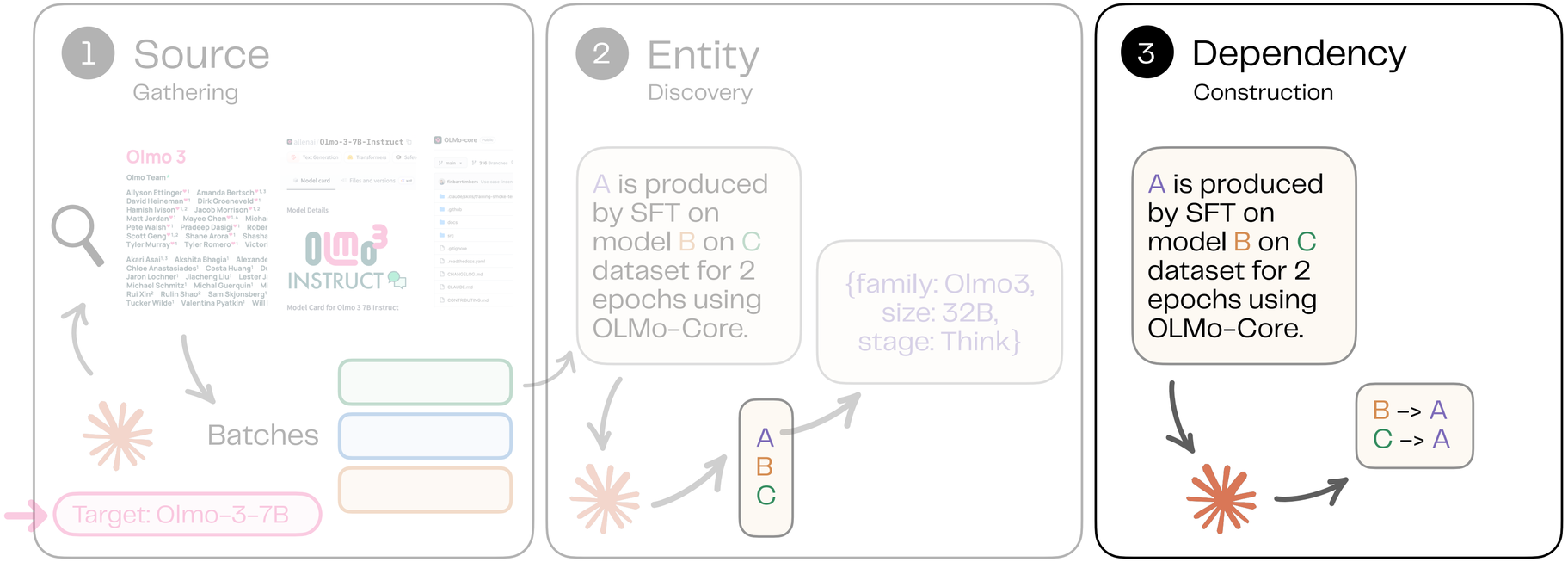
Dependencies are (1) **multi-hop** (2) **multi-type** (3) ~~scattered across sources~~ (4) ~~underspecified~~ (5) ~~inherently ambiguous to define~~



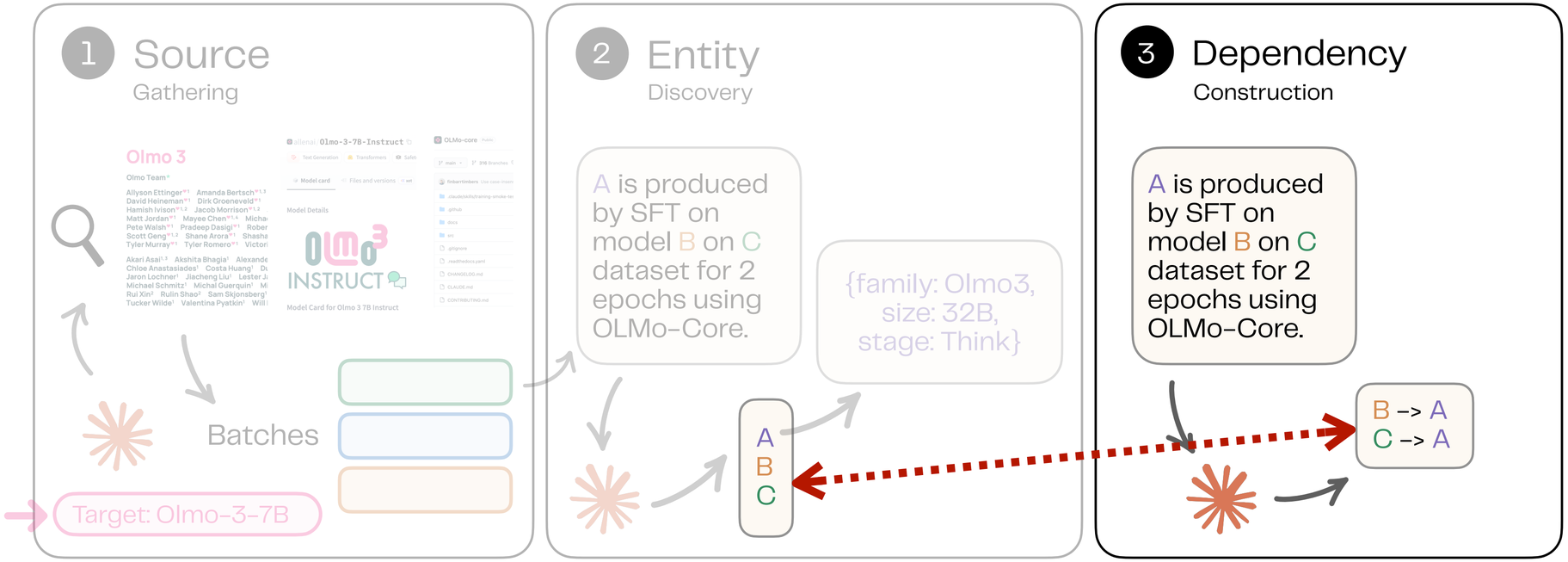
Dependencies are (1) **multi-hop** (2) ~~**multi-type**~~ (3) ~~scattered across sources~~ (4) ~~underspecified~~ (5) ~~inherently ambiguous to define~~



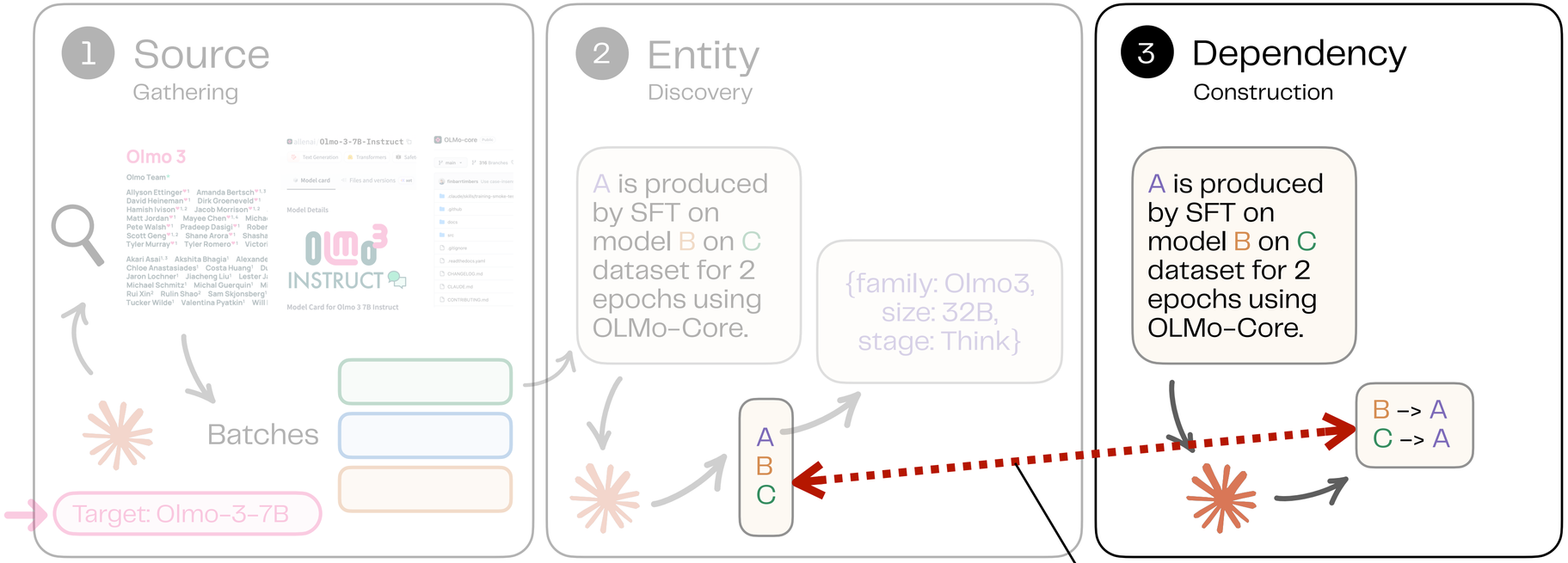
Dependencies are (1) **multi-hop** (2) ~~multi-type~~ (3) ~~scattered across sources~~ (4) ~~underspecified~~ (5) ~~inherently ambiguous to define~~



Dependencies are (1) **multi-hop** (2) ~~multi-type~~ (3) ~~scattered across sources~~ (4) ~~underspecified~~ (5) ~~inherently ambiguous to define~~

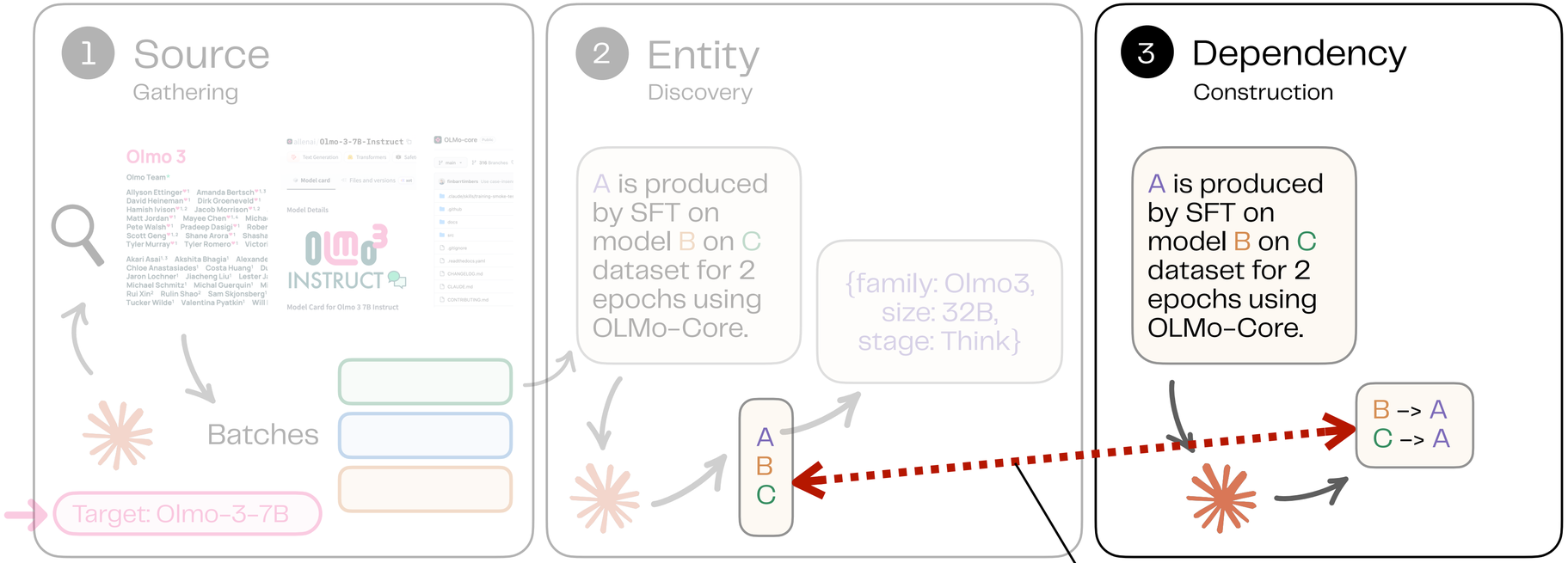


Dependencies are (1) **multi-hop** (2) ~~multi-type~~ (3) ~~scattered across sources~~ (4) ~~underspecified~~ (5) ~~inherently ambiguous to define~~



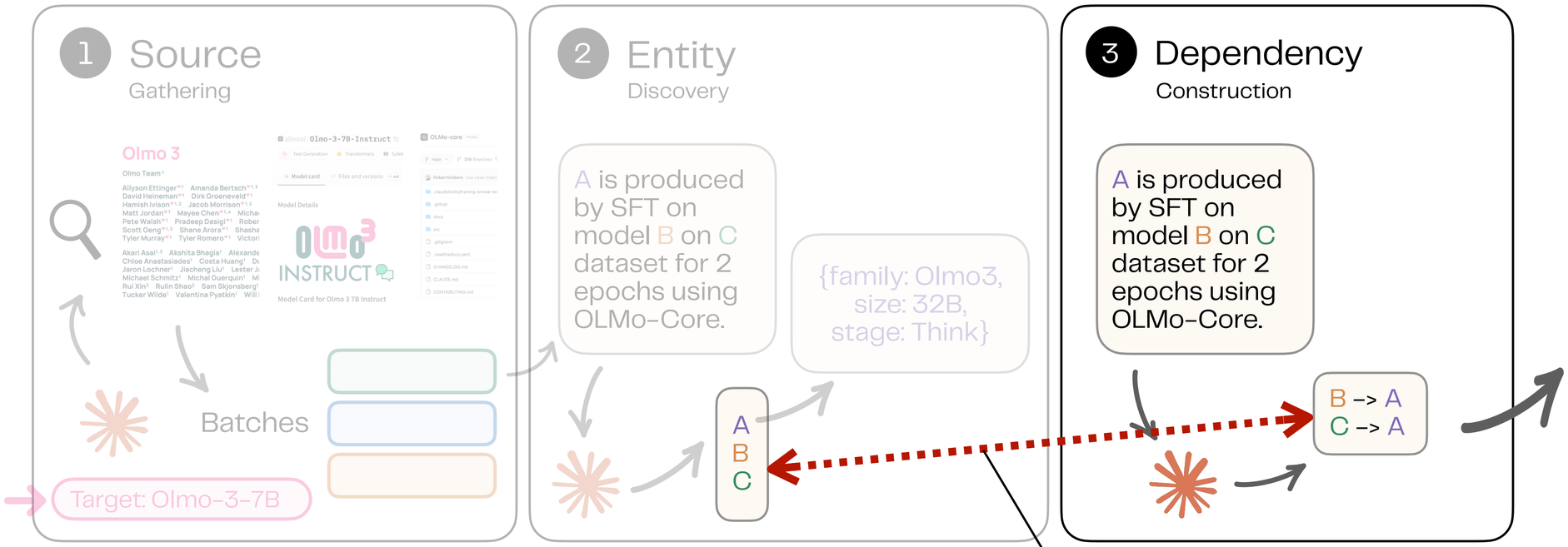
Recall that we have already identified the nodes in the graph; all that remains is to connect them

Dependencies are (1) **multi-hop** (2) ~~multi-type~~ (3) ~~scattered across sources~~ (4) ~~underspecified~~ (5) ~~inherently ambiguous to define~~



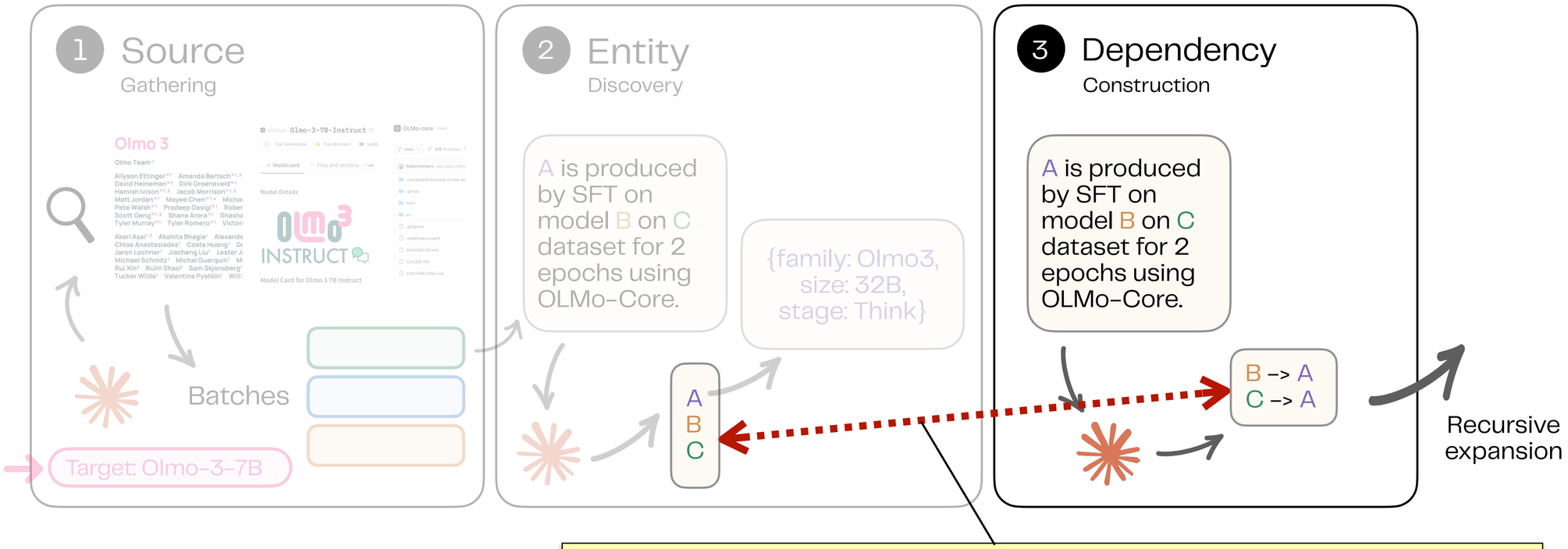
Recall that we have already identified the nodes in the graph; all that remains is to connect them

Dependencies are (1) **multi-hop** (2) ~~multi-type~~ (3) ~~scattered across sources~~ (4) ~~underspecified~~ (5) ~~inherently ambiguous to define~~



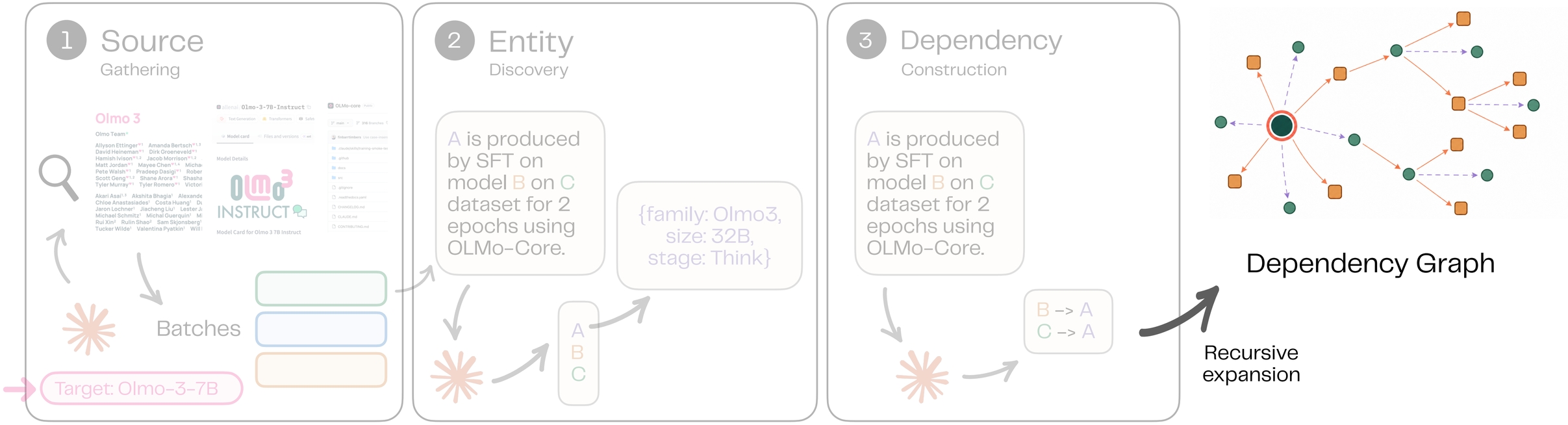
Recall that we have already identified the nodes in the graph; all that remains is to connect them

Dependencies are (1) **multi-hop** (2) ~~multi-type~~ (3) ~~scattered across sources~~ (4) ~~underspecified~~ (5) ~~inherently ambiguous to define~~

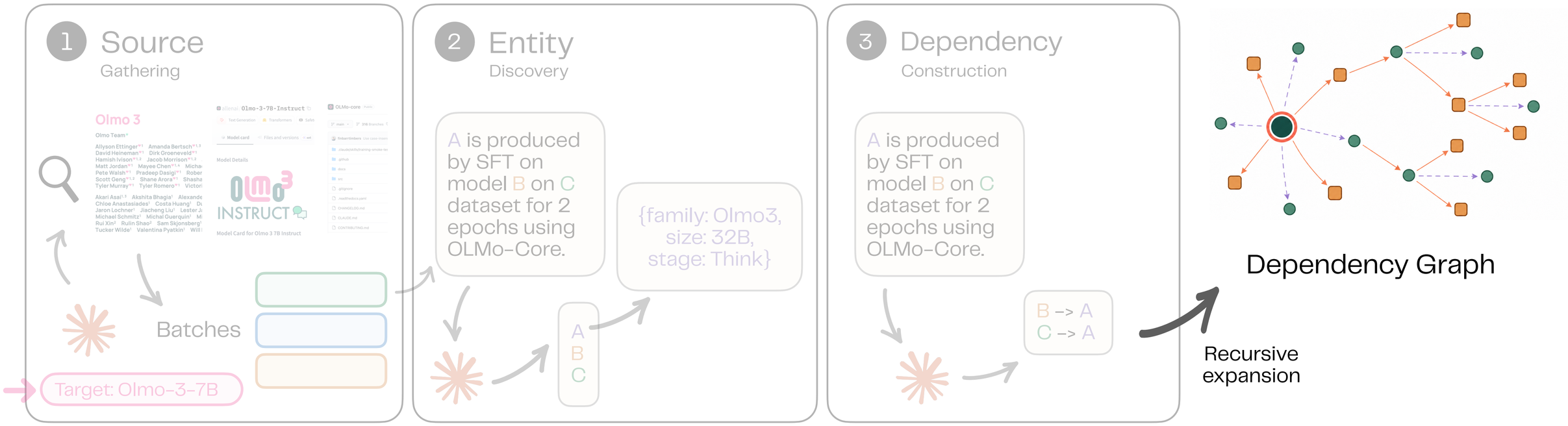


Recall that we have already identified the nodes in the graph; all that remains is to connect them

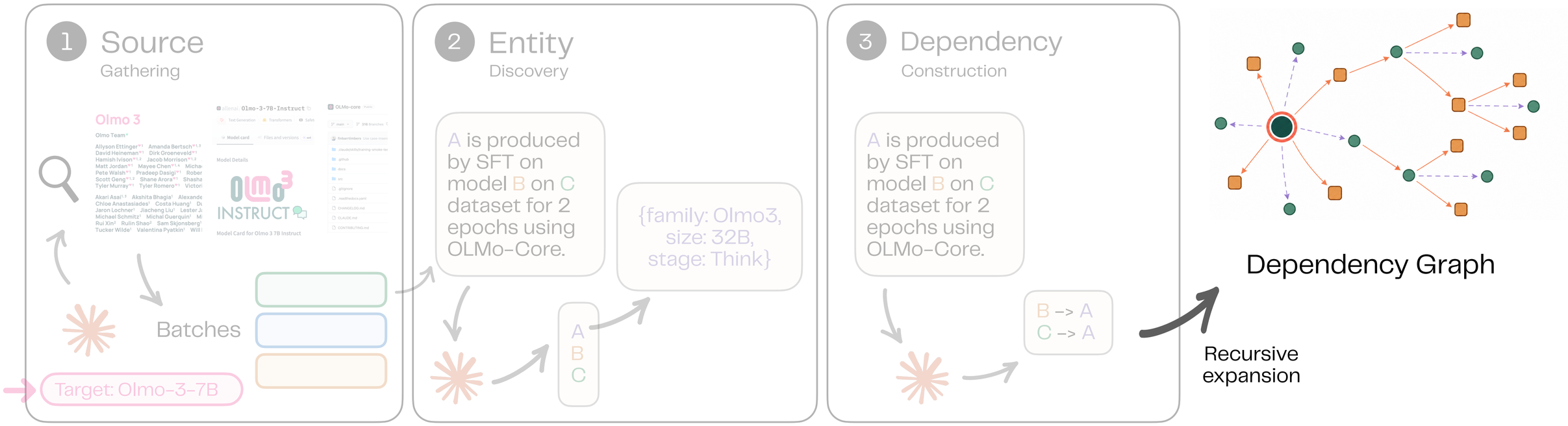
Dependencies are (1) **multi-hop** (2) ~~multi-type~~ (3) ~~scattered across sources~~ (4) ~~underspecified~~ (5) ~~inherently ambiguous to define~~



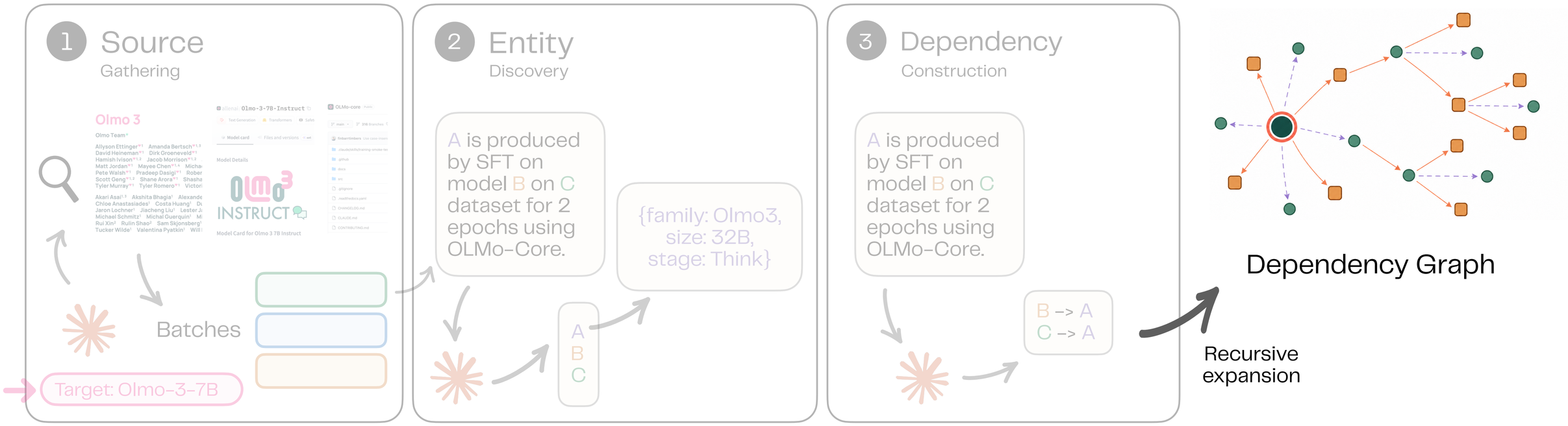
Dependencies are (1) ~~multi-hop~~ (2) ~~multi-type~~ (3) ~~scattered across sources~~ (4) ~~underspecified~~ (5) ~~inherently ambiguous to define~~



Design of ModSleuth

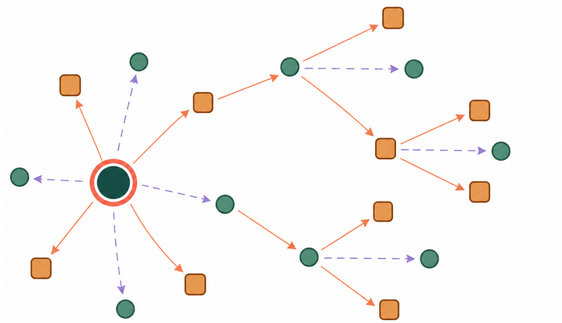


Design of ModSleuth

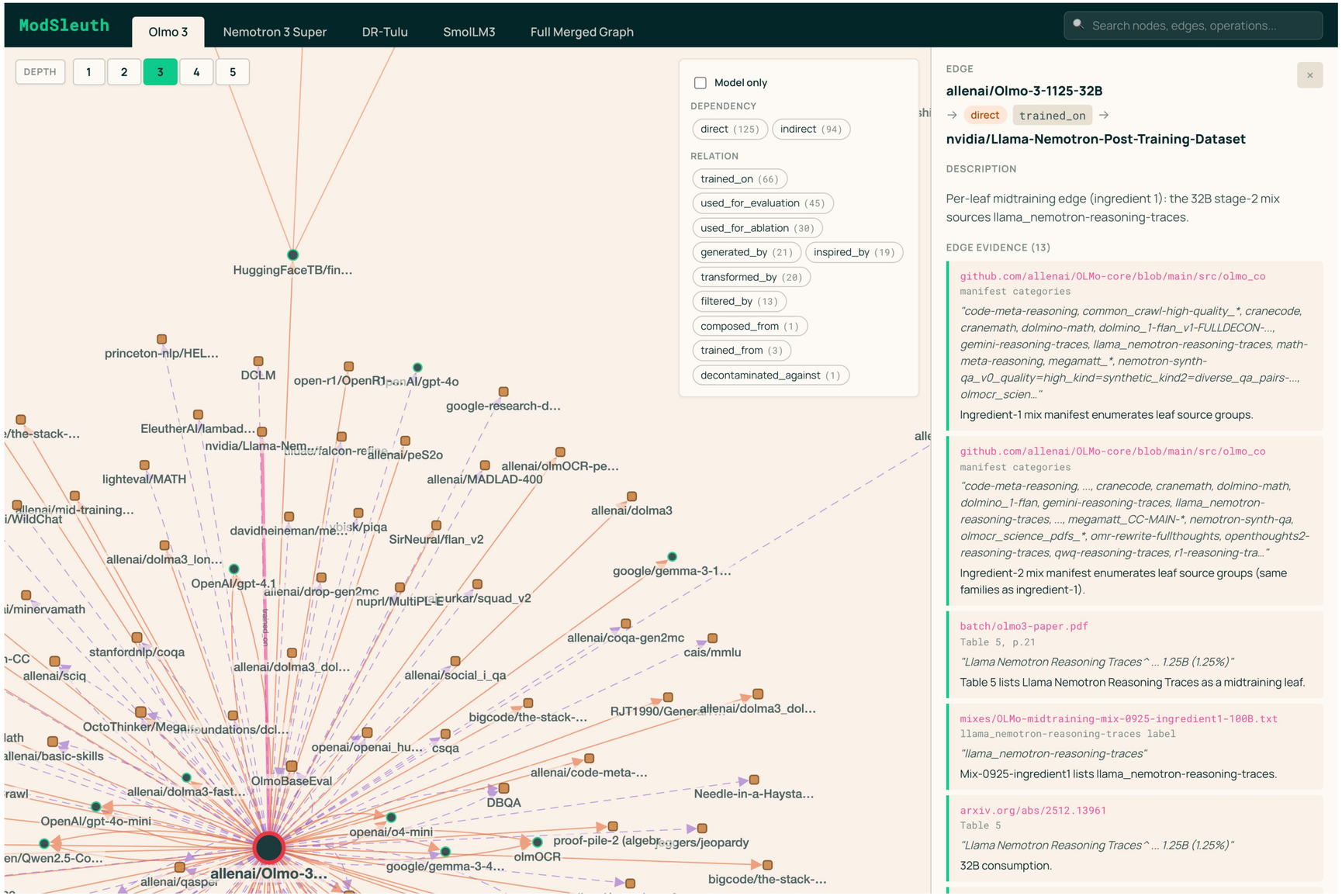


After having the **original graph** from our system, we use Claude Sonnet to audit every edge and **only keep and release** the *verified* edges. (We also experimented the agreement w/ other verifiers.)

Output of ModSleuth

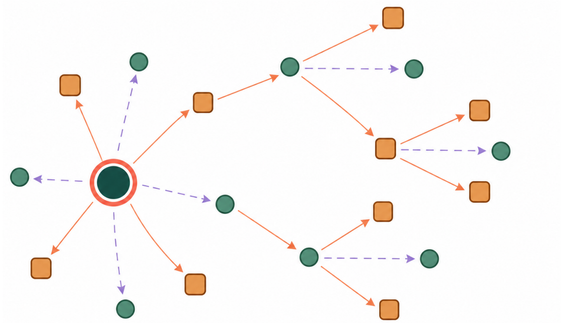


Dependency Graph
(All verified)

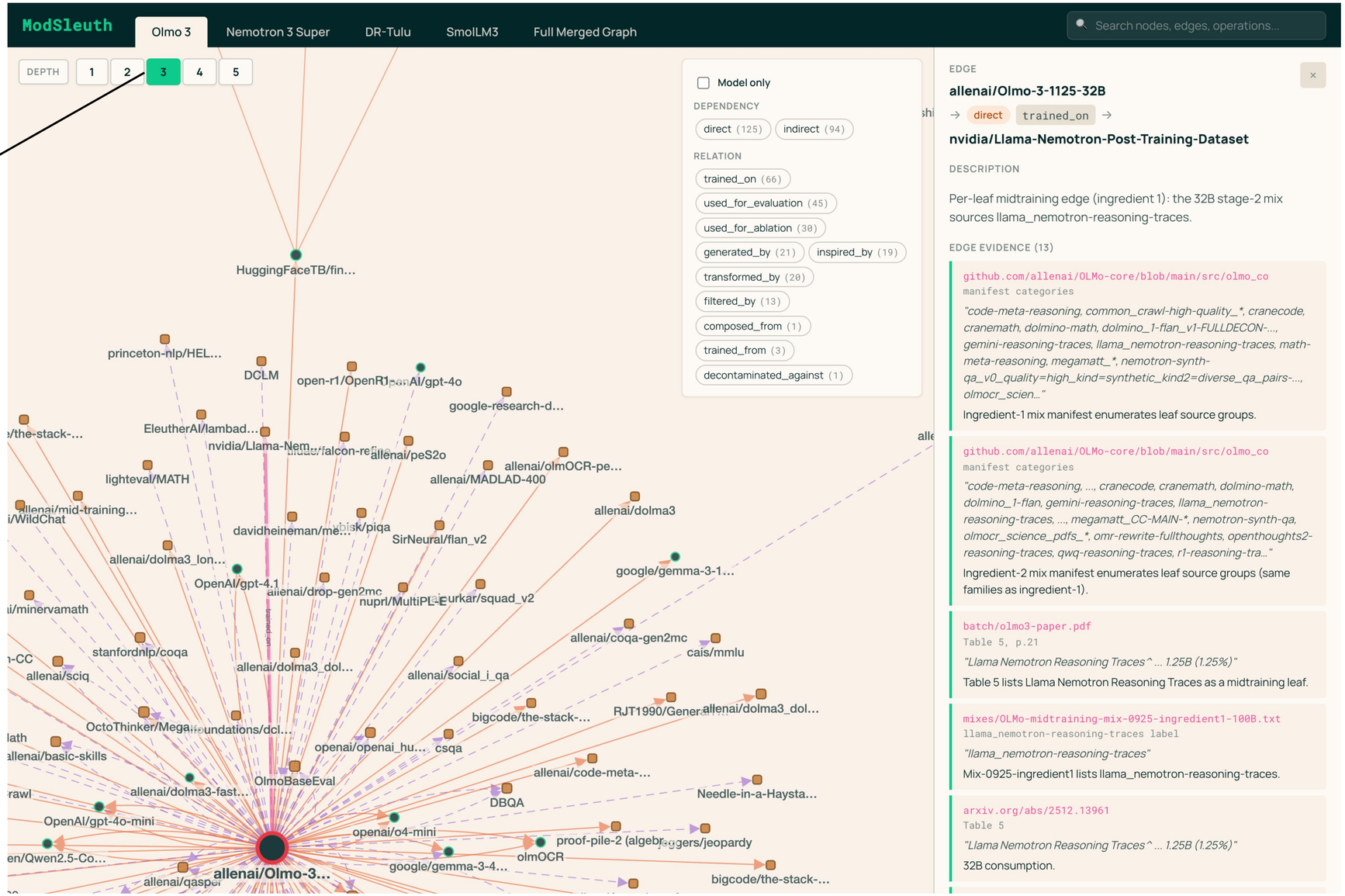


Output of ModSleuth

Recursive Exploration



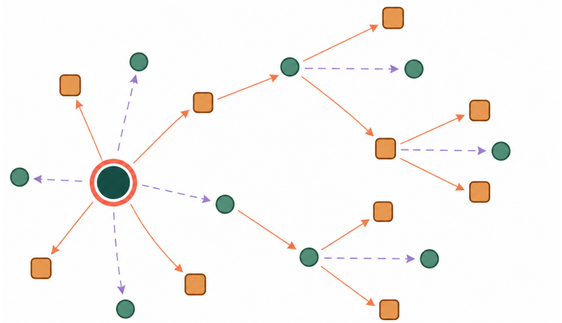
Dependency Graph
(All verified)



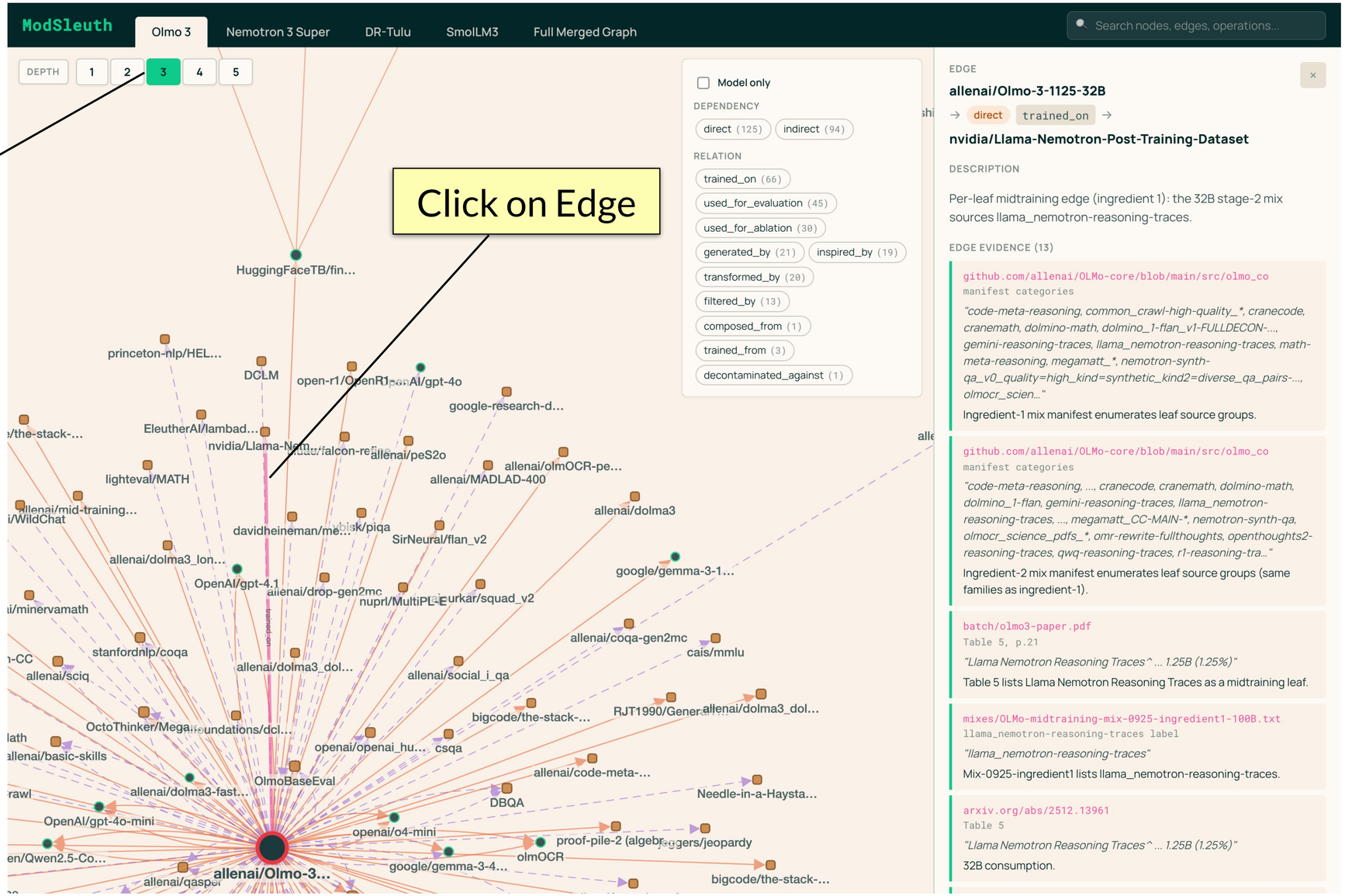
Output of ModSleuth

Recursive Exploration

Click on Edge



Dependency Graph
(All verified)

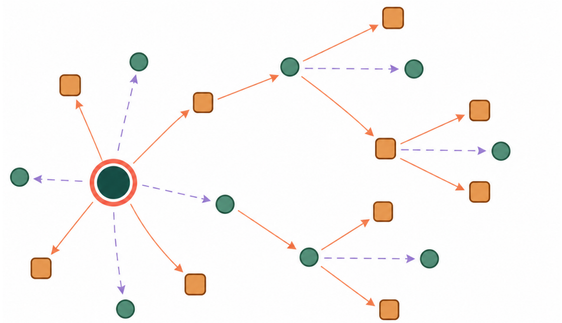


Output of ModSleuth

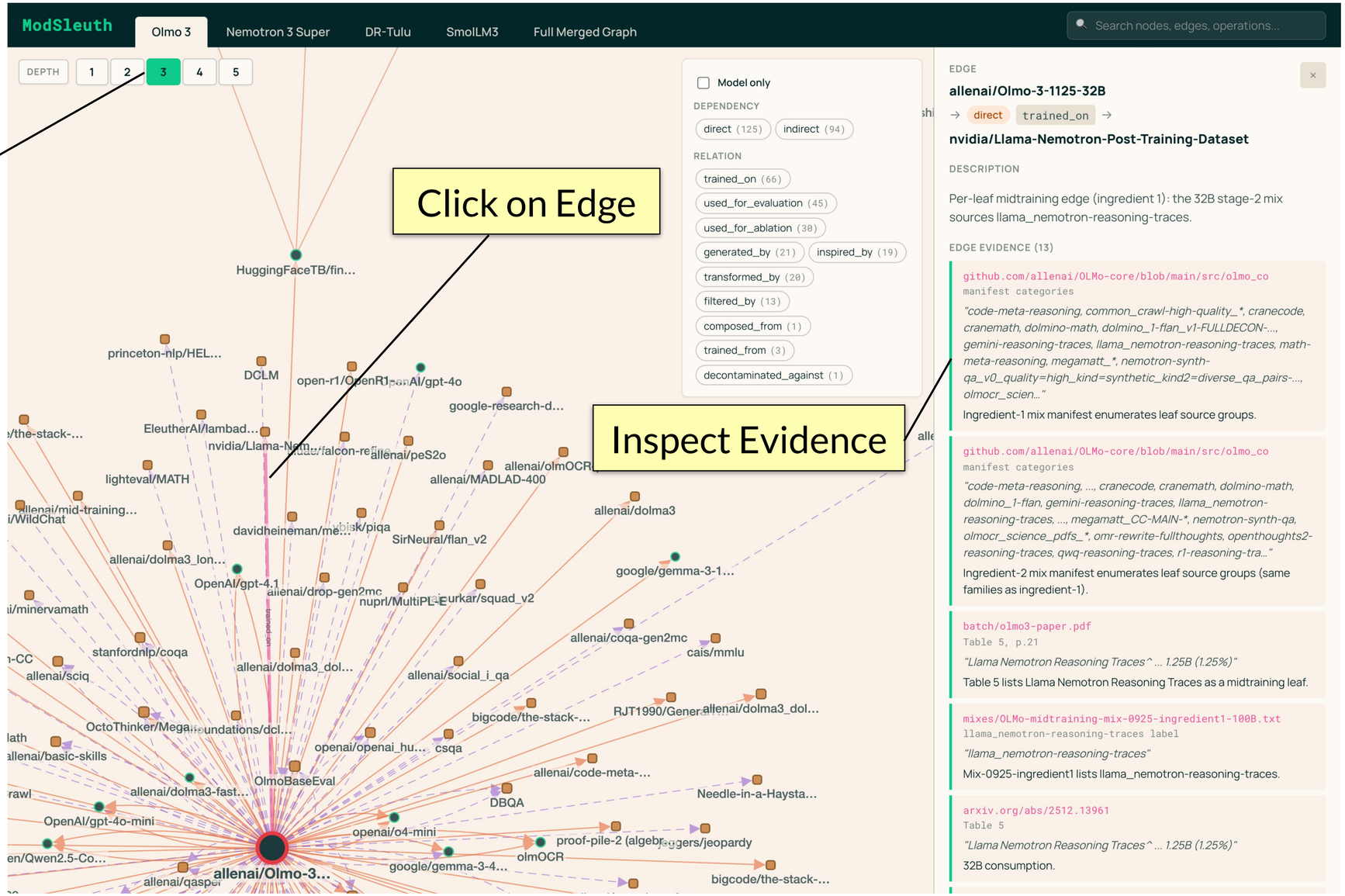
Recursive Exploration

Click on Edge

Inspect Evidence



Dependency Graph
(All verified)



Evaluating ModSleuth

Problem: No complete ground truth

Experts are accurate, but often incomplete

Evaluating ModSleuth

Problem: No complete ground truth

Experts are accurate, but often incomplete

Target Model — Dataset — Generator

Often stops after 1-2 hops

Less-obvious roles are easy to omit

- Judges
- Filters
- Rewriters
- Embedding models

Evaluating ModSleuth

Problem: No complete ground truth

Experts are accurate, but often incomplete

Target Model — Dataset — Generator

Often stops after 1-2 hops

Less-obvious roles are easy to omit

- Judges
- Filters
- Rewriters
- Embedding models

ModSleuth traces up to 8 hops

Evaluating ModSleuth

Problem: No complete ground truth

Experts are accurate, but often incomplete

Target Model — Dataset — Generator

Often stops after 1-2 hops

Less-obvious roles are easy to omit

- Judges
- Filters
- Rewriters
- Embedding models

ModSleuth traces up to 8 hops

Solution: Evaluate verified recovery

How many correct dependencies can each system recover?

Evaluating ModSleuth

Problem: No complete ground truth

Experts are accurate, but often incomplete

Target Model — Dataset — Generator

Often stops after 1-2 hops

Less-obvious roles are easy to omit

- Judges
- Filters
- Rewriters
- Embedding models

ModSleuth traces up to 8 hops

Solution: Evaluate verified recovery

How many correct dependencies can each system recover?

Recovered Dependency

with cited evidence

↓

Claude Sonnet 4.6

with web search

↓

Verified | Unclear | Refuted

Evaluating ModSleuth

Problem: No complete ground truth

Experts are accurate, but often incomplete

Target Model — Dataset — Generator

Often stops after 1-2 hops

Less-obvious roles are easy to omit

- Judges
- Filters
- Rewriters
- Embedding models

ModSleuth traces up to 8 hops

Solution: Evaluate verified recovery

How many correct dependencies can each system recover?

Recovered Dependency

with cited evidence

↓

Claude Sonnet 4.6

with web search

↓

Verified | Unclear | Refuted

Key Metric:

 # verified dependencies recovered

Evaluation Results

	Olmo 3	Nemotron 3	DR Tulu	SmolLM3	Total
Baselines					
GPT-5.5 Pro	59	156	46	53	314
GPT-5.4 Pro	87	75	44	77	283
CC-single	91	78	37	69	275
ChatGPT Deep Research	37	70	28	36	171
Ours					
ModSleuth (depth-1)	182	237	42	23	484
ModSleuth (unbounded)	305	613	44	98	1,060
ModSleuth (BFS reach.)†	**481**	**1,023**	**62**	**127**	**1,654**

Strongest Baseline

Evaluation Results

	Olmo 3	Nemotron 3	DR Tulu	SmolLM3	Total
Baselines					
GPT-5.5 Pro	59	156	46	53	314
GPT-5.4 Pro	87	75	44	77	283
CC-single	91	78	37	69	275
ChatGPT Deep Research	37	70	28	36	171
Ours					
ModSleuth (depth-1)	182	237	42	23	484
ModSleuth (unbounded)	305	613	44	98	1,060
ModSleuth (BFS reach.)†	**481**	**1,023**	**62**	**127**	**1,654**

3x Strongest Baseline

Depth-1 ⊂ Unbounded ⊂ BFS-Reachability

Depth-1

Target's own dependencies

Unbounded*

All dependencies recovered during target's investigation.

Primary baseline comparison

BFS-Reachability

Full reachable graph

Includes upstream findings

Talk Overview

1. What counts as a dependency, and why is tracing them hard?

   Much more than a list of names! Every dependency enters through some operations (generating, filtering), many hops upstream, documented in scattered and ambiguous sources.

2. Can we *automatically* and *reliably* recover the dependency graph from public artifacts?

   Yes! ModSleuth, an agentic system we developed, recursively expands the graph and grounds every edge in source evidence—recovering 3x more verified dependencies than baselines.

3. What do the recovered graphs reveal?

   Plenty that no single page shows: hidden upstream models, license terms riding along multiple hops, benchmarks on both sides of training and eval, etc.

Quantitative Findings

Audit role	Olmo 3		Nemotron 3		DR Tulu		SmolLM3		Total
	Model	Data	Model	Data	Model	Data	Model	Data	
Direct dependencies									
Training inputs	0	172	0	547	0	11	0	87	813
Upstream operations on training data	85	11	253	13	7	1	7	6	350
Weight-level model lineage	4	0	20	0	3	0	3	0	28
Direct total	**89**	**183**	**273**	**560**	**10**	**12**	**10**	**93**	**1,191**
Indirect dependencies									
Evaluation / ablation	6	163	20	122	8	28	3	10	360
Methodology / audit influence	8	32	24	24	1	3	7	4	103
Indirect total	**14**	**195**	**44**	**146**	**9**	**31**	**10**	**14**	**463**
Grand total	**103**	**378**	**317**	**706**	**19**	**43**	**20**	**107**	**1,654**

Quantitative Findings

Audit role	Olmo 3		Nemotron 3		DR Tulu		SmolLM3		Total
	Model	Data	Model	Data	Model	Data	Model	Data	
Direct dependencies									
Training inputs	0	172	0	547	0	11	0	87	813
Upstream operations on training data	85	11	253	13	7	1	7	6	350
Weight-level model lineage	4	0	20	0	3	0	3	0	28
Direct total	**89**	**183**	**273**	**560**	**10**	**12**	**10**	**93**	**1,191**
Indirect dependencies									
Evaluation / ablation	6	163	20	122	8	28	3	10	360
Methodology / audit influence	8	32	24	24	1	3	7	4	103
Indirect total	**14**	**195**	**44**	**146**	**9**	**31**	**10**	**14**	**463**
Grand total	**103**	**378**	**317**	**706**	**19**	**43**	**20**	**107**	**1,654**

Quantitative Findings

	Internal		External				
Target	Model	Data	Model	Data	Int. total	Ext. total	Ext. %
Olmo 3	13	106	90	272	119	362	75.3%
Nemotron 3	40	203	277	503	243	780	76.2%
DR Tulu	2	9	17	34	11	51	82.3%
SmolLM3	3	28	17	79	31	96	75.6%

An edge is *internal* if its upstream artifact shares the target's organization (e.g. Ai2 artifacts for Olmo 3 and DR Tulu), and *external* otherwise.

75-82% of dependencies come outside the target organization

Key Audit Patterns

Hidden Upstream Models

License-relevant Paths

Code-level Provenance

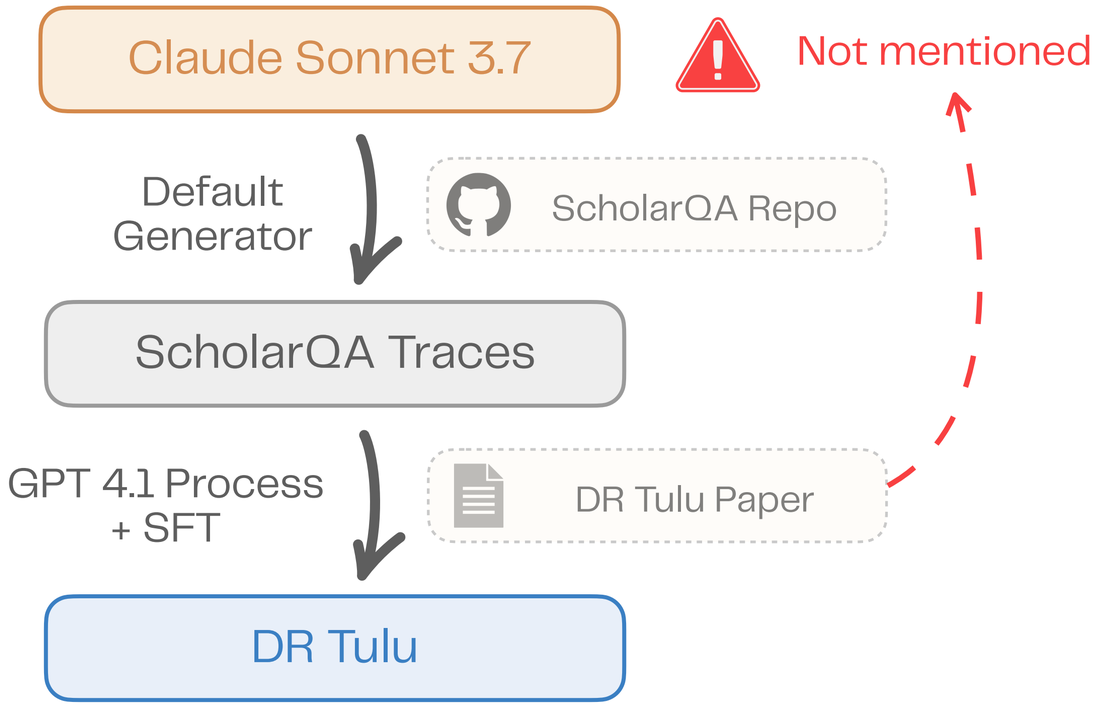
Train-Eval Coupling

Model-mediated Selection

Good Practices

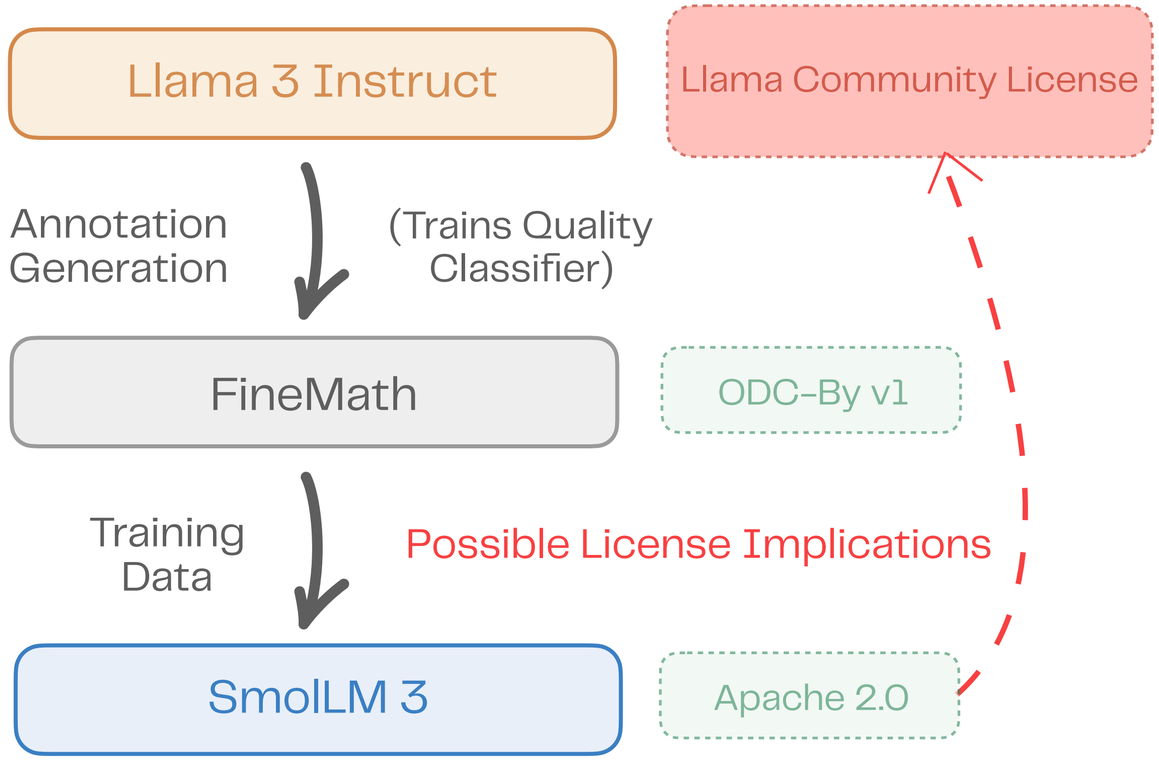
Hidden Upstream Models

Dependencies can be hidden behind intermediate datasets, filters, classifiers, teachers, and tools.



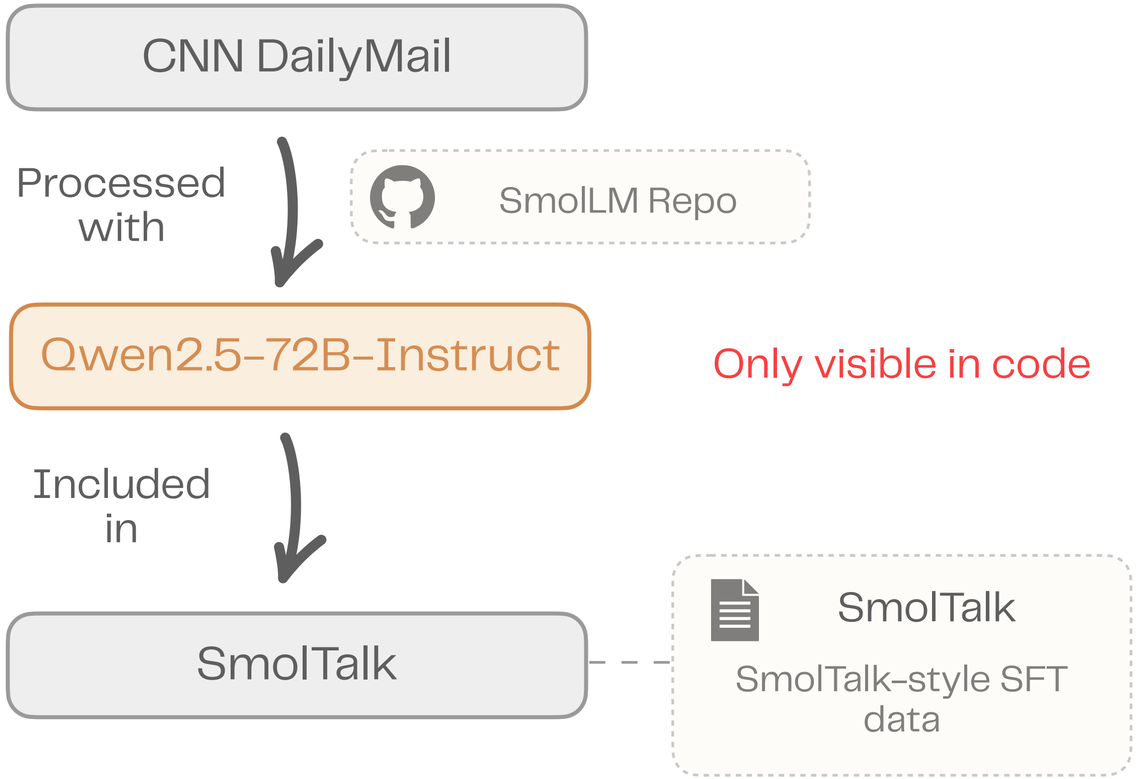
License-Relevant Paths

Synthetic data and annotation pipelines can carry license or terms-of-use implications from upstream models.



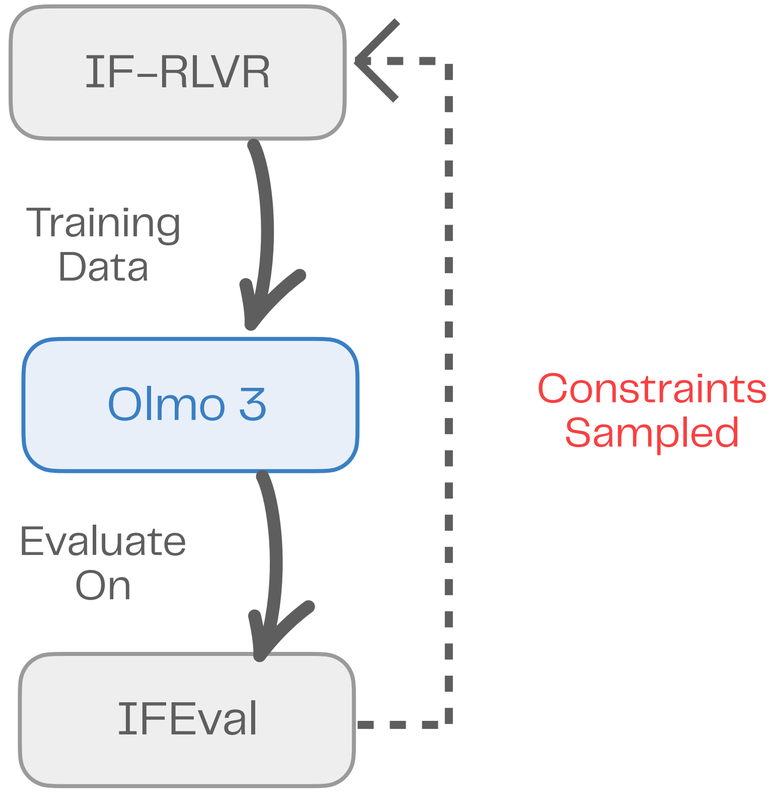
Code-level Provenance

Training scripts, YAMLs, and data-prep code show dependencies hidden by paper/ dataset card summaries.



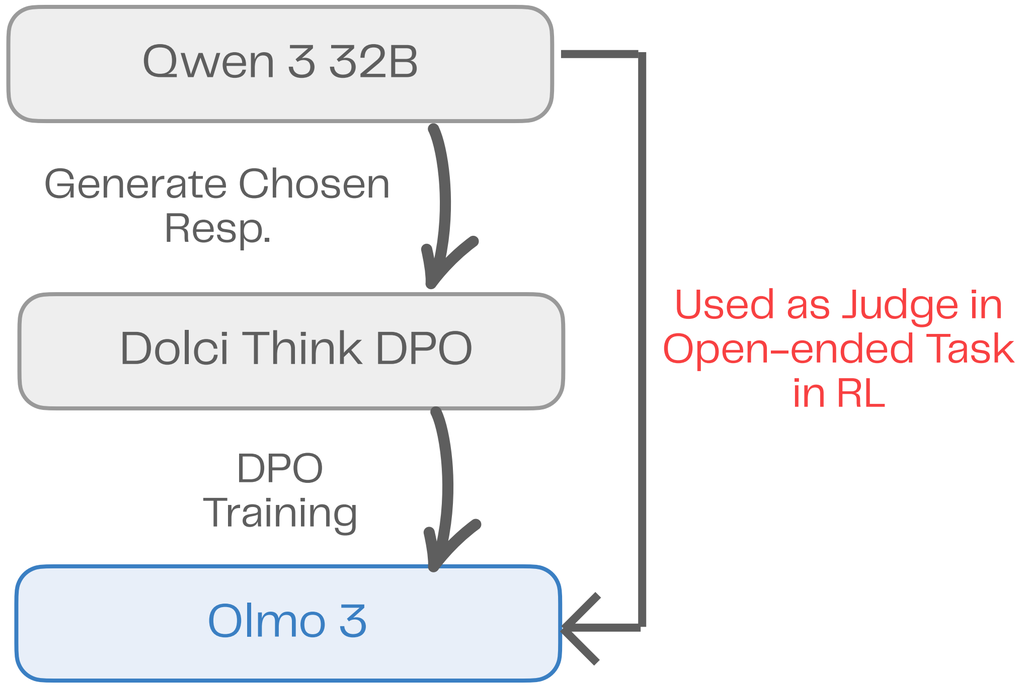
Train-Eval Coupling

Benchmarks can enter the training pipeline as prompts, data, validation environments, or synthetic expansion, then reappear at evaluation time



Model-mediated Selection

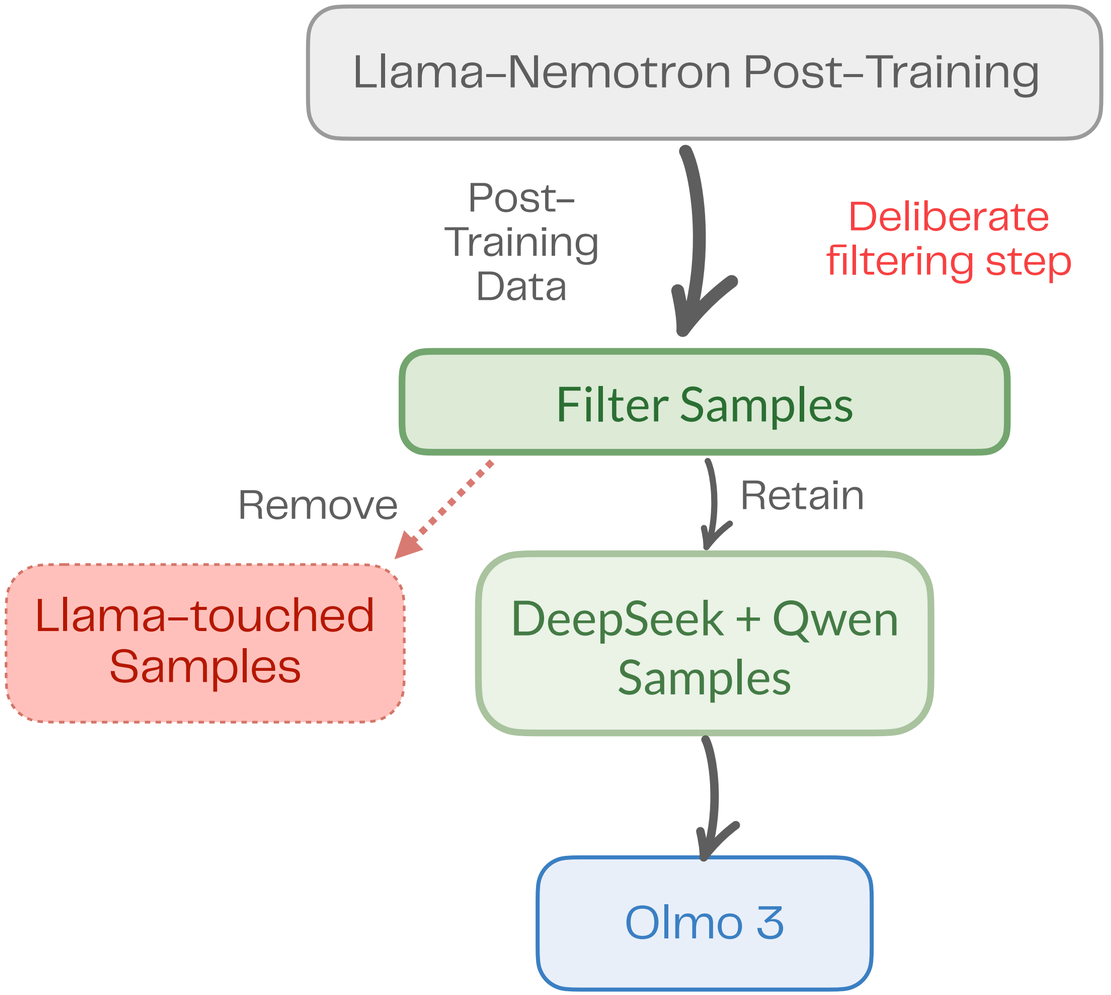
Generators, judges, filters, and reward models decide which examples are created, kept, or rewarded.



Good Practices

The graph surfaces good practices, not just risks

- Permissive teacher swaps
- Filtering
- Decontamination



Talk Overview

1. What counts as a dependency, and why is tracing them hard?

   Much more than a list of names! Every dependency enters through some operations (generating, filtering), many hops upstream, documented in scattered and ambiguous sources.

2. Can we *automatically* and *reliably* recover the dependency graph from public artifacts?

   Yes! ModSleuth, an agentic system we developed, recursively expands the graph and grounds every edge in source evidence—recovering 3x more verified dependencies than baselines.

3. What do the recovered graphs reveal?

   Plenty that no single page shows: hidden upstream models, license terms riding along multiple hops, benchmarks on both sides of training and eval, etc.

Talk Overview

1. What counts as a dependency, and why is tracing them hard?

   Much more than a list of names! Every dependency enters through some operations (generating, filtering), many hops upstream, documented in scattered and ambiguous sources.

2. Can we *automatically* and *reliably* recover the dependency graph from public artifacts?

   Yes! ModSleuth, an agentic system we developed, recursively expands the graph and grounds every edge in source evidence—recovering 3x more verified dependencies than baselines.

3. What do the recovered graphs reveal?

   Plenty that no single page shows: hidden upstream models, license terms riding along multiple hops, benchmarks on both sides of training and eval, etc.

Thanks for listening!

Paper
https://arxiv.org/abs/2606.12385

GitHub
https://github.com/cal-data-audit/modsleuth

Demo
https://modsleuth.cal-data-audit.org

Please leave your kind feedback
at https://tinyurl.com/modsleuth-fdbk