



ACQUISITION INNOVATION
RESEARCH CENTER



CLEARED
For Open Publication

Sep 09, 2026

Department of War
OFFICE OF PREPUBLICATION AND SECURITY REVIEW

SLIDES ONLY

NO SCRIPT PROVIDED

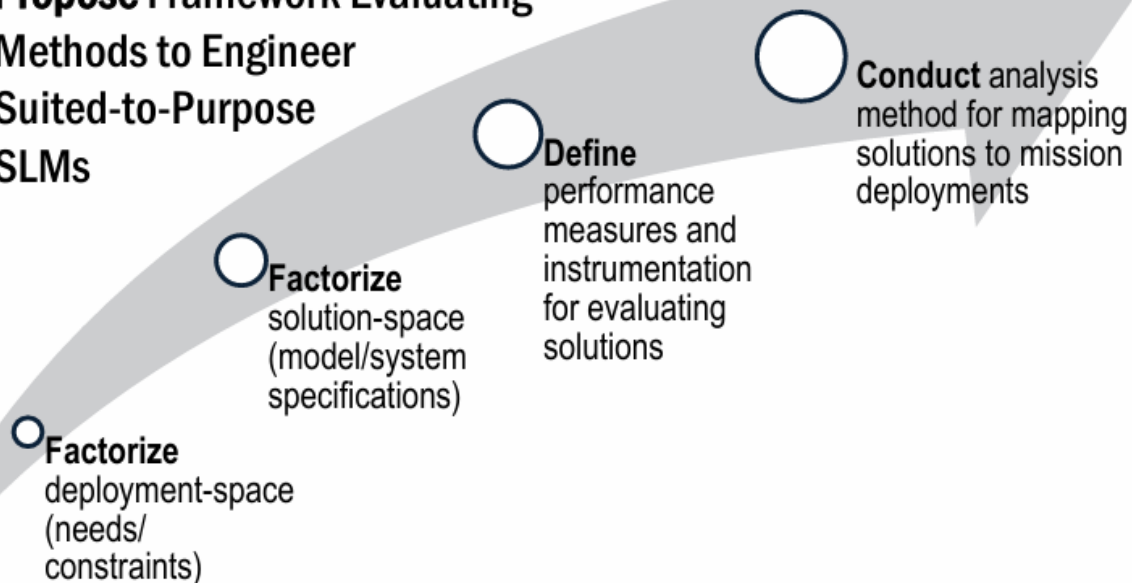
EVALUATING METHODS TO GENERATE MISSION-RELEVANT SMALL LANGUAGE MODELS

AI4SE & SE4AI WORKSHOP | T&E OF LLMs AND AGENTS
SEPTEMBER 23, 2026

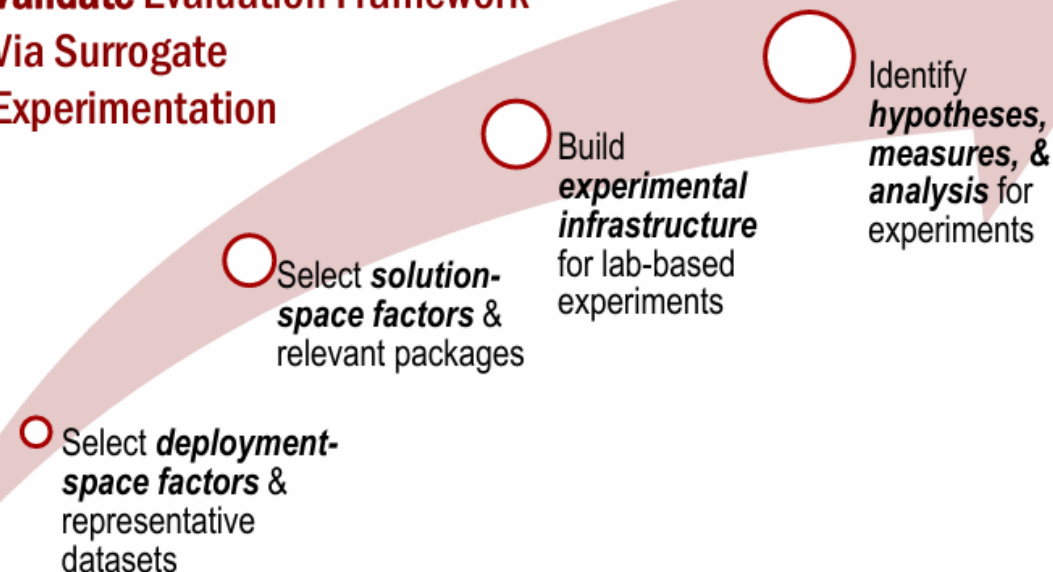
PRESENTER: EMMA MENO
CO-AUTHORS: DANIEL SOBIEN,
M. ALEXANDER KYER, JUSTIN KROMETIS,
ERIN LANUS, LAURA FREEMAN

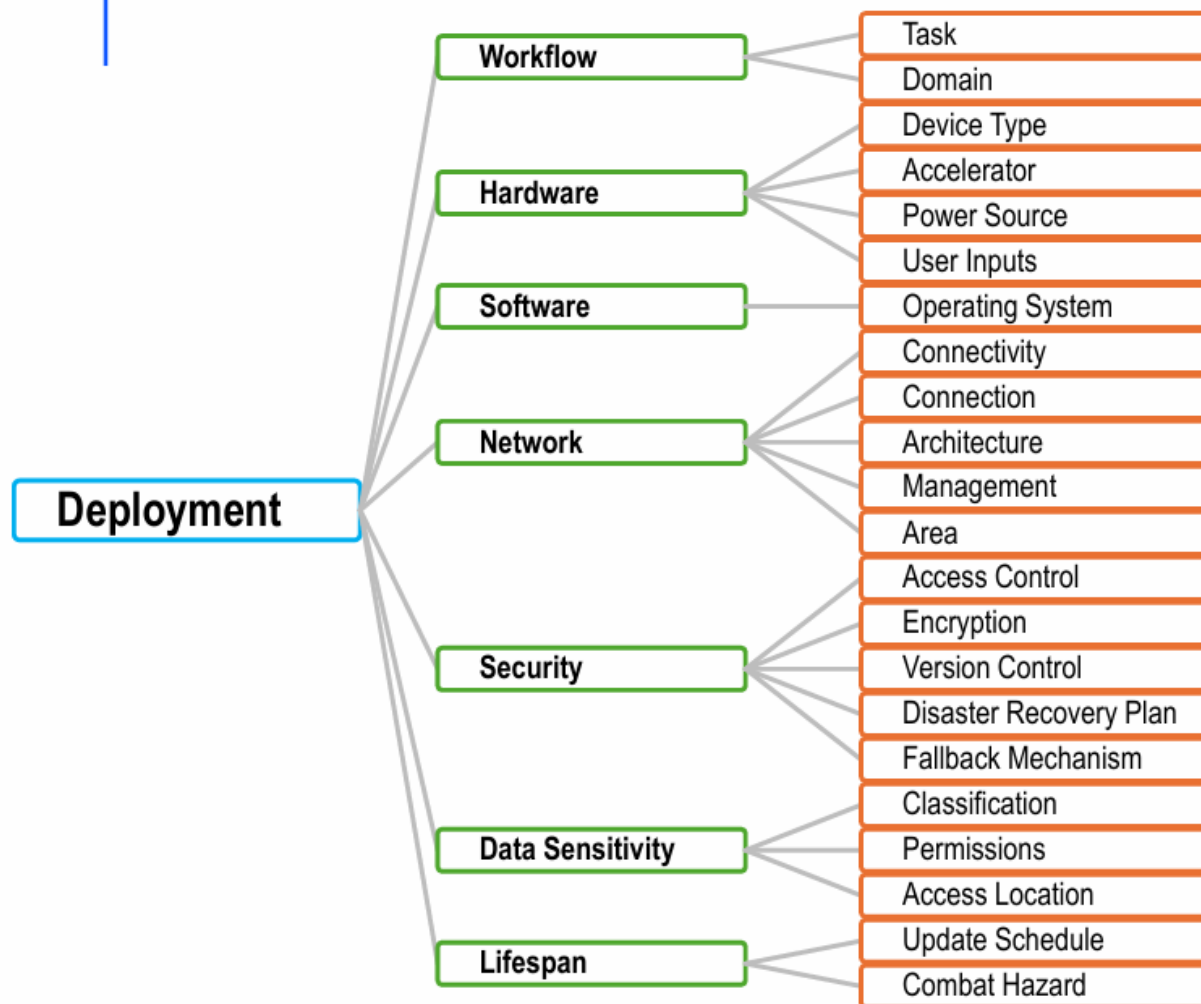
Small Language Models (SLMs) show promise in improving **specialized capabilities** across military-relevant functions under **computational resource and infrastructure constraints** at the tactical edge in warfighting environments. This work addresses the challenge of **assessing whether a model is fit for purpose** and an **evaluation of methods to engineer such a model**.

Part I:
Propose Framework Evaluating
Methods to Engineer
Suited-to-Purpose
SLMs



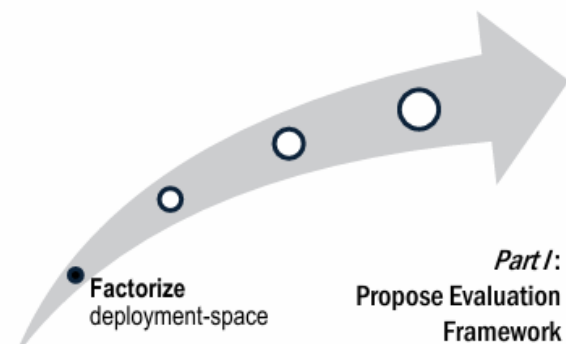
Part II:
Validate Evaluation Framework
Via Surrogate
Experimentation

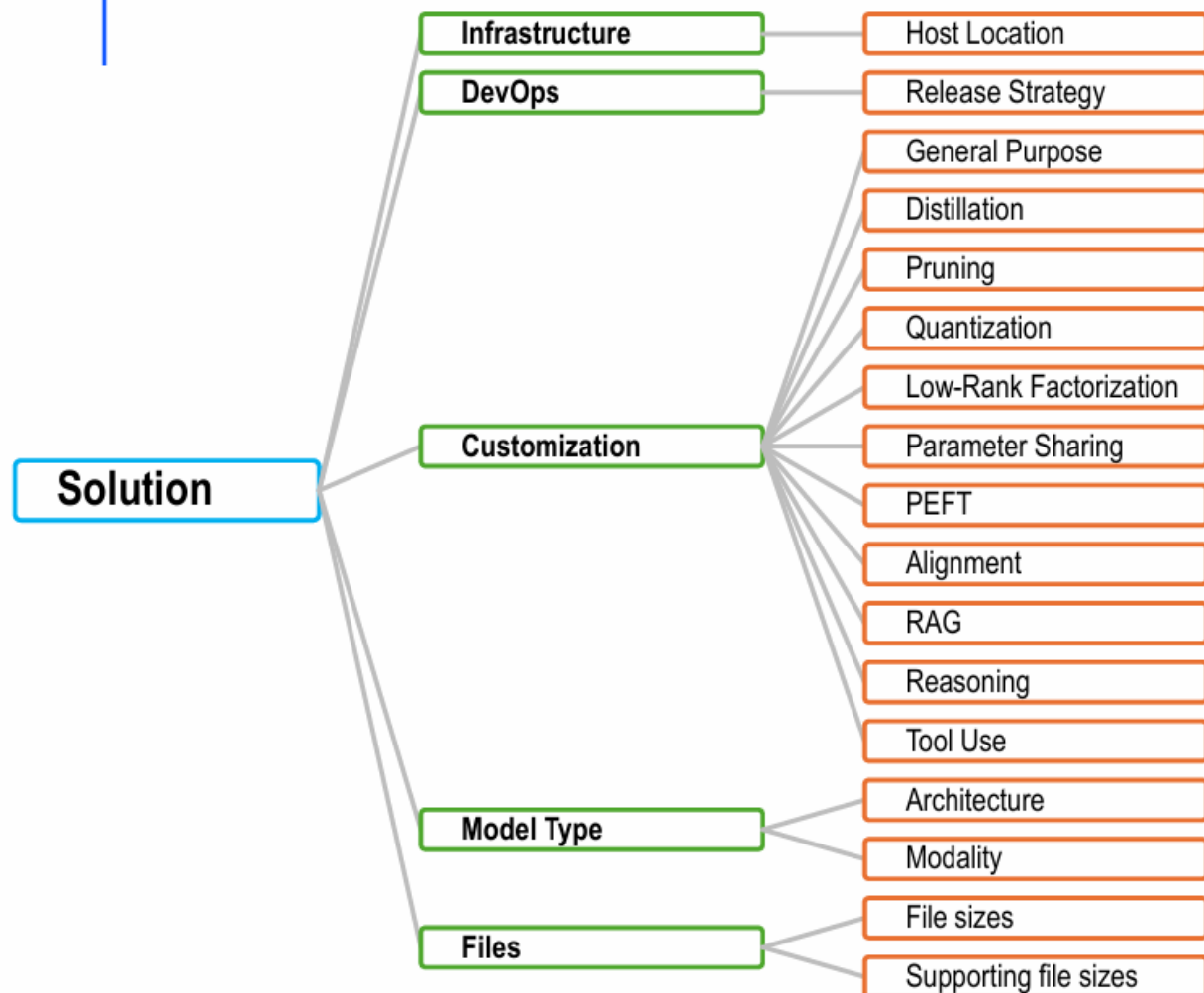




DEPLOYMENT space factors describe the need, including:

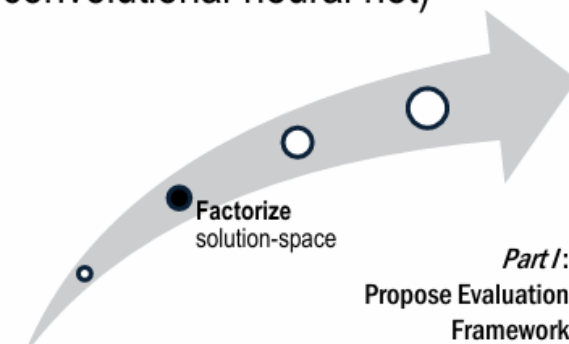
- The **workflow** (task + domain) to which the system will be applied
- The available **hardware, software, and networking**
- **System security** requirements
- **Data sensitivity**
- The **lifespan**, e.g., update schedule & combat hazard





SOLUTION space factors describe the model (i.e., candidate system to meet the need), including:

- **Infrastructure** (e.g., does the model run locally, on-prem, or in the cloud?)
- **DevOps**, including whether new versions are released all at once or incrementally via CI/CD
- **Customization** to the workflow
- **Model type** (e.g., transformer or convolutional neural net)
- **File** requirements and sizes



Property	What to Measure	How to Measure
Correctness	Response provided by solution	(Custom) Benchmark
Correctness	Decision reached by system with user	Oracle test set, human SME evaluator
Mission success	Aggregation of steps to meet goal	Oracle test set, human SME evaluator
Cost	Model cost to use/acquire	Past estimates, test logging, cost calculator
Cost	Model cost to update	Cost calculator
Cost	Human user cost	Past estimates, test logging
Run-time	Latency to run inference, complete task	Test logging
Acquisition Time	Time to receive initial model, update model	Past performance estimate
Computation	Power/battery life, processing load, storage (model, data for prompt/context)	Test logging

Test design and metric collection should be ***tailored to workflow and level of test***

- Correctness metrics will be informed by specific task
- Some measurements (e.g., human SME supported evaluation) may be too resource intensive to apply to all tests

Define
performance
measures &
instrumentation

Part I:

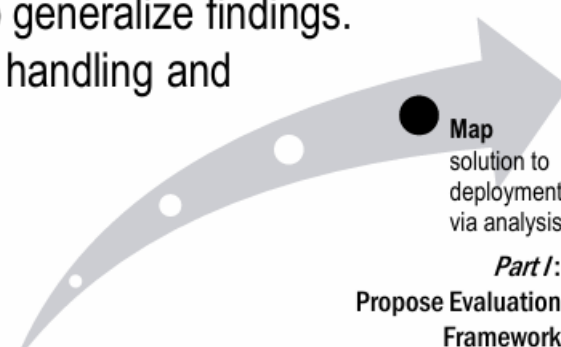
Propose Evaluation
Framework

What would a candidate experimental plan look like?

- **Simple Experimental Plan:** Run experiments on 3 configurations for workflow and compare on measures (e.g., performance, time, cost)
 - 0 – Baseline workflow (no AI)
 - 1 – Baseline workflow plus General-Purpose LLM
 - 2 – Baseline workflow plus Fit-for-Purpose SLM (distilled and/or customized)

What are some challenges with a simple experimental plan?

- **Deployment constraints are important for decision:** platform (1) may not be possible for some deployments and prioritization of different measures in evaluation may vary depending on operational scenario.
- **Domain specificity is important for decision:** experimentation across a variety of domains needed to generalize findings.
- **Fit-to-purpose method matters:** comparison needs to include general purpose w/out specialized data handling and combinations of model reduction and specialization (ideally including sequencing).



Map
solution to
deployment
via analysis

Part I:
Propose Evaluation
Framework

Mission-Relevant Function:

Deployment: Tactical edge (low compute / offline)

Solution: Specialized reasoning-based task



Experimental Surrogate:

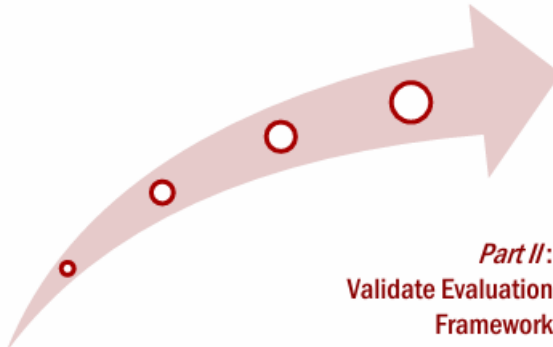
Deployment: SLM training and running locally

Solution: Graduate-level math functions*

- **Benchmark** = prompts & reference texts or “gold standard” responses that can be used to either refine or evaluate models
- To reduce data contamination and prevent need for custom benchmarks, the team used a **published multi-domain benchmark** and selected an **LLM pre-dating that benchmark** from which SLMs were generated

* Previous studies [1] have shown SLMs can lack robustness in solving math problems. It is currently outside of the scope of this experiment to adequately address abstract reasoning but should be noted as part of wider experimentation framework.

Part II:
Validate Evaluation
Framework

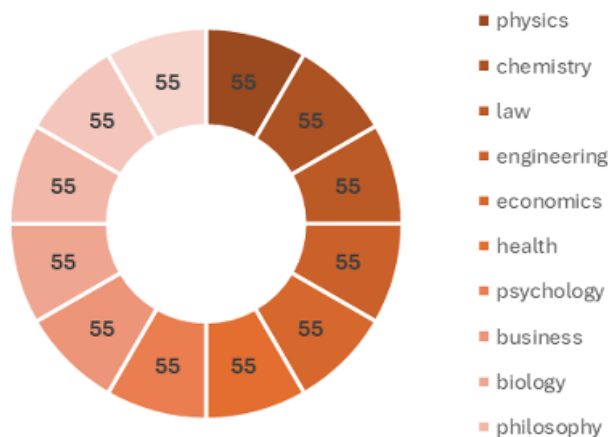


- **Benchmark:** Measuring Massive Multitask Language Understanding (MMLU)-Pro [2]
- **Total No. of Questions:** 12032 | **No. of Answer Choices:** 10 | **No. of Domains:** 14
- **Description:** Graduate-level questions curated from academic exams/textbooks on STEM website and SciBench/TheoremQA benchmarks emphasizing deduction, multi-hop reasoning, and Chain-of-Thought, with additional answer options generated by LLMs verified for plausibility and reasoning depth under expert review

Other benchmark datasets (disjoint from the “all math” training set) included:

- **Physics** (tangentially related domain / numeric)
- **Law** (tangentially related domain / logic)
- **History** (unrelated domain)

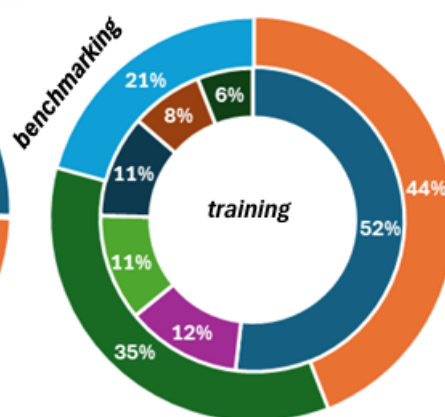
Cross-Domain Unlabeled Dataset (Knowledge Distillation)



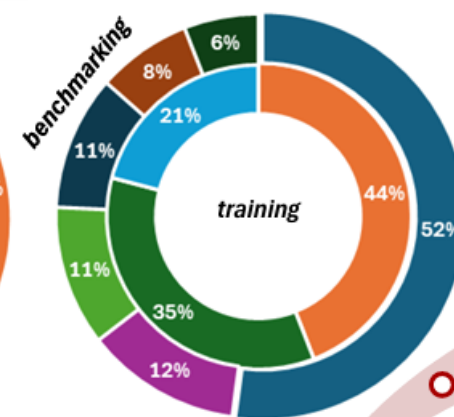
All Math → All Math



Hard Math → Easy Math



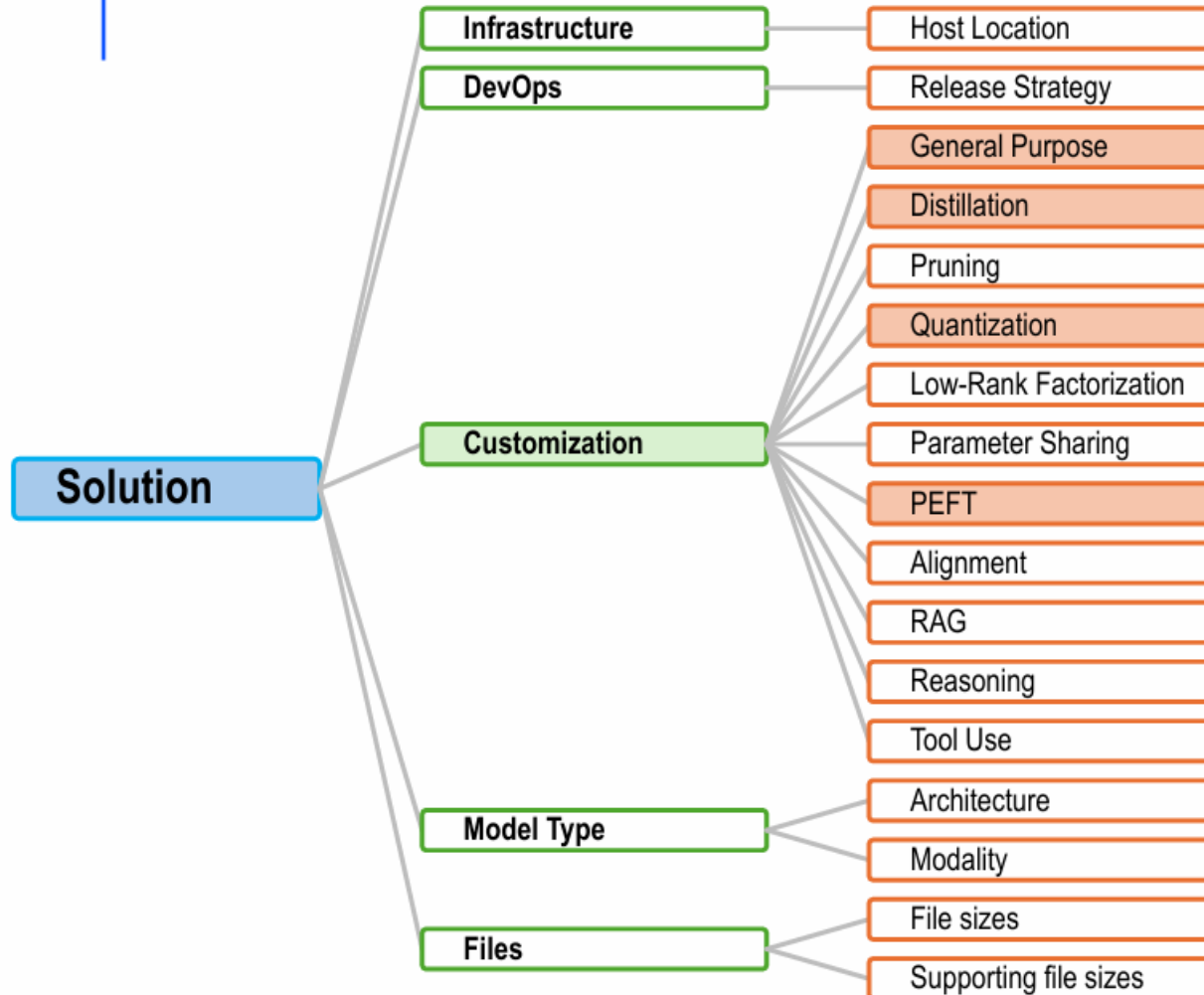
Easy Math → Hard Math



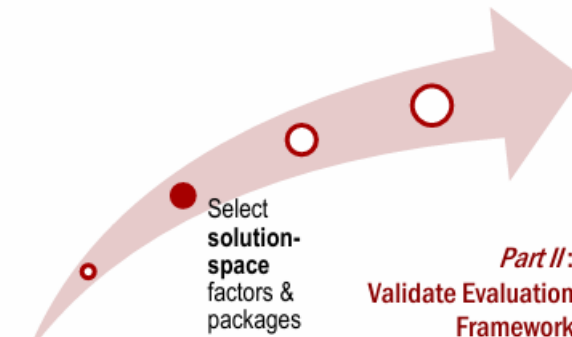
- theoremQA-math
- elem_math
- high_school_math
- high_school_stat
- abstract_algebra
- college_math
- stat
- diff
- calculus

Select deployment-space factors & datasets

Part II:
Validate Evaluation Framework



Customization Technique	Method	Relevant Package(s)
Quantization	Reducing computational/ memory costs while preserving as much accuracy as possible	HuggingFace bitsandbytes
Distillation	Compression via training small student model to mimic behavior of larger teacher model (on cross-domain dataset or task-specific dataset)	HuggingFace Distillation Trainer
Parameter Efficient Fine-Tuning (PEFT)	Adapting pre-trained models to downstream applications	HuggingFace PEFT, Trainer, Transformers



- Initial experiments conducted in secure environment leveraging Virtual Desktop Infrastructure (VDI) on GPU-accelerated JupyterHub instance
 - Contains Python 3.13 and PyTorch 2.2.1 / CUDA 12.3
 - Connects to 20GB GPU x 1
- Code implementation for lab-based experiments focused on the following measures:

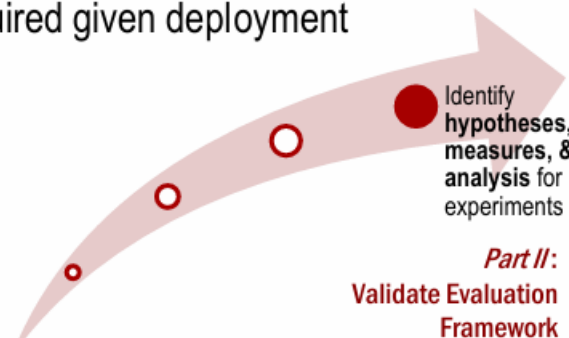
Property	What to Measure	How to Measure
Correctness	Response provided by solution	Number of multiple-choice questions answered correctly (i.e., accuracy)
Run-time	Latency to run inference, complete task	Inference time via <code>perf_counter()</code> in Python built-in <code>time</code> module
Acquisition Time	Time to receive initial model, update model	Surrogate measure time to generate SLM (e.g., fine-tuning, quantization) via <code>perf_counter()</code>
Computation	Processing load, storage (model, data for prompt/context)	Logging metrics outputted by HuggingFace Trainer (e.g., <code>num_tokens</code>)

Build
experimental
infrastructure
for lab-based
experiments

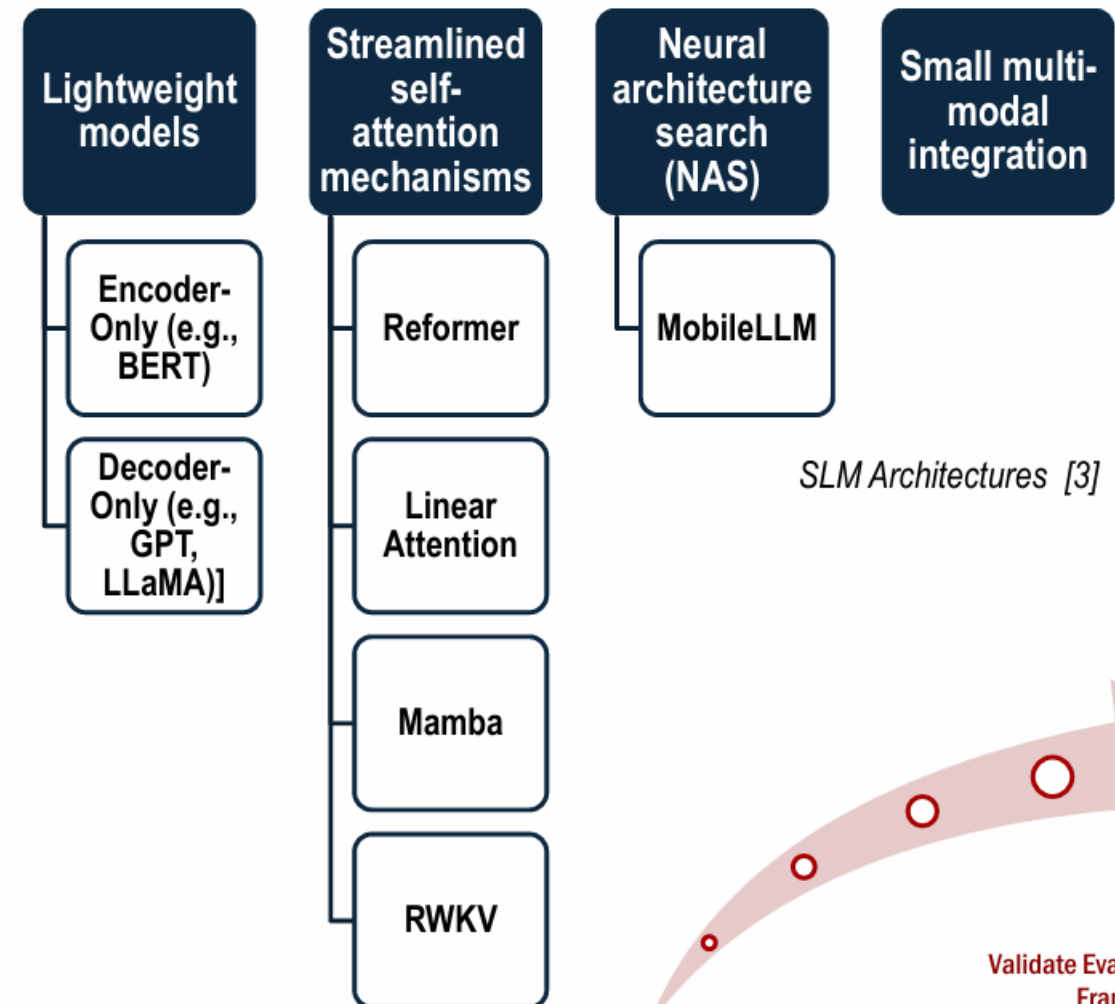
Part II:
Validate Evaluation
Framework

What would the surrogate experimental plan look like?

- **Experimental Plan:** Run experiments on configurations for workflow & compare on identified
 - 0 – Baseline workflow (subject matter expert, no AI)
 - 1 – Baseline workflow plus General-Purpose LLM (GPT-3.5Turbo or GPT-OSS)
 - 2 – Baseline workflow plus **Fit-for-Purpose SLM** (GPT2-large), engineered via:
 - 2a – Quantization
 - 2b – Distillation (cross-domain)
 - 2c – Distillation (task-specific)
 - 2d – PEFT
 - 2e – Quantization -> PEFT
 - 2f – PEFT -> Quantization
 - 2g – Distillation -> Quantization
 - 2h – Quantization -> Distillation
 - 2i – Quantization -> Distillation -> PEFT
 - 2j – Quantization -> PEFT -> Distillation
 - 2k – PEFT -> Quantization -> Distillation
- **Hypotheses:**
 - Adding AI (1 or 2) to Baseline Workflow (0) will increase cost but decrease time, ideally with same performance
 - A custom model (2) will cost more than a general-purpose model (1), but improve on other metrics
 - A reduced model (2) will have worse performance across a variety of tasks than general purpose (1) but is required given deployment constraints (network connectivity, compute)
 - A custom model (2) will have better performance for domain specific tasks than general purpose (1)
 - Distillation and fine-tuning method techniques will cost more than quantization techniques but may result in better performance for specialized workflow
 - Quantization -> PEFT will be cheaper and lower in performance compared to PEFT -> Quantization.



- 1) **The selected SLM architecture should be suited for the task.**
 - For example, question-answering tasks typically exhibit best performance with encoder-decoder models.
- 2) **Limitations in testing environments (concurrent with or separate from deployment factors) can limit adequate assessment of solution factors.**
 - Surrogate experimentation on VDI infrastructure reached performance limitations when fine-tuning (via PEFT) to generate SLMs.
 - Training and running inference on SLMs may require a different set of factors, but **both** should be considered as part of defining factors for evaluation.
 - E.g. Tactical edge systems might have to ingest live operational data and either update onsite (in limited resource environment) or offsite (with more compute)



Part II:
Validate Evaluation
Framework

3) Fine-tuning techniques for multiple-choice questions can vary dependent on the workflow need, warranting further study.

- For the surface-level experimentation performed here, the PEFT training dataset was multiple-choice mathematics questions. Two candidate approaches were set up in code:
 - AutoModelForCausalLM** (text generation)
 - AutoModelForMultipleChoice** (question answering)
- Possibility exists to train on text generation (i.e., provide answer to the math question without candidate options) and benchmark on multiple choice question answering

```
{
  "role" : "system",
  "content" :
    "You are a knowledge expert, you are supposed
    to answer the multi-choice question to derive
    your final answer as 'The answer is ... `.'"
},
{
  "role" : "user",
  "content" :
    "Q: What is 2 + 2?
    Options are:
    A : 5
    B : 4
    ...
    J : 10"
}
```

GPT4o Prompt Template

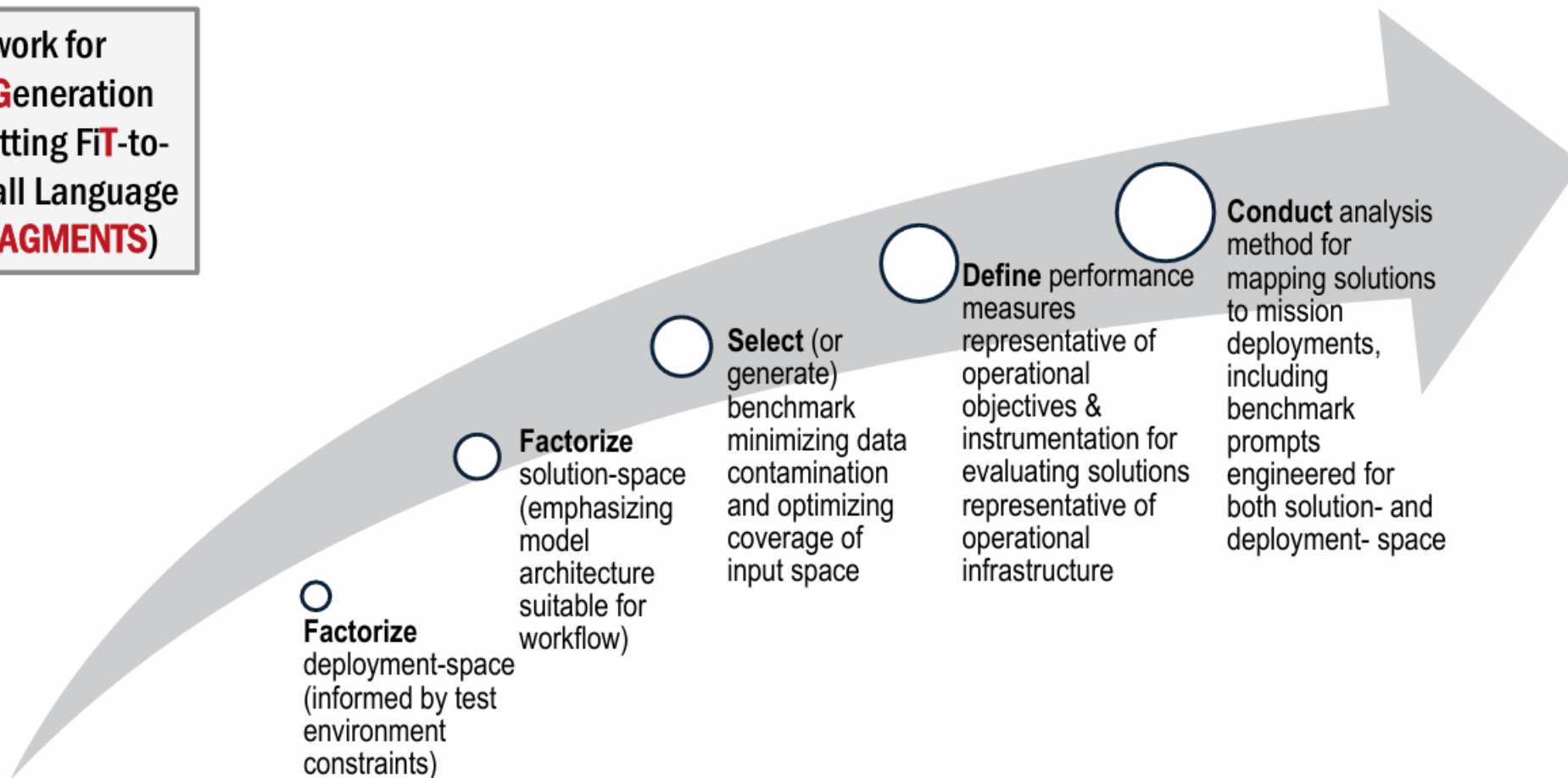
from https://huggingface.co/datasets/TIGER-Lab/MMLU-Pro/blob/main/run_gpt4o.py

4) Further prompt engineering may be required for best results across customization techniques.

- Different model families have different templates for question-answering prompting.

Part II:
Validate Evaluation
Framework

FRamework for
Assessing **G**eneration
Methods **N**etting **F**IT-to-
Purpose **S**mall Language
Models (**FRAGMENTS**)



References

- [1] S. Gao, A. Bosselut, S. Bengio, and E. Abbe, “AbstRaL: Augmenting LLM’s Reasoning by Reinforcing Abstract Thinking”, arXiv [cs.CL]. 2026.
- [2] Y. Wang et al., “MMLU-Pro: A More Robust and Challenging Multi-Task Language Understanding Benchmark”, arXiv [cs.CL]. 2024.
- [3] T. Sakib, M. Hosain, and M. Morol, “Small Language Models: Architectures, Techniques, Evaluation, Problems and Future Adaptation”, arXiv [cs.CL]. 2025

Presenter: Emma Meno
Co-authors: Daniel Sobien, M. Alexander Kyer, Justin Krometis, Erin Lanus, Laura Freeman
Organization: Virginia Tech National Security Institute