CS 1671 / CS 2071 / ISSP 2071
Human Language Technologies

Session 22: Exam review

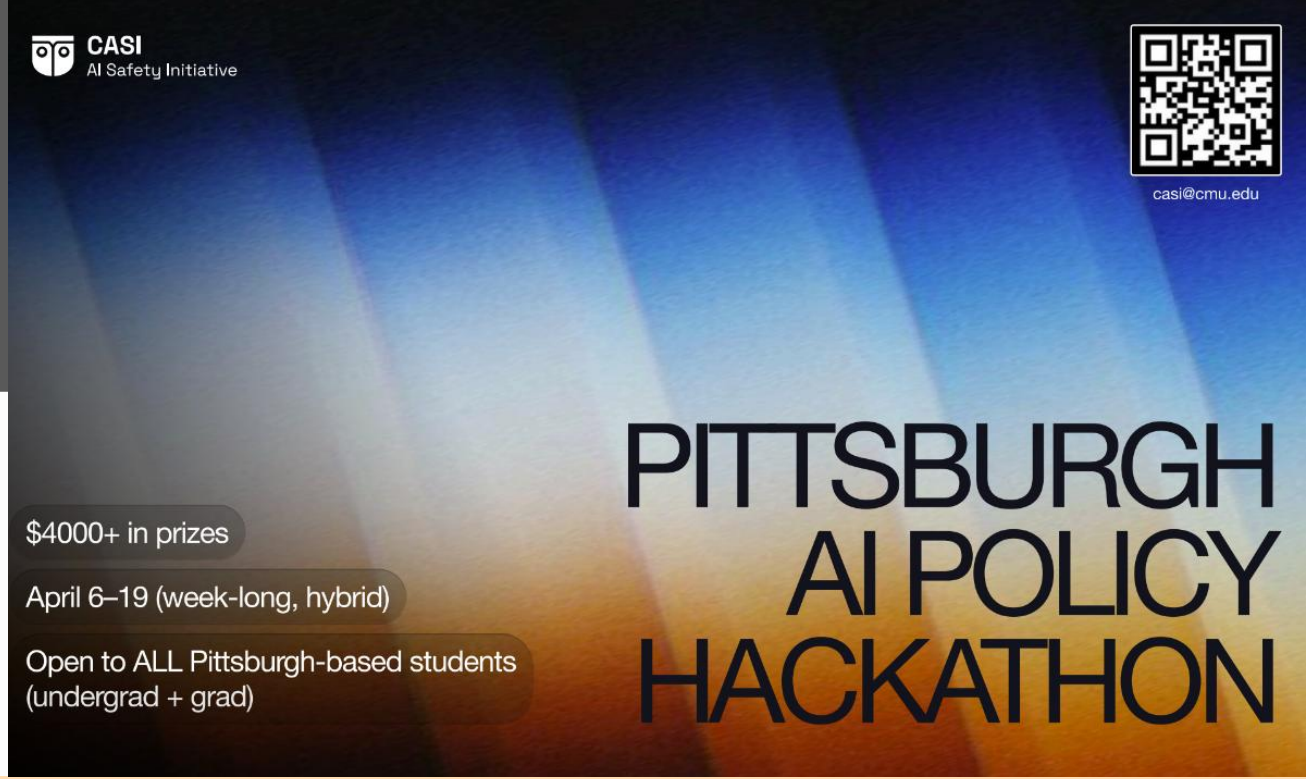
Michael Miller Yoder

April 6, 2026

School of Computing and Information

AI policy hackathon from Carnegie AI Safety Initiative

- Register by today, Apr 6
- https://luma.com/cmfkkf8z for info and registration



Wed, Apr 15: Policy brief submission deadline (11:59pm ET).

Sun, Apr 19: Final presentation day on CMU's campus (in-person). Selected teams present to judges, followed by lunch, networking, and a prize ceremony.

Assessments: exam

- In-person exam is in class this Wed Apr 8
- Covers all material up to this point
  - Content in Module 1 not directly covered
- Paper exam
- True/false questions, short answer, and long answer to work through problems
- One page of double-sided notes will be permitted
- Some formulas will be provided with the exam
- Some math calculations. Can bring a calculator if you want, but it's fine to leave things in fractional form.

Assessments: homework

- Homework 3 is **due next Thu Apr 16**
- Run Jupyter notebooks from templates on the CRCD
- Part 1: LLM prompting
  - Use CRCD CPUs
- Part 2: Instruction tuning of an LLM
  - Use CRCD GPUs

Overview: Exam review

- **Your questions: ask me anything**
- Go through high-level concepts from Modules 2-4
    - Feel free to ask questions throughout

Structure of this course

MODULE 1	Prerequisite skills for NLP	text normalization, linear alg., prob., machine learning

	Approaches	How text is represented	NLP tasks
MODULE 2	statistical machine learning	n-grams	language modeling text classification
MODULE 3	neural networks	static word vectors	text classification
MODULE 4	transformers and LLMs	contextual word vectors	language modeling text classification

MODULE 5	NLP applications and ethics	

Questions?

Module 2: N-grams and statistical NLP

Module 2 N-grams and statistical NLP: how text is represented

- n-grams
- term-document matrices
- term-term matrices

Module 2 N-grams and statistical NLP: algorithms

- N-gram language modeling
- Logistic regression for text classification
    - Parameters (one for every feature) learned with stochastic gradient descent

Practice question: N-grams and statistical NLP algorithms

- Find the formula for predicting the next word using a trigram language model. Explain each part, including what it refers to and how it is calculated from data.

Module 3: Neural networks and word2vec

Module 3 Neural networks and word2vec: how text is represented

- Dense word embeddings (vectors), learned with e.g. word2vec
- Word2vec
  - Logistic regression to classify words as occurring together or not
    - Positive examples: words that occur together in a corpus within a context window
    - Negative examples: random words with target word
    - Example from a part of a corpus: “the dog barked two times”. Target word is “dog”
      - Positive example: (dog, barked)
      - Negative example: (dog, interstellar)
  - From randomly initialized word vectors, moves vectors for words that co-occur together closer in vector space

Module 3 Neural networks and word2vec: algorithms

- Feedforward neural networks for text classification
  - Parameters learned from stochastic gradient descent

Practice question: Neural networks and word2vec

- Draw a diagram of a neural network with a size and number of layers of your choice. The input should be word2vec embeddings.
  - Label each part and describe how it is computed

Module 4: Transformers and LLMs

Module 4 Transformers and LLMs: how text is represented

- Contextual word embeddings: different vector (embedding) for every token
- Text is first tokenized into subword tokens
- Embedding for each word type + embedding for position in the sentence

Module 4 Transformers and LLMs: algorithms

- Transformer models and self-attention
  - One output vector for every input token after many transformations
  - Output vectors incorporate information from other words in a sentence through self-attention
- Pretrained transformer-based models: LLMs
  - Decoder-only models trained on (causal, left-to-right) language modeling: GPT series models
  - Encoder-only models trained on masked language modeling: BERT family of models
  - Encoder-decoder models first encode an input sentence to a vector and then use that as input to start doing language modeling (decoding) to produce an output sentence

Module 4 Transformers and LLMs: algorithms

- Finetune pretrained models for specific tasks
  - For language modeling
    - Continued pretraining
    - Instruction finetuning (SFT)
    - Parameter-efficient finetuning (LoRA)
  - For classification
    - Finetuning masked language models
- Prompting: zero-shot, few-shot, chain-of-thought

Practice questions: Transformers and LLMs

1. Describe the goal of self-attention in transformers. What does the output of self-attention look like, and what information does that output contain?
2. You are asked to design an LLM that can answer questions about a company's codebase. What approach would you take, and what data would that approach rely on?

Questions?

Best of luck on the exam!