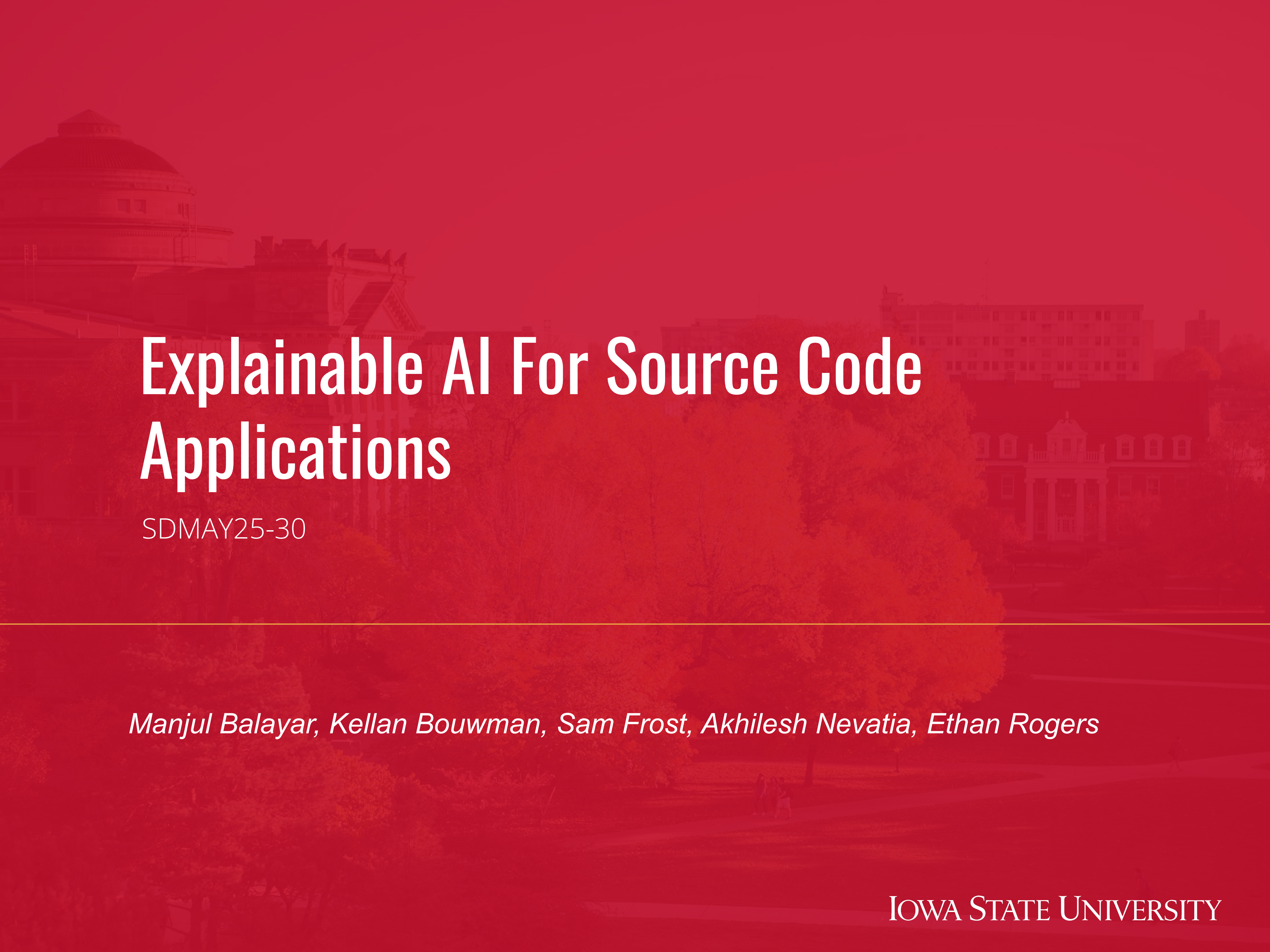


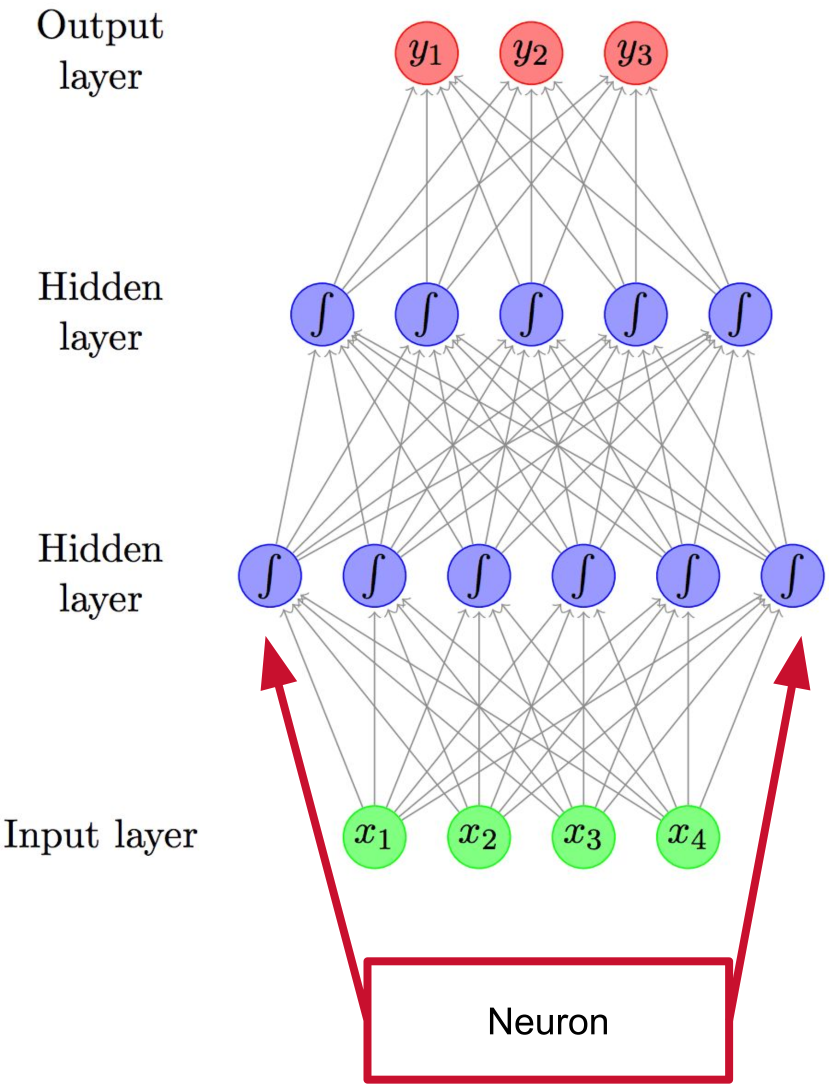
Explainable AI For Source Code Applications

SDMAY25-30

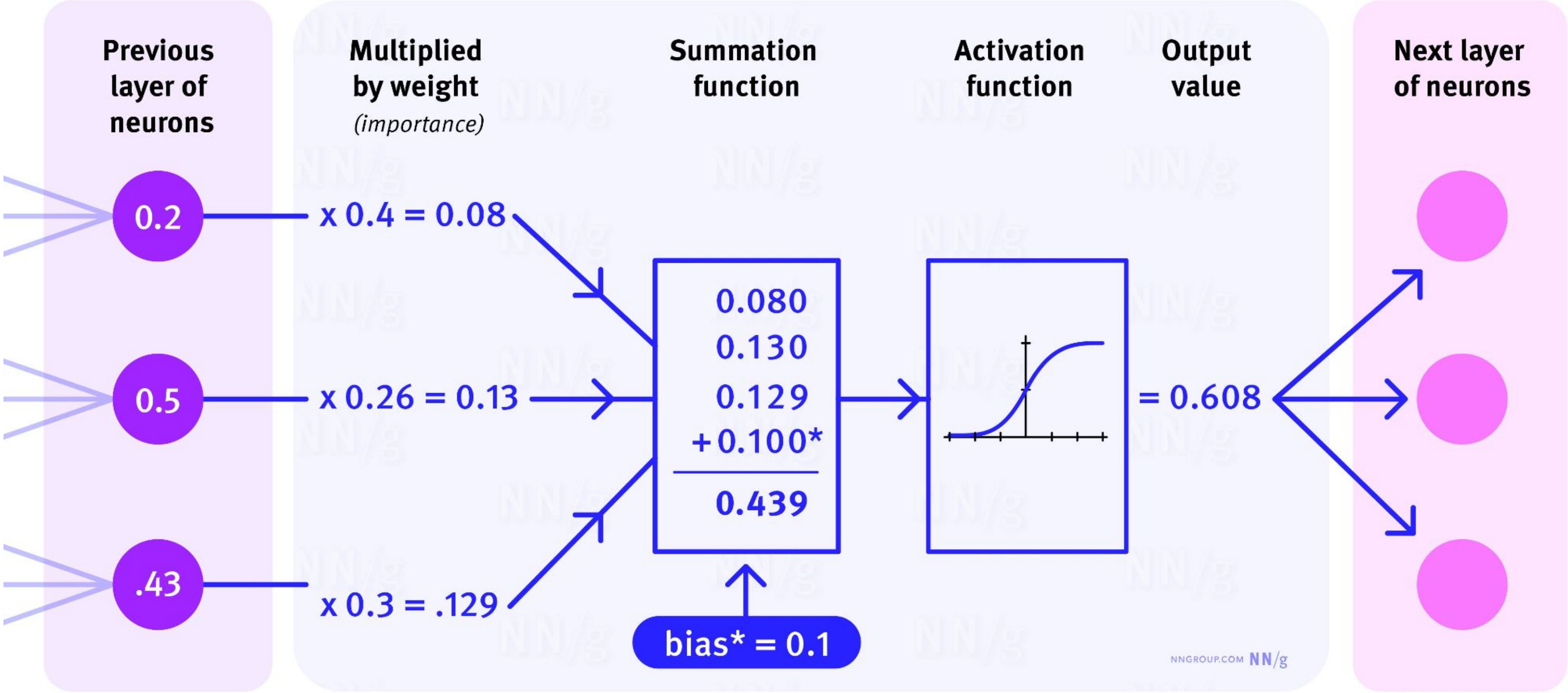
Manjul Balayar, Kellan Bouwman, Sam Frost, Akhilesh Nevatia, Ethan Rogers

IOWA STATE UNIVERSITY

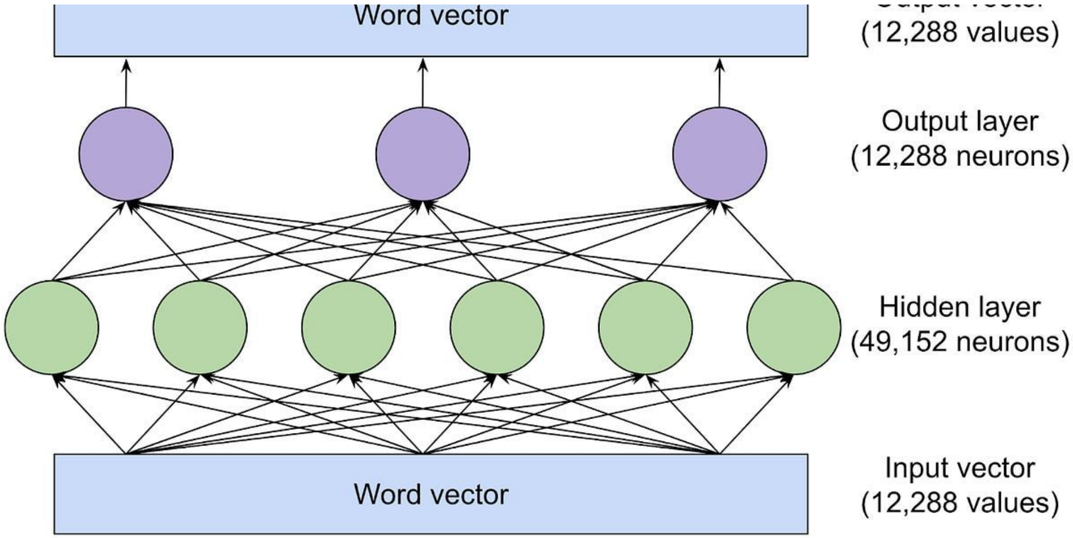
- Machine Learning / Statistical Learning:
  - Machine learning: A broad discipline that focuses on how computers can learn from data
  - Statistical learning: A branch of artificial intelligence that focuses on turning raw data into actionable information. Statistical learning theory is a framework that uses statistical and functional analysis to build models that can make predictions and draw conclusions from data
- Neuron:
  - A collection of a set of inputs, a set of weights, and an activation function.
  - It translates these inputs into a single output.
- Deep Learning:
  - Type of machine learning based on artificial neural networks in which multiple layers of processing are used to extract progressively higher level features from data

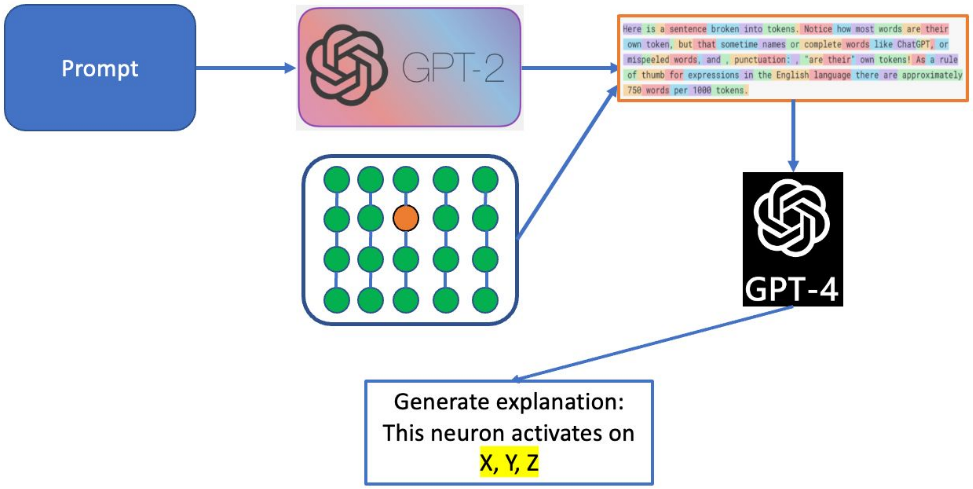


How a Single Artificial Neuron Works



- LLM:
  - Large Language Model
  - Form of Deep learning on NLP (Natural Language Processing)
- Generative AI:
  - A type of AI that uses generative models to create new content, such as text, images, videos, music, and audio
  - Based off an LLM (example: ChatGPT3.5)
- Neuron Activation
  - Non-linear function that we apply over the input data coming to a particular neuron and the output from the function will be sent to the neurons present in the next layer as input
  - A Path of Neurons





Here is a sentence broken into tokens. Notice how most words are their own token, but that sometime names or complete words like ChatGPT, or mispeeled words, and , punctuation: , "are their" own tokens! As a rule of thumb for expressions in the English language there are approximately 750 words per 1000 tokens.

Generate explanation:
This neuron activates on
X, Y, Z

Project Overview

- Client
    - Dr. Ali Jannesari/ISU SwAPP Lab
- Abstract
    - Focus on auto-labeling code datasets using AST tools, regular expressions, and LLM-generated labels.
- Goal
    - Deepen the understanding of LLMs and generative AI by analyzing neuron activations and applying metrics and heuristics. The project will also unify two existing code bases into a flexible and scalable framework that can be deployed seamlessly across Colab, local environments, and HPC clusters.

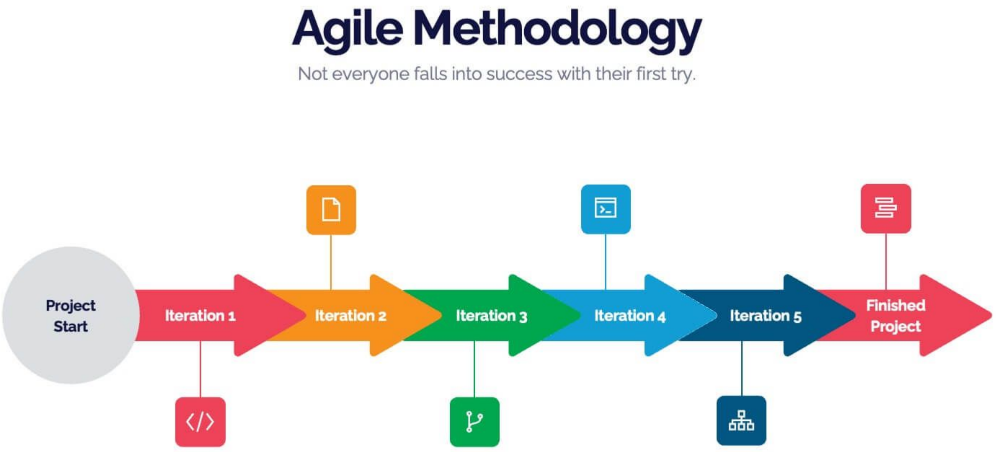
Project Overview - Continued

- Users
  - Researchers
    - ML Engineers / Researchers
    - Prompt Engineers / Researchers
    - Computer Scientists
  - Students
    - Graduate Students
    - Undergraduate Students
  - Industry Professionals
- We aim to include students, researchers, and industry professionals as key users. Our documentation will prioritize accessibility and clarity for all experience levels, while the codebase will remain flexible and scalable to meet both academic and industry needs.

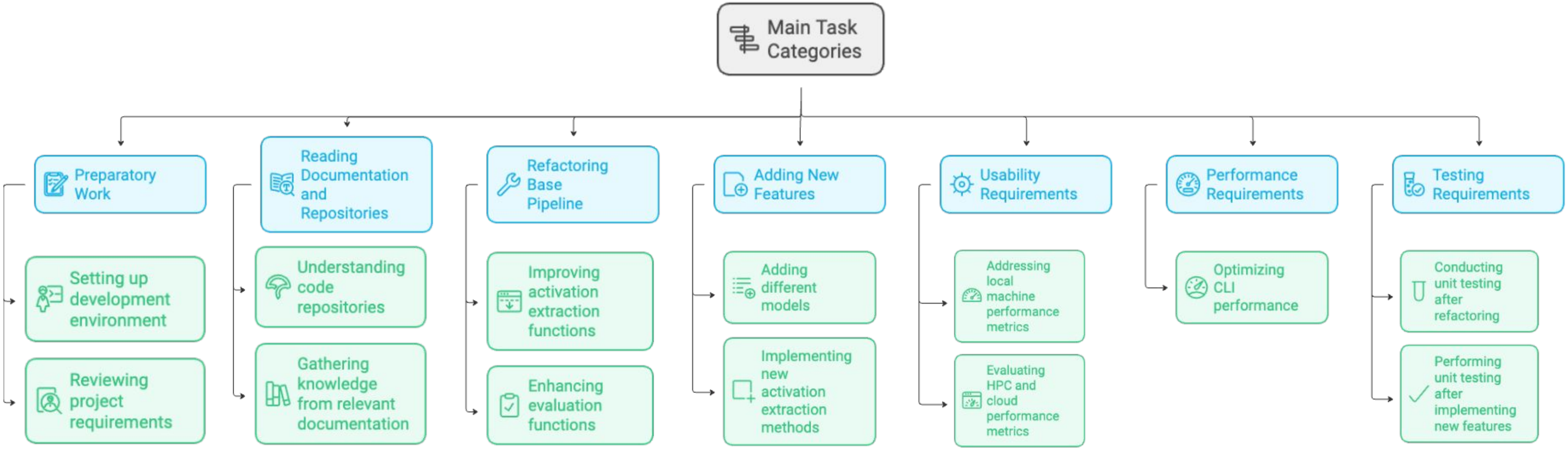
Project Management Style

Agile

- Flexibility
    - Adjust to new requirements and emerging technologies.
- Collaboration
    - Frequent feedback loops with stakeholders.
- Familiarity



Task Decomposition



Main Task Categories

Preparatory Work
- Setting up development environment
- Reviewing project requirements

Reading Documentation and Repositories
- Understanding code repositories
- Gathering knowledge from relevant documentation

Refactoring Base Pipeline
- Improving activation extraction functions
- Enhancing evaluation functions

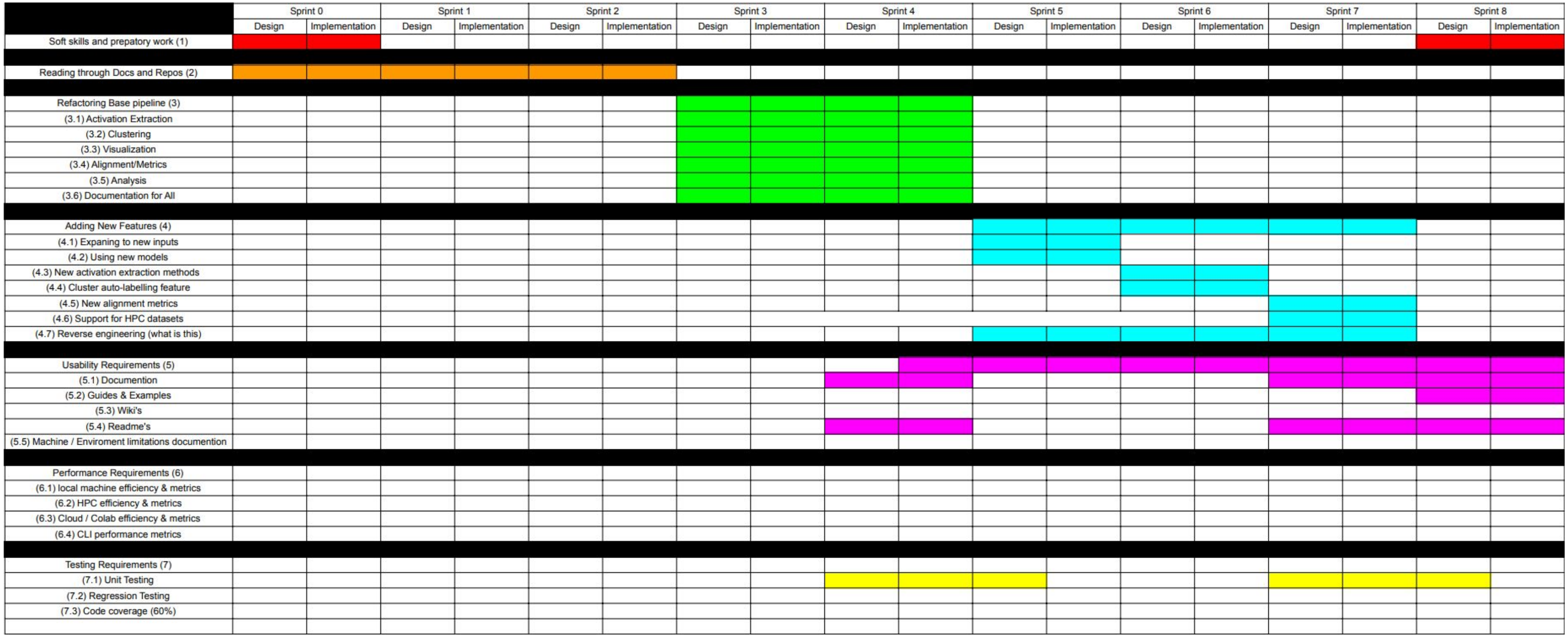
Adding New Features
- Adding different models
- Implementing new activation extraction methods

Usability Requirements
- Addressing local machine performance metrics
- Evaluating HPC and cloud performance metrics

Performance Requirements
- Optimizing CLI performance

Testing Requirements
- Conducting unit testing after refactoring
- Performing unit testing after implementing new features

Task Decomposition - Gantt Chart



	Sprint 0		Sprint 1		Sprint 2		Sprint 3		Sprint 4		Sprint 5		Sprint 6		Sprint 7		Sprint 8	
	Design	Implementation	Design	Implementation	Design	Implementation	Design	Implementation	Design	Implementation	Design	Implementation	Design	Implementation	Design	Implementation	Design	Implementation
Soft skills and prepatory work (1)	🟥	🟥															🟥	🟥
Reading through Docs and Repos (2)	🟧	🟧	🟧	🟧	🟧	🟧												
Refactoring Base pipeline (3)							🟩	🟩	🟩	🟩								
(3.1) Activation Extraction							🟩	🟩	🟩	🟩								
(3.2) Clustering							🟩	🟩	🟩	🟩								
(3.3) Visualization							🟩	🟩	🟩	🟩								
(3.4) Alignment/Metrics							🟩	🟩	🟩	🟩								
(3.5) Analysis							🟩	🟩	🟩	🟩								
(3.6) Documentation for All							🟩	🟩	🟩	🟩								
Adding New Features (4)											🟦	🟦	🟦	🟦	🟦	🟦		
(4.1) Expaning to new inputs											🟦	🟦						
(4.2) Using new models											🟦	🟦						
(4.3) New activation extraction methods													🟦	🟦				
(4.4) Cluster auto-labelling feature													🟦	🟦				
(4.5) New alignment metrics															🟦	🟦		
(4.6) Support for HPC datasets															🟦	🟦		
(4.7) Reverse engineering (what is this)											🟦	🟦	🟦	🟦	🟦	🟦		
Usability Requirements (5)										🟪	🟪	🟪	🟪	🟪	🟪	🟪	🟪	🟪
(5.1) Documention									🟪	🟪					🟪	🟪	🟪	🟪
(5.2) Guides & Examples																	🟪	🟪
(5.3) Wiki's																		
(5.4) Readme's									🟪	🟪					🟪	🟪	🟪	🟪
(5.5) Machine / Enviroment limitations documention																		
Performance Requirements (6)																		
(6.1) local machine efficiency & metrics																		
(6.2) HPC efficiency & metrics																		
(6.3) Cloud / Colab efficiency & metrics																		
(6.4) CLI performance metrics																		
Testing Requirements (7)																		
(7.1) Unit Testing									🟨	🟨	🟨				🟨	🟨	🟨	
(7.2) Regression Testing																		
(7.3) Code coverage (60%)																		

Milestones, Metrics, and Evaluation Criteria

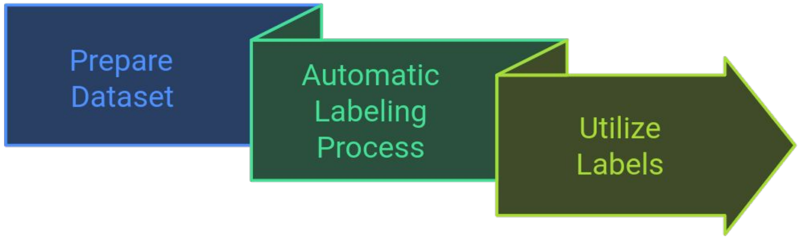
Semester 1

- Months 1-2

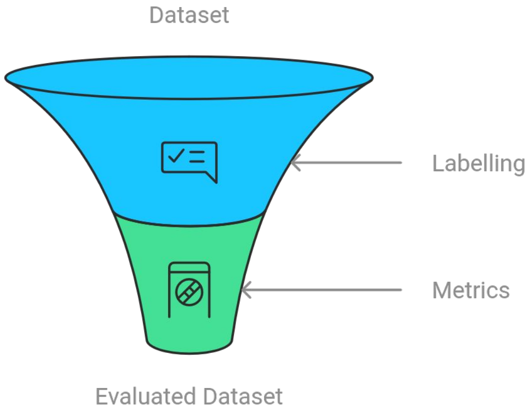
  Dataset preparatory pipeline and automatic labelling

  - Deliverable: Functional pipeline and annotated datasets (java, CUDA, etc.)
- Months 3-4
  - Preliminary Evaluation Setup
  - Deliverable: Initial evaluation metrics

Auto-Labeling Pipeline Implementation



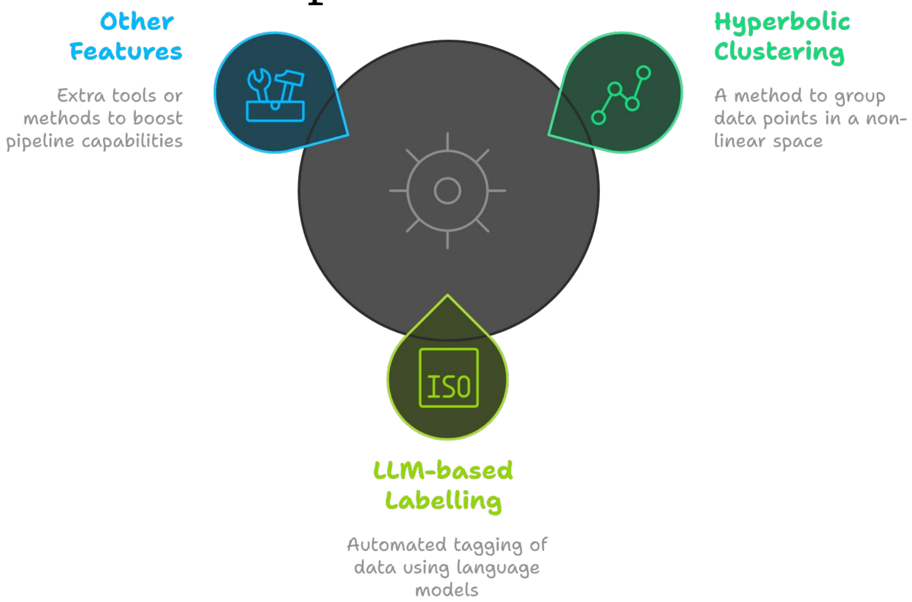
Evaluation Process of Generated Labels



Milestones, Metrics, and Evaluation Criteria

Semester 2

- Months 5-7
  - Full Evaluation with Multiple Datasets
  - Deliverable: Comprehensive evaluation report.
- Months 6-8
  - Additional Features
  - Deliverable: Tested Added Functionality
- Months 7-8
  - Final Report and Presentation
  - Deliverable: Final documentation and presentation.



Other Features

Extra tools or methods to boost pipeline capabilities

Hyperbolic Clustering

A method to group data points in a non-linear space

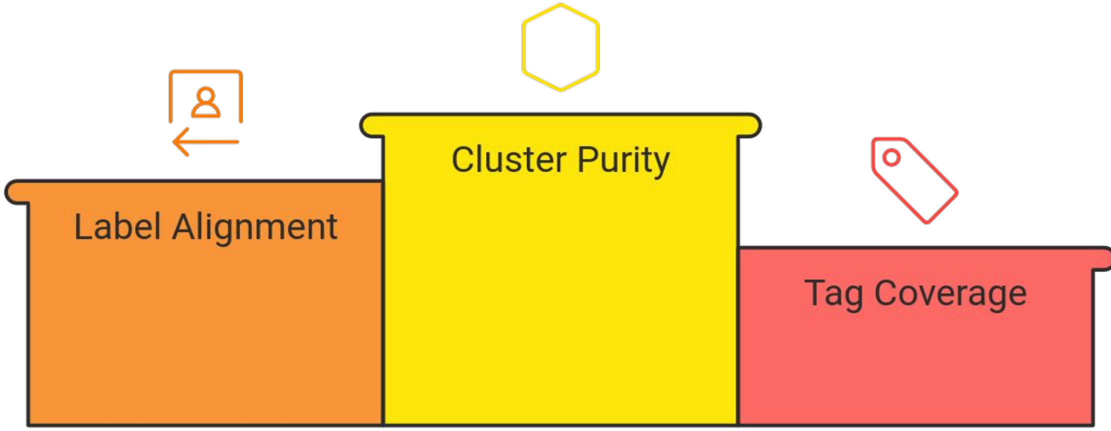
LLM-based Labelling

Automated tagging of data using language models

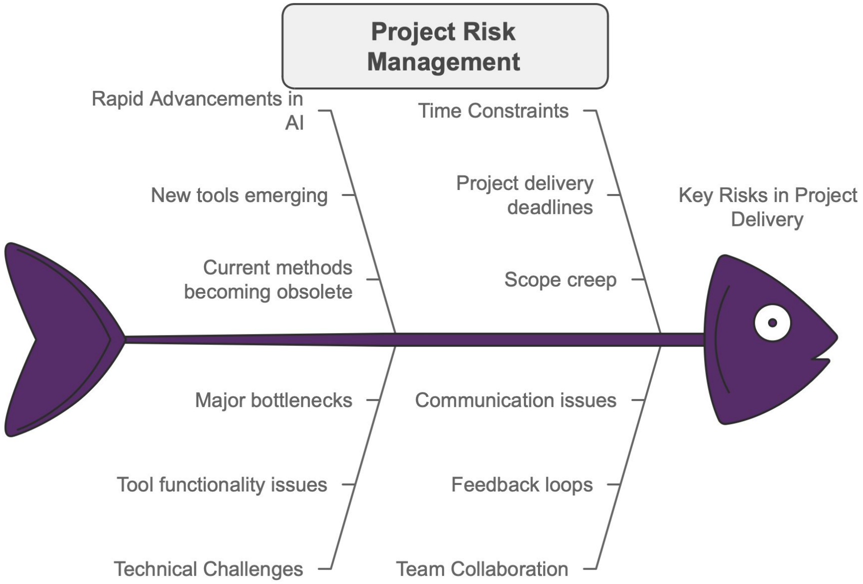
Milestones, Metrics, and Evaluation Criteria

Evaluation Metrics

- Tag Coverage
  - Percentage of code elements correctly labeled.
- Alignment Score
  - How well clusters match auto-labeled tags.
  - Metrics: Precision, Recall, F1 Score.
- Cluster Quality
  - Purity: Cluster item similarity



Risks and Risk Mitigation



Project Risk Management

Rapid Advancements in AI

New tools emerging

Current methods becoming obsolete

Time Constraints

Project delivery deadlines

Scope creep

Key Risks in Project Delivery

Major bottlenecks

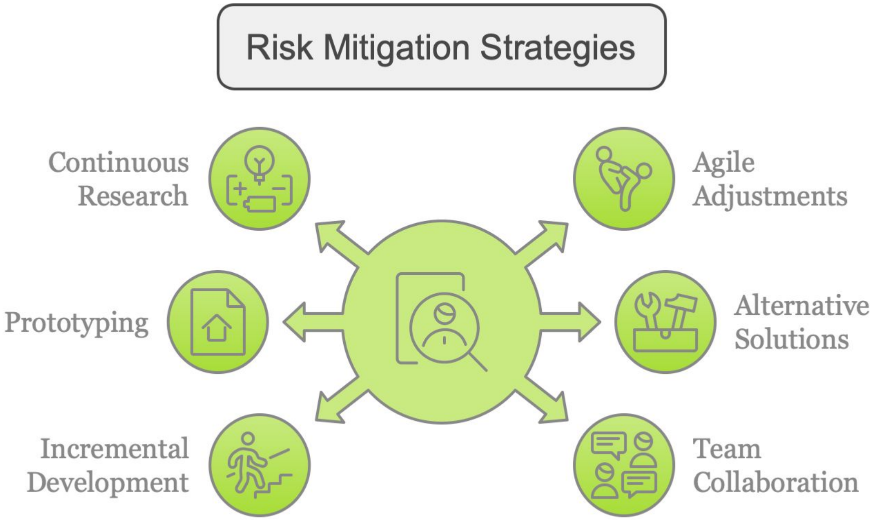
Tool functionality issues

Technical Challenges

Communication issues

Feedback loops

Team Collaboration



Risk Mitigation Strategies

Continuous Research

Agile Adjustments

Prototyping

Alternative Solutions

Incremental Development

Team Collaboration

Risks and Risk Mitigation

- Risk 1: Rapid Advancements in AI Making Work Redundant
- Mitigation:
  - Continuous Research:
    - Regularly survey latest publications
    - Stay updated with latest Research and Developments, by discussing the same in weekly meetings
  - Agile Adjustments:
    - Helps Pivot focus Quickly if an emerging tool proves to be more beneficial
    - If current approach becomes obsolete

Risks and Risk Mitigation

- Risk 2: Technical Challenges with AST Tools or LLMs
- Mitigation:
  - Prototyping:
    - Early testing of tools and functionality
    - Working on major bottlenecks identified first
  - Alternative Solutions:
    - Keeping backup options ready, such as alternative tools / methods.

Risks and Risk Mitigation

- Risk 3: Time Constraints for Project Delivery
- Mitigation:
  - Incremental Development:
    - Sticking with an agile approach with defined sprints
    - Frequent deliverables to manage progress effectively
  - Team Collaboration:
    - Ensuring Recurring Feedback loops with Clients and Team-members
    - Helps resolve issues early and avoid scope creep

Conclusion

- Enhancing AI model interpretability for code through auto-labeling and AST tools.
- Addressing key risks with agile development and robust risk mitigation strategies.
- Building scalable solutions for future research and practical applications in code analysis