Role of AI in Software Industry

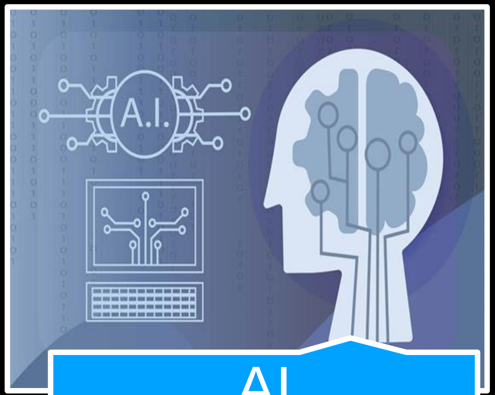
Dr. Sridhar Chimalakonda
Department of Computer Science & Engineering
Indian Institute of Technology, Tirupati, India
ch@iittp.ac.in

RISHA →

Research in Intelligent Software & Human Analytics Lab

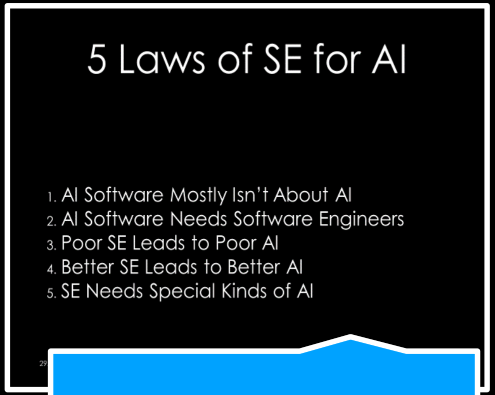
What's the goal of today's talk?

AI Today



AI Fundamentals

AI4SE



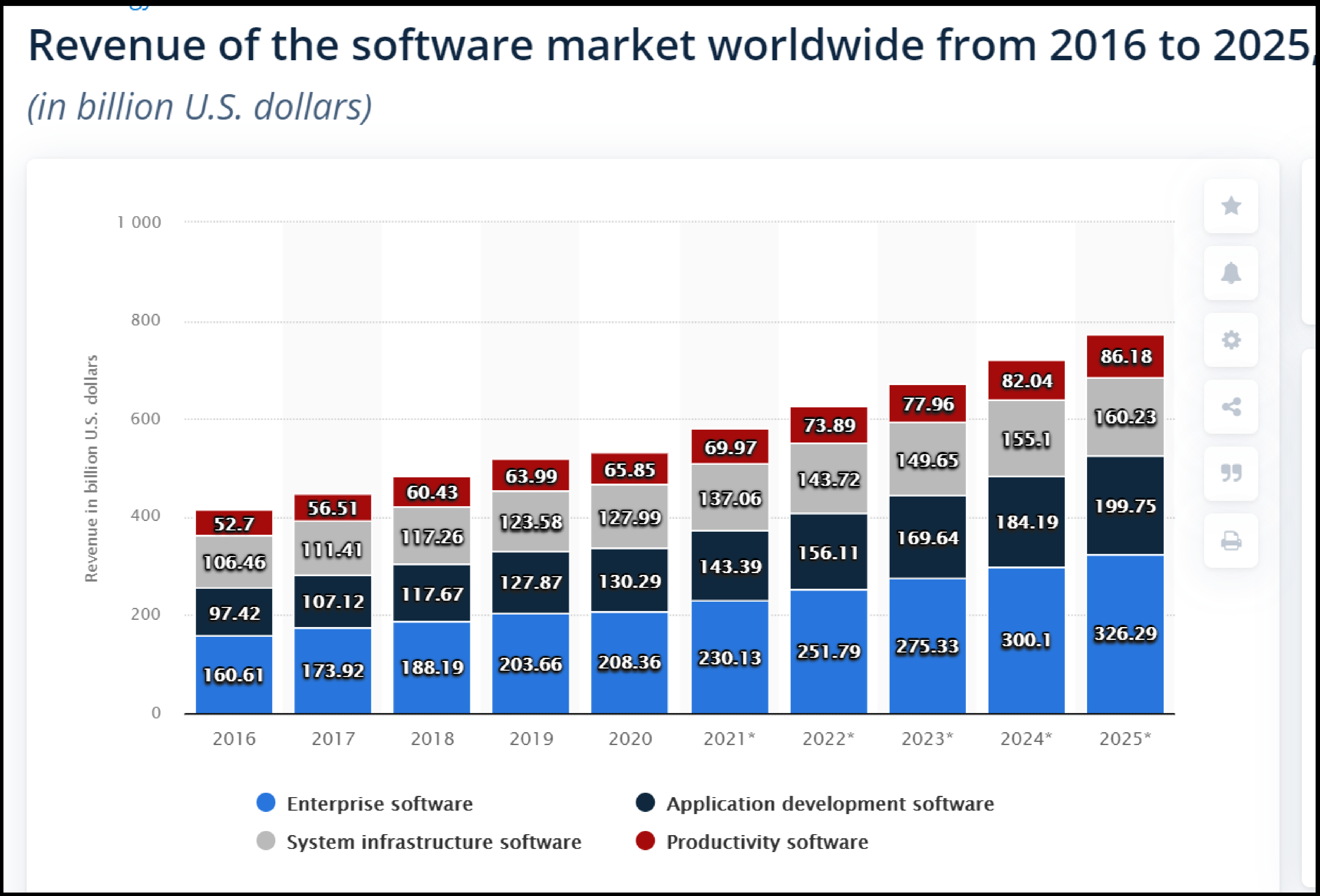
5 Laws of SE for AI

1. AI Software Mostly Isn't About AI
2. AI Software Needs Software Engineers
3. Poor SE Leads to Poor AI
4. Better SE Leads to Better AI
5. SE Needs Special Kinds of AI

SE4AI

All images/cliparts used in this presentation are from public domain and full credit to the original creators and authors!

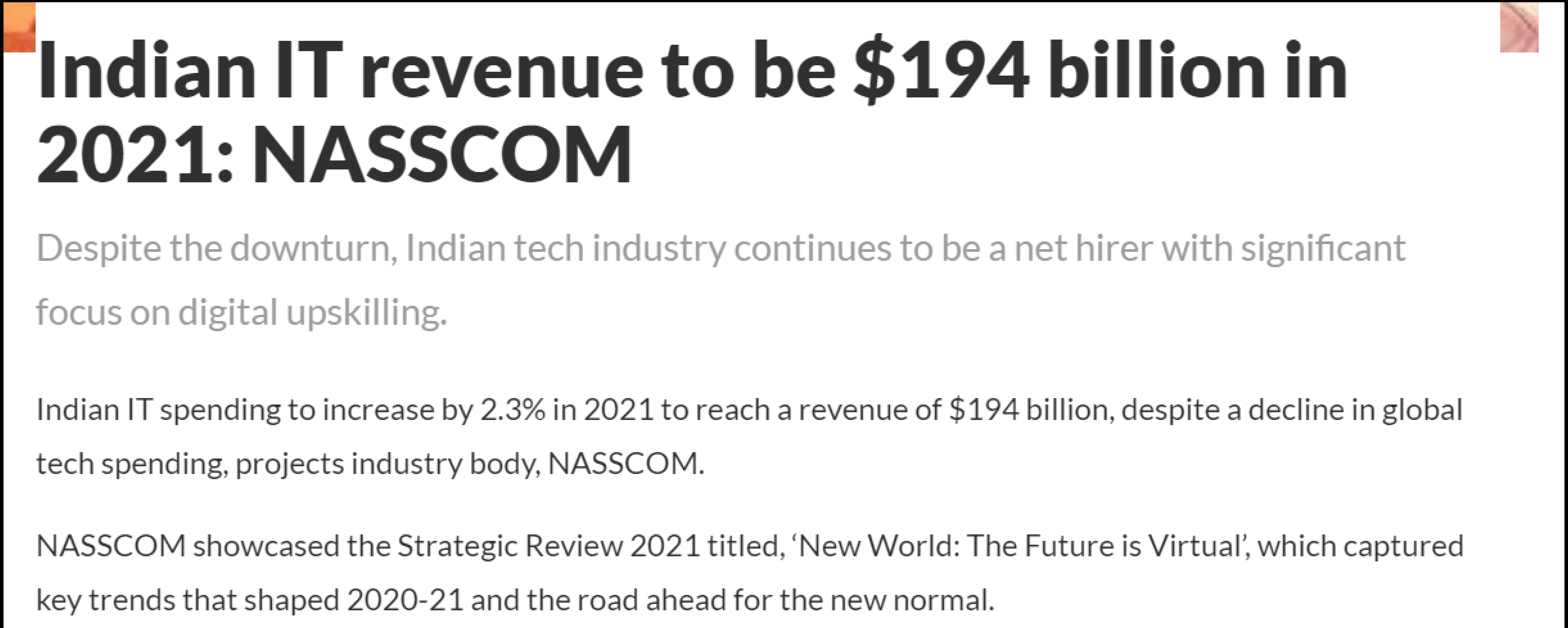
Software market?



Revenue of the software market worldwide from 2016 to 2025

(in billion U.S. dollars)

Digital India
Power To Empower



Indian IT revenue to be $194 billion in 2021: NASSCOM

Despite the downturn, Indian tech industry continues to be a net hirer with significant focus on digital upskilling.

Indian IT spending to increase by 2.3% in 2021 to reach a revenue of $194 billion, despite a decline in global tech spending, projects industry body, NASSCOM.

NASSCOM showcased the Strategic Review 2021 titled, ‘New World: The Future is Virtual’, which captured key trends that shaped 2020-21 and the road ahead for the new normal.

Can we rely on software?

AI in Today's World

Image Source: https://medium.com/dataseries/artificial-intelligence-a-i-where-are-we-today-a3638df630f2

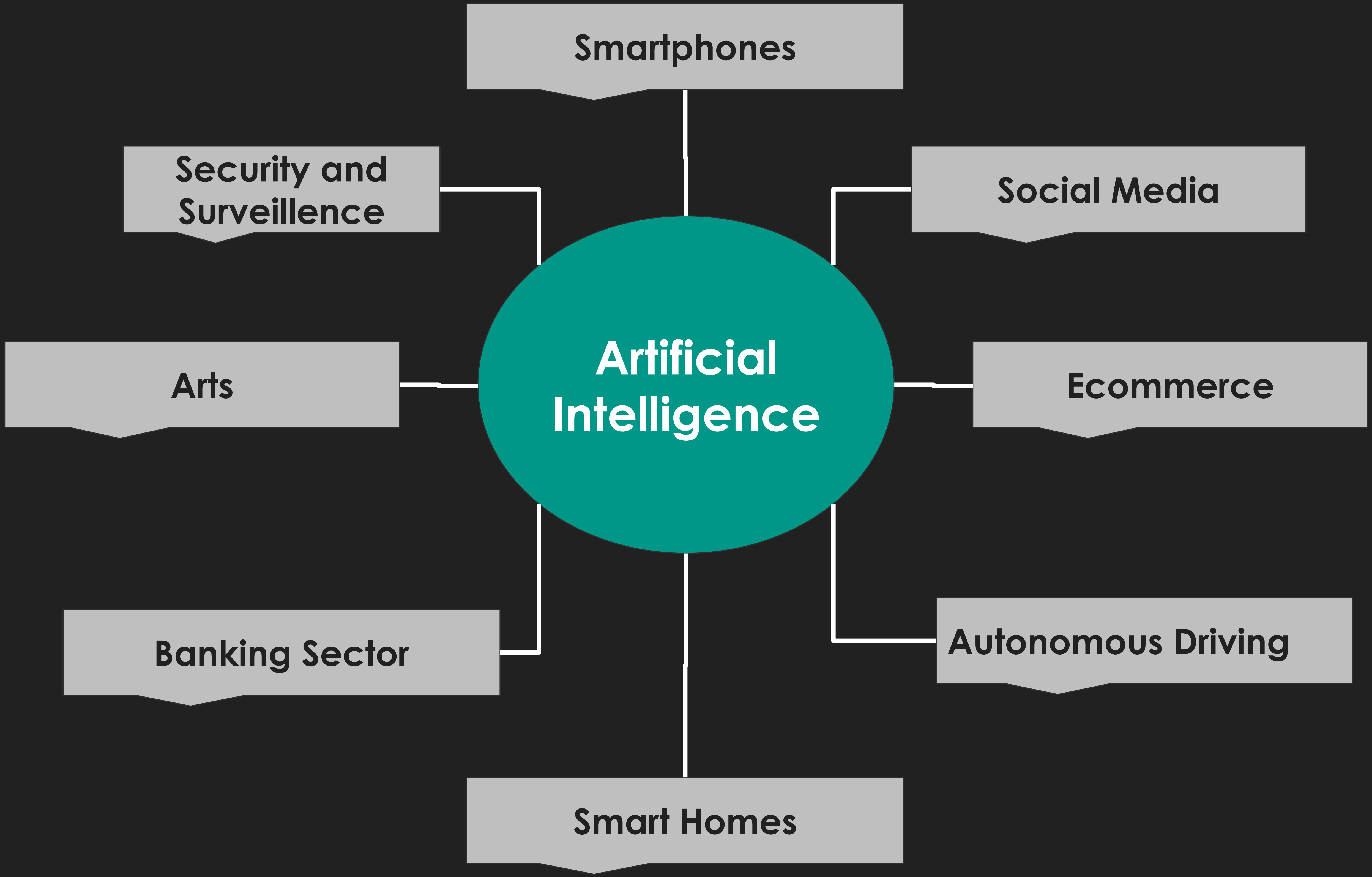
What is AI?

"It is the science and engineering of making intelligent machines, especially intelligent computer programs. It is related to the similar task of using computers to understand human intelligence, but AI does not have to confine itself to methods that are biologically observable."

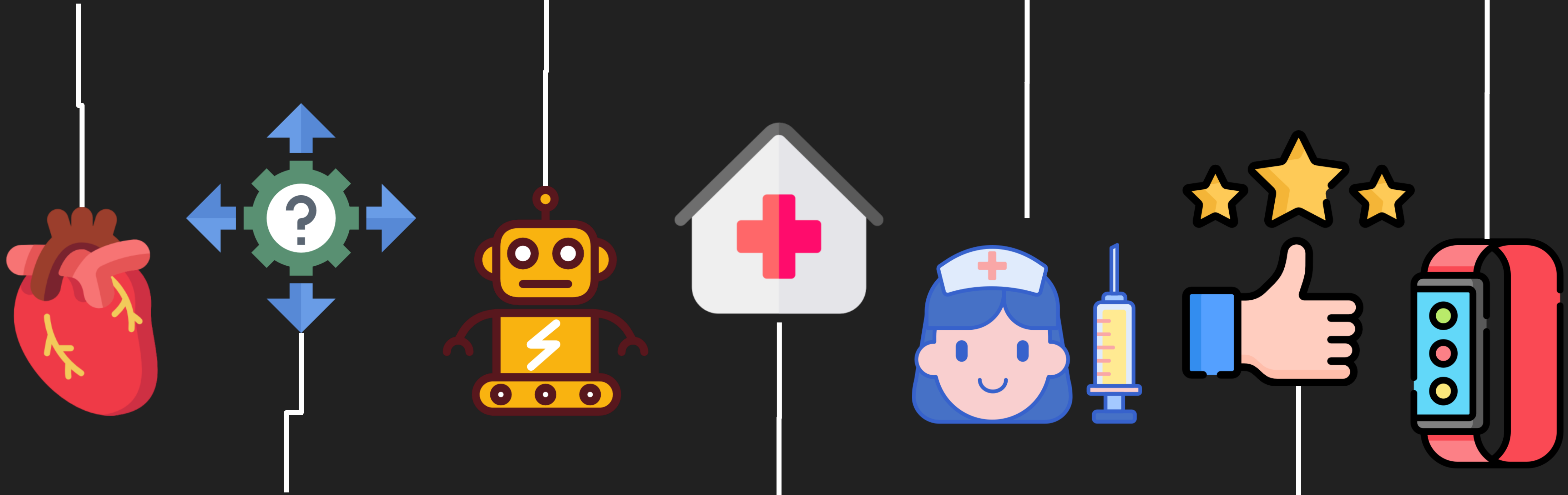
John McCarthy, 2004 [1976]

https://homes.di.unimi.it/borghese/Teaching/AdvancedIntelligentSystems/Old/IntelligentSystems_2008_2009/Old/IntelligentSystems_2005_2006/Documents/Symbolic/04_McCarthy_whatisai.pdf

The AI Landscape



AI in Healthcare



Early detection of ailment

Better Decision Making

Help in treatment

Extended Services

Associated Care

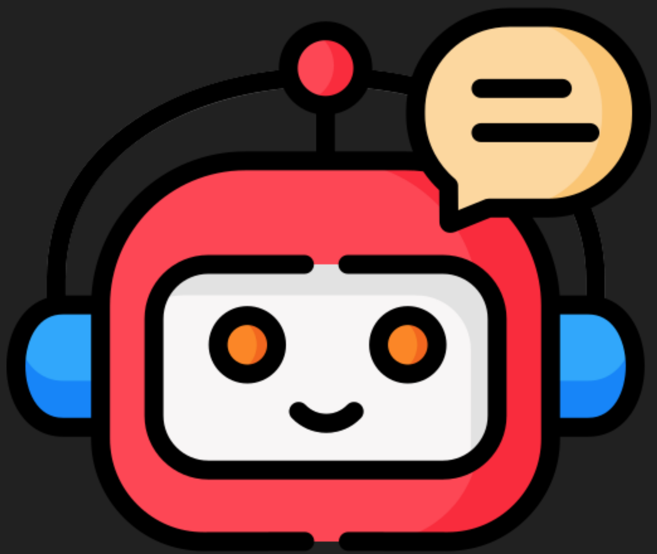
Superior Experience

Wearables

Source: Icons taken from https://flaticon.com

AI in Banking

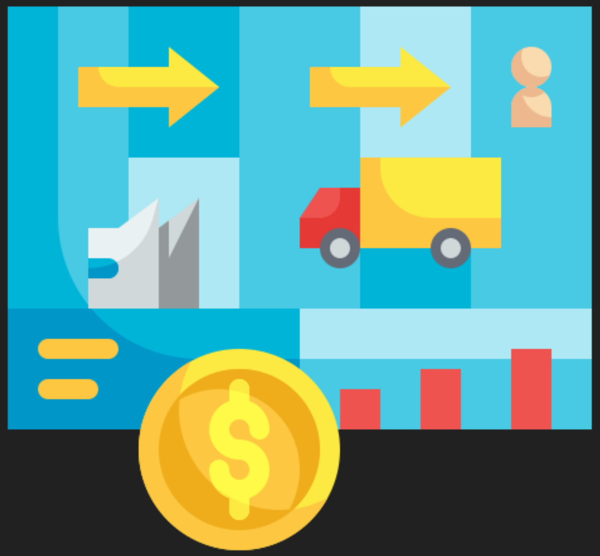
Chatbots



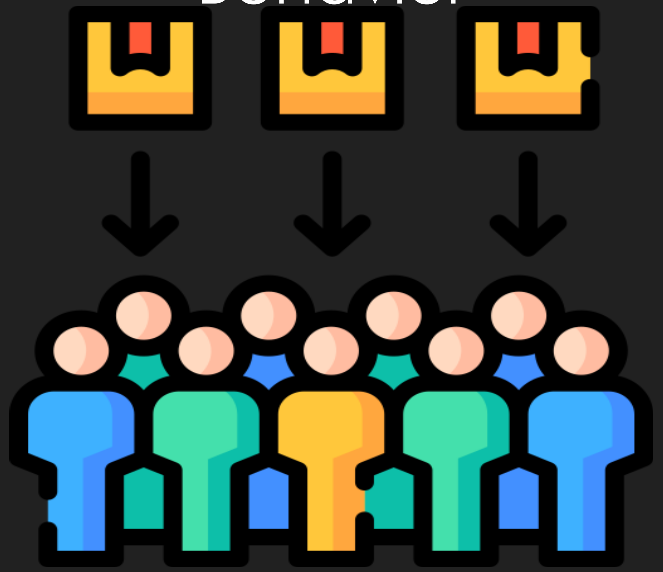
Analytics

Risk Compliance

Business Models

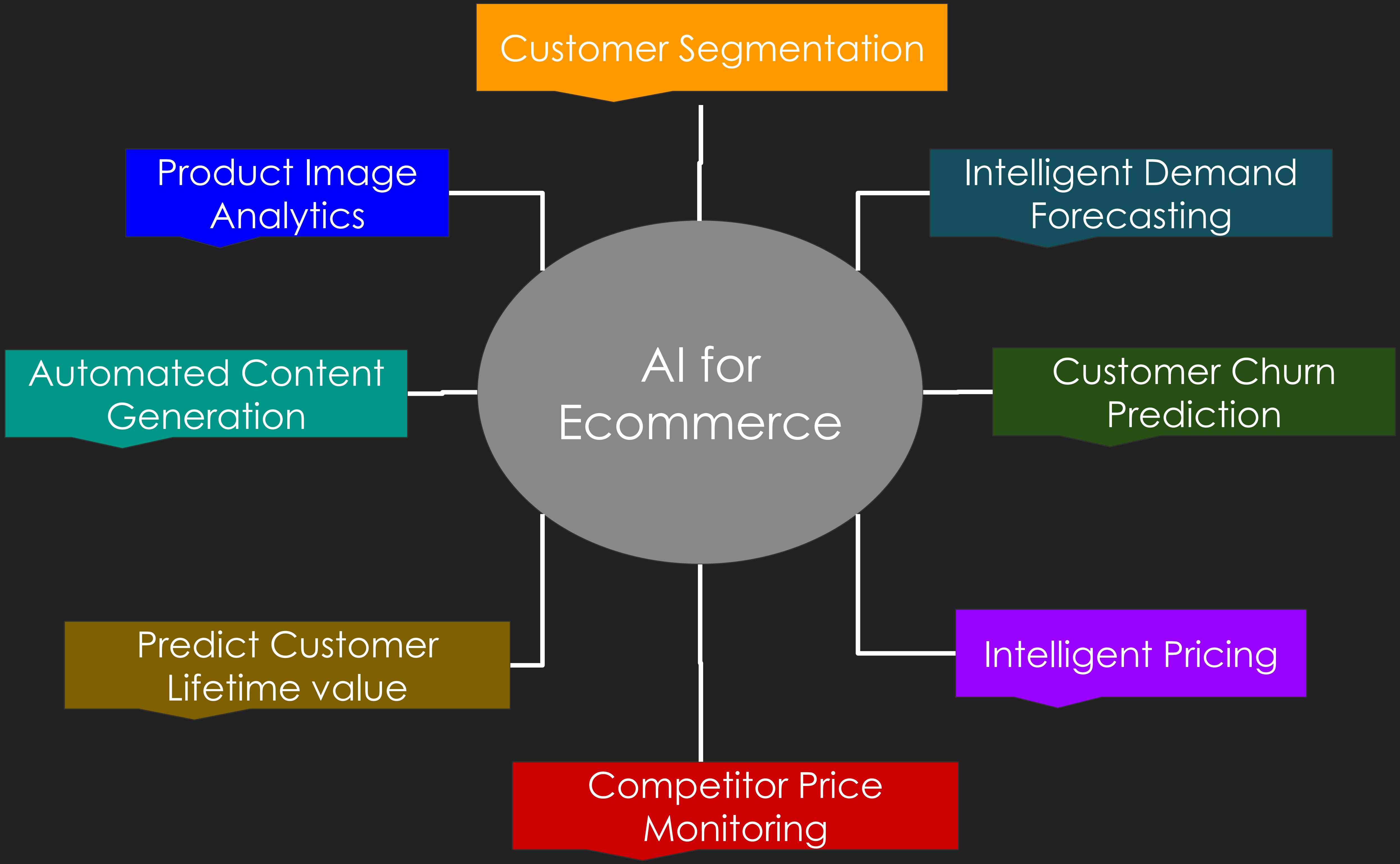


Customer Behavior



Source: Icons taken from https://flaticon.com

AI in E-Commerce


Customer Segmentation
Product Image Analytics
Intelligent Demand Forecasting
Automated Content Generation
AI for Ecommerce
Customer Churn Prediction
Predict Customer Lifetime value
Intelligent Pricing
Competitor Price Monitoring

AI Applications

Sophia: https://www.hansonrobotics.com/sophia/
Echo: https://www.theverge.com/2020/9/24/21452347/amazon-echo-4th-generation-features-price-release-date-alexa (Image Source)
Wyamo Self-Driving Cars: https://waymo.com/
NUMERAI: https://numer.ai/
AlphaGo Zero: https://deepmind.com/blog/article/alphago-zero-starting-scratch

How are companies leveraging AI?

Image Source: Logos of companies

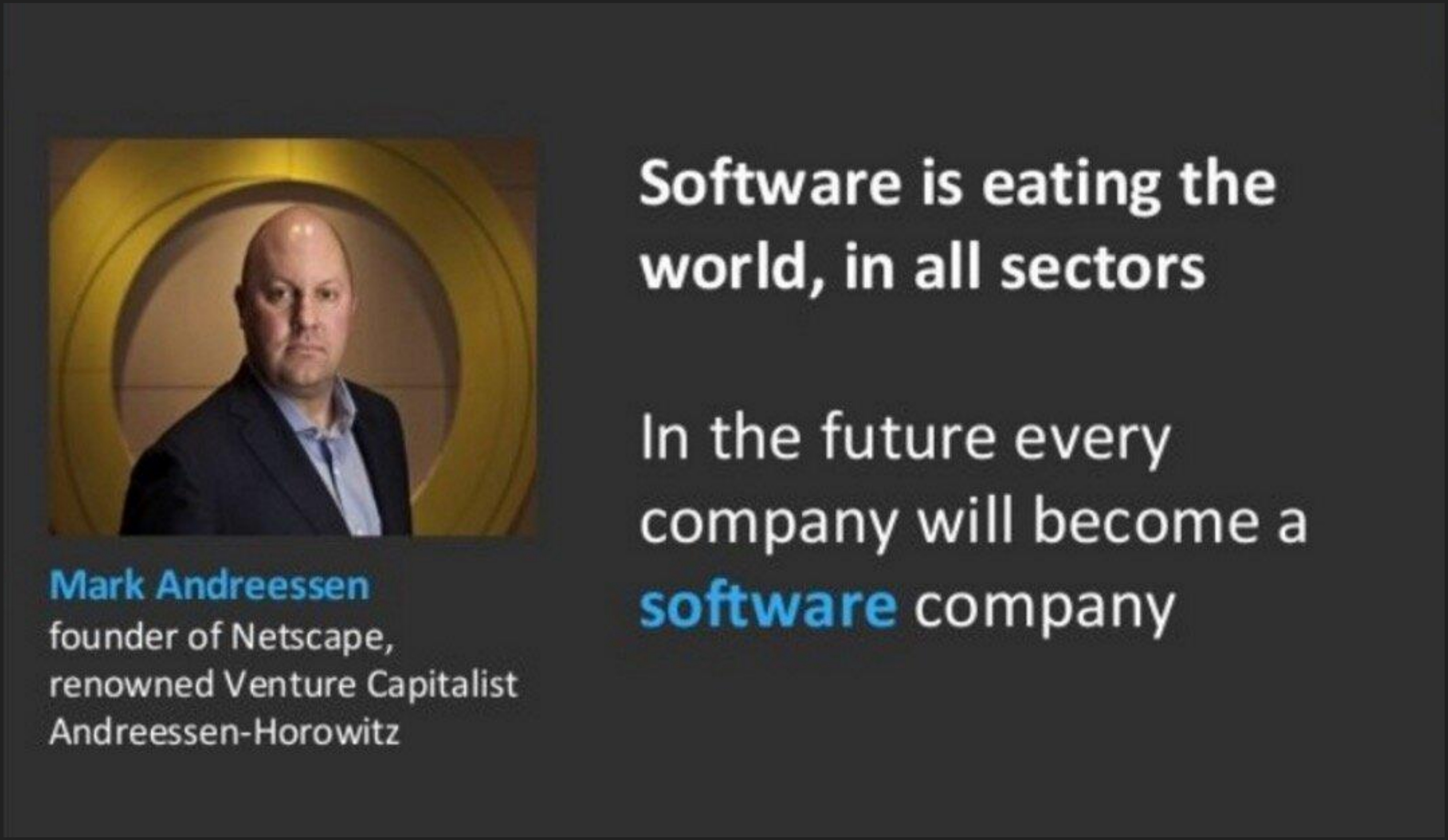
What is common across all these?

🤔

Software

Can we have AI without Software?

In 2011…

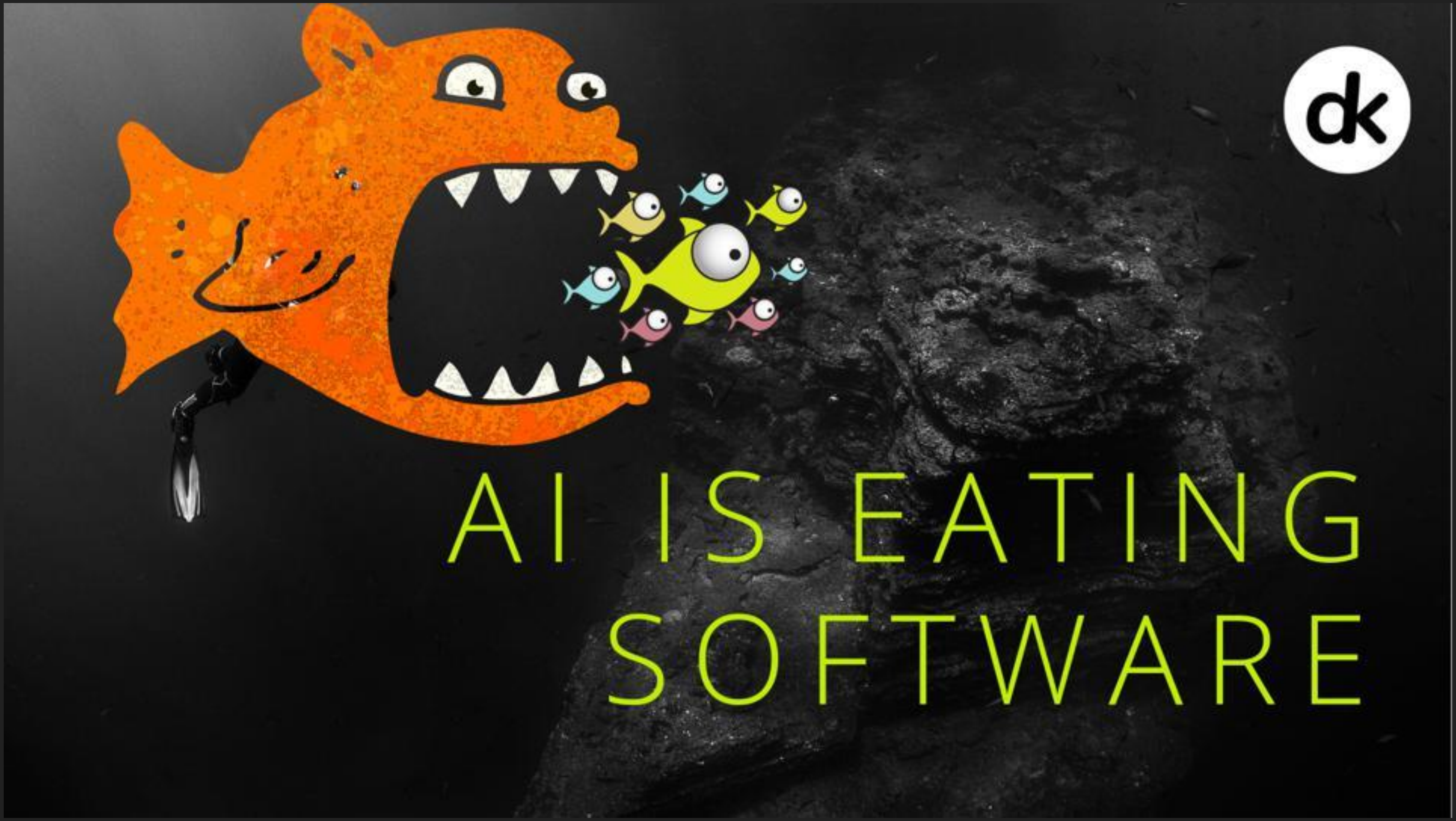


Mark Andreessen
founder of Netscape,
renowned Venture Capitalist
Andreessen-Horowitz

Software is eating the world, in all sectors

In the future every company will become a **software** company

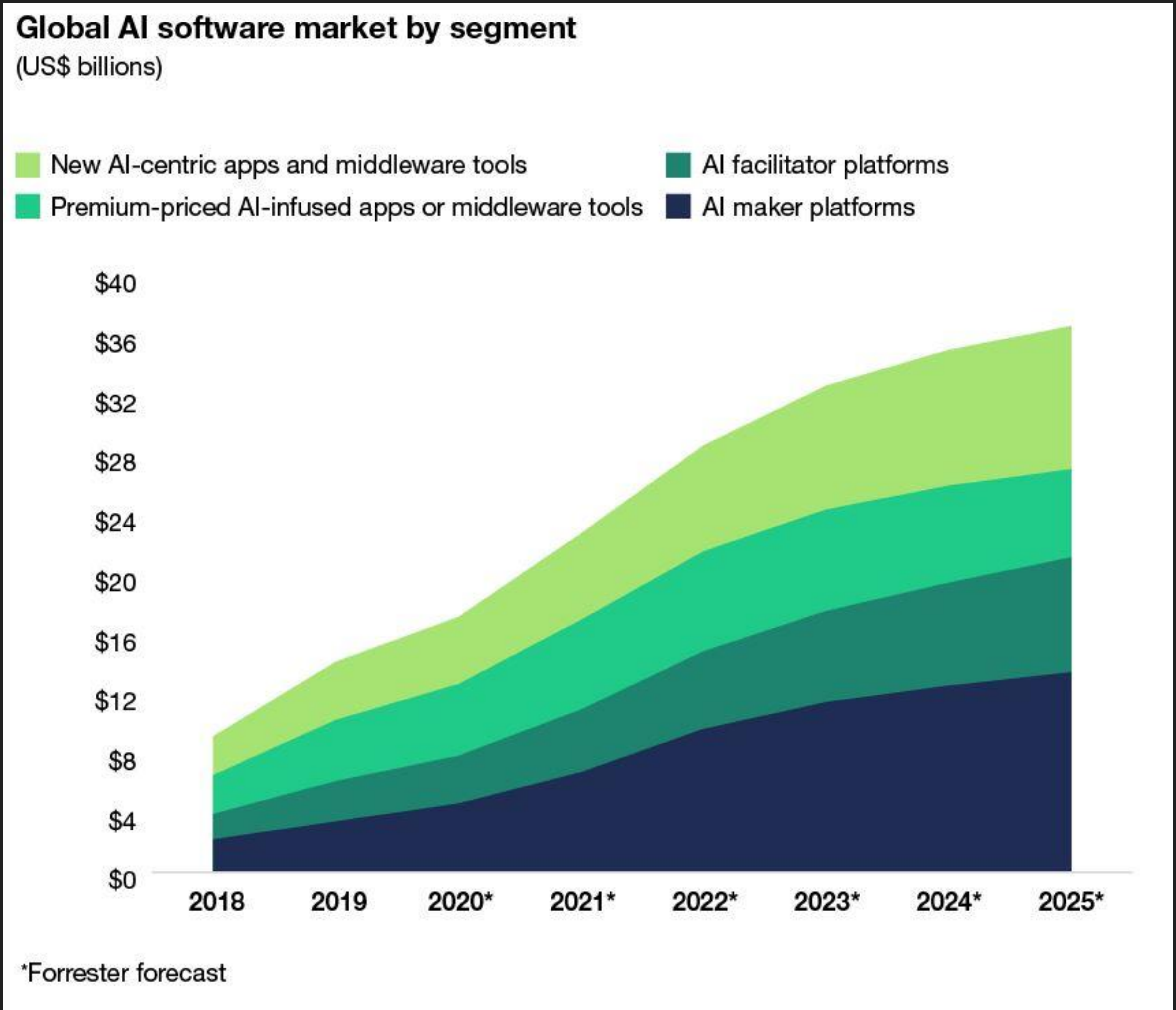
Now…



https://www.forbes.com/sites/cognitiveworld/2019/08/29/software-ate-the-world-now-ai-is-eating-software/?sh=42b8dde45810

Every company is an AI company today?

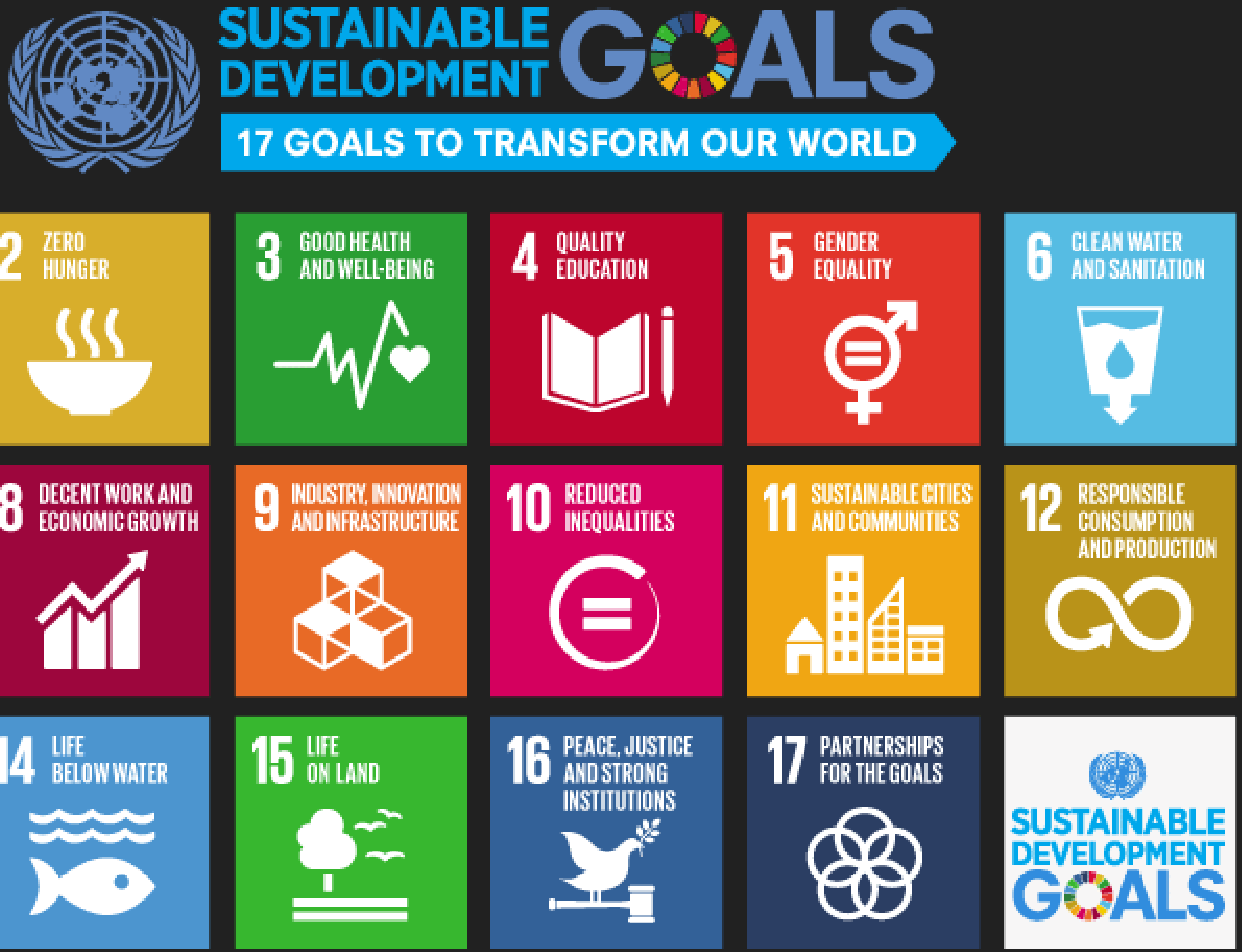
Growth of AI Software market



Source: https://go.forrester.com/blogs/sizing-the-ai-software-market-not-as-big-as-investors-expect-but-still-37-billion-by-2025/

AI for Social Good

AI for SDGs


SUSTAINABLE DEVELOPMENT GOALS
17 GOALS TO TRANSFORM OUR WORLD
1 NO POVERTY
2 ZERO HUNGER
3 GOOD HEALTH AND WELL-BEING
4 QUALITY EDUCATION
5 GENDER EQUALITY
6 CLEAN WATER AND SANITATION
7 AFFORDABLE AND CLEAN ENERGY
8 DECENT WORK AND ECONOMIC GROWTH
9 INDUSTRY, INNOVATION AND INFRASTRUCTURE
10 REDUCED INEQUALITIES
11 SUSTAINABLE CITIES AND COMMUNITIES
12 RESPONSIBLE CONSUMPTION AND PRODUCTION
13 CLIMATE ACTION
14 LIFE BELOW WATER
15 LIFE ON LAND
16 PEACE, JUSTICE AND STRONG INSTITUTIONS
17 PARTNERSHIPS FOR THE GOALS
SUSTAINABLE DEVELOPMENT GOALS

AI-based Vaccine Development

- Decades to months?

Source: https://www.technologyreview.com/2021/01/14/1016122/these-five-ai-developments-will-shape-2021-and-beyond/

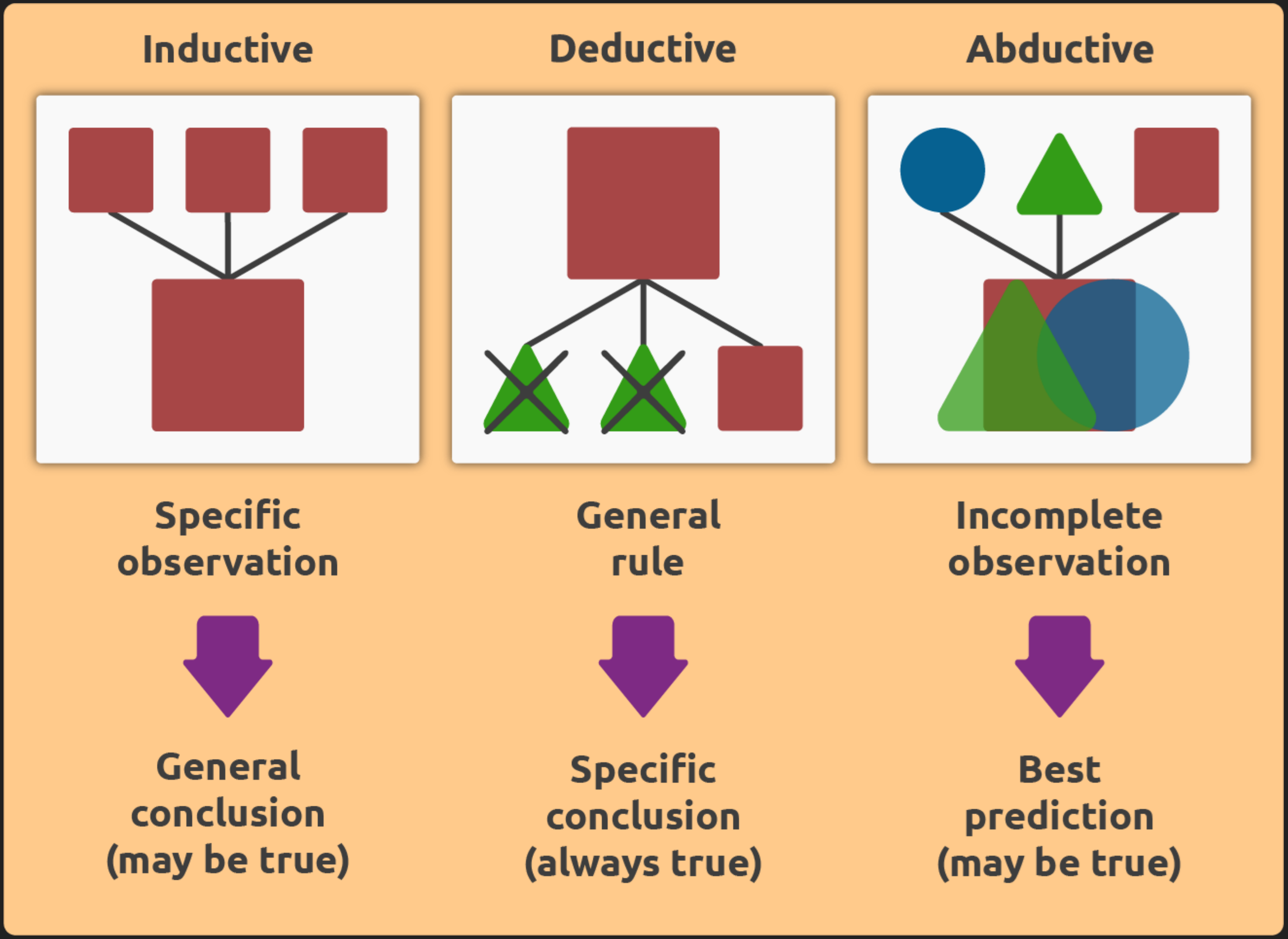
AI Fundamentals

Reasoning in AI

- Reason is the capacity of consciously applying logic to seek truth and draw conclusions from new or existing information.
- In artificial intelligence, the reasoning is essential so that the machine can also think rationally as a human brain, and can perform like a human.

Image Source: http://clipart-library.com/clipart/1801634.htm

Types of Reasoning [in AI]



Inductive

Specific observation

General conclusion (may be true)

Deductive

General rule

Specific conclusion (always true)

Abductive

Incomplete observation

Best prediction (may be true)

Source: https://www.braingymmer.com/en/blog/logical-reasoning/

AI4SE

Improve Software Quality

What and Why AI can support SE today?

- Millions of software repositories
- Source code, comments
- Readme files, docs
- Pull Requests, Commits, Issues
- Metadata (stars, contributors, time data…)
- Stack Overflow, Issue Trackers...

Image Credit: https://www.nvidia.com/en-gb/deep-learning-ai/solutions/

Why is it hard?

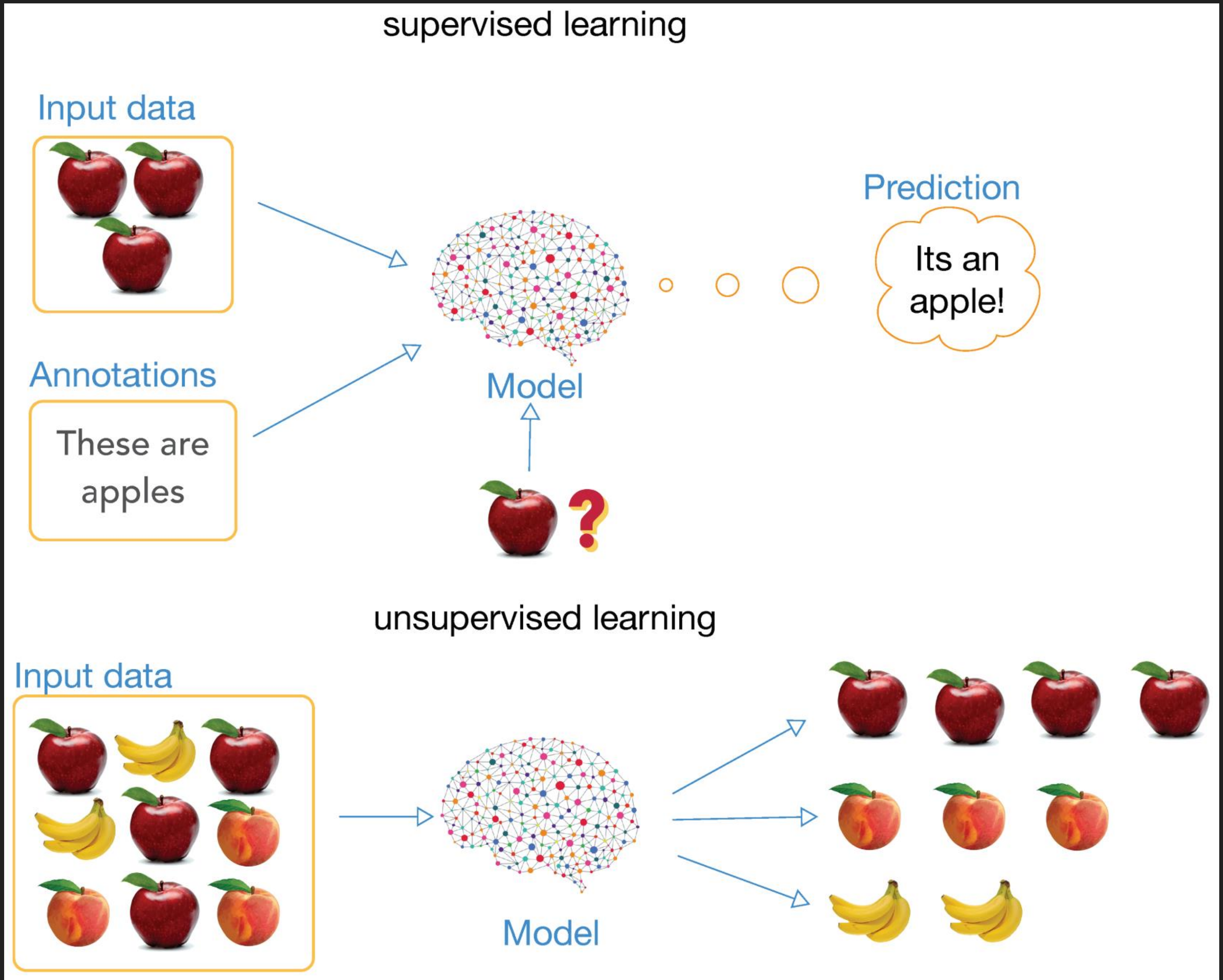
- I*ll*-formed Vs Well Formed problems
- No physical artifacts
- Lack of clarity - **Imprecise and Uncertainty is common!**
- Mind boggling complexity
- Failures often but not tolerable
- Change is expected rapidly

AI4SE

- Rule-Based, Search?
- Supervised Learning
- Unsupervised Learning
- Semi Supervised Learning
- Reinforcement Learning
- Deep Learning
- Transfer Learning
- Natural Language Processing
- Computer Vision
- Speech Assistants

- Code Comprehension
- Source Code Summarization
- Semantic Code Search
- Code Smells
- Code Assistants…
- …
- API Recommendations
- API Deprecation
- Legacy Code Migration
- Design Patterns
- Software Evolution

Supervised and Unsupervised Learning



- SL in SE: Estimation of effort based on structured requirement using a model trained with historical data.
- UL in SE: Detection of DDoS (Denial of Service) attack on web servers.

Ma, Y., Liu, K., Guan, Z., Xu, X., Qian, X., & Bao, H. (2018). Background augmentation generative adversarial networks (BAGANs): Effective data generation based on GAN-augmented 3D synthesizing. *Symmetry*, *10*(12), 734.

Natural Language Processing (NLP) & SE

- *Can we understand and extract requirements from product reviews?*
- *Can we generate design diagrams from source code?*
- *Can we generate test cases from requirements?*
- *Is software written for machines or humans?*

Examples of AI4SE

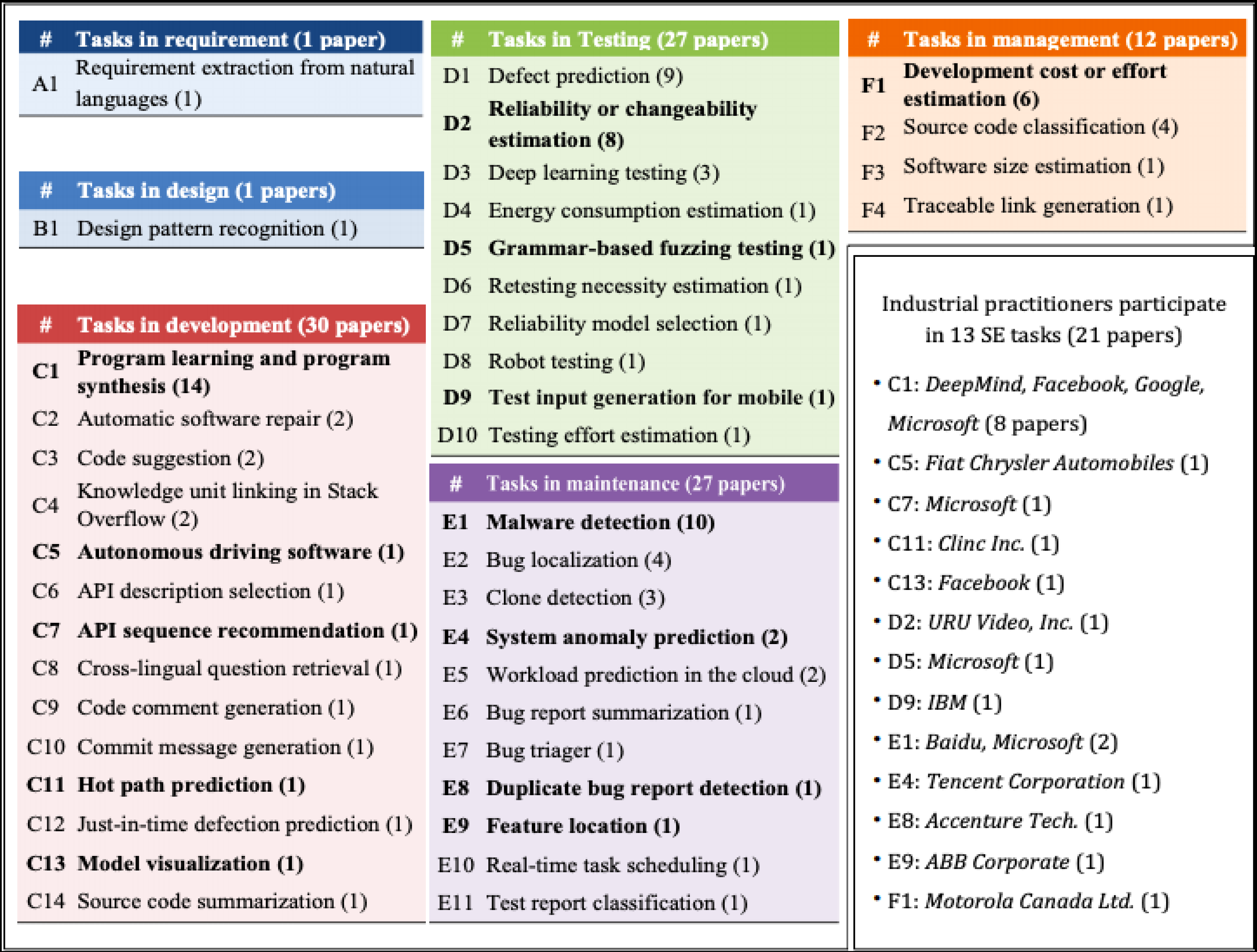
Where AI can help?

- Software process and product engineering; software development lifecycle models; agile software development; software deployment

- Requirements engineering; software architecture; software design; Unified Modeling Language (UML); design patterns;

- Software construction; testing; verification and validation; software metrics; software project management;

- Advances such as reuse, reengineering and evolution.



#	Tasks in requirement (1 paper)
A1	Requirement extraction from natural languages (1)

#	Tasks in design (1 papers)
B1	Design pattern recognition (1)

#	Tasks in development (30 papers)
C1	**Program learning and program synthesis (14)**
C2	Automatic software repair (2)
C3	Code suggestion (2)
C4	Knowledge unit linking in Stack Overflow (2)
C5	**Autonomous driving software (1)**
C6	API description selection (1)
C7	**API sequence recommendation (1)**
C8	Cross-lingual question retrieval (1)
C9	Code comment generation (1)
C10	Commit message generation (1)
C11	**Hot path prediction (1)**
C12	Just-in-time defection prediction (1)
C13	**Model visualization (1)**
C14	Source code summarization (1)

#	Tasks in Testing (27 papers)
D1	Defect prediction (9)
D2	**Reliability or changeability estimation (8)**
D3	Deep learning testing (3)
D4	Energy consumption estimation (1)
D5	**Grammar-based fuzzing testing (1)**
D6	Retesting necessity estimation (1)
D7	Reliability model selection (1)
D8	Robot testing (1)
D9	**Test input generation for mobile (1)**
D10	Testing effort estimation (1)

#	Tasks in maintenance (27 papers)
E1	**Malware detection (10)**
E2	Bug localization (4)
E3	Clone detection (3)
E4	**System anomaly prediction (2)**
E5	Workload prediction in the cloud (2)
E6	Bug report summarization (1)
E7	Bug triager (1)
E8	**Duplicate bug report detection (1)**
E9	**Feature location (1)**
E10	Real-time task scheduling (1)
E11	Test report classification (1)

#	Tasks in management (12 papers)
F1	**Development cost or effort estimation (6)**
F2	Source code classification (4)
F3	Software size estimation (1)
F4	Traceable link generation (1)

Industrial practitioners participate in 13 SE tasks (21 papers)

- C1: *DeepMind, Facebook, Google, Microsoft* (8 papers)
- C5: *Fiat Chrysler Automobiles* (1)
- C7: *Microsoft* (1)
- C11: *Clinc Inc.* (1)
- C13: *Facebook* (1)
- D2: *URU Video, Inc.* (1)
- D5: *Microsoft* (1)
- D9: *IBM* (1)
- E1: *Baidu, Microsoft* (2)
- E4: *Tencent Corporation* (1)
- E8: *Accenture Tech.* (1)
- E9: *ABB Corporate* (1)
- F1: *Motorola Canada Ltd.* (1)

Li, X., Jiang, H., Ren, Z., Li, G., & Zhang, J. (2018). Deep learning in software engineering. *arXiv preprint arXiv:1805.04825*.

Software Requirements

- Majority of Software requirements are [1]:
  - Written in natural language
  - Unstructured

- Identifying defects during the requirement analysis is crucial

- Understanding and modelling software requirements is a time consuming task [2]

1) Madala, K., Gaither, D., Nielsen, R., & Do, H. (2017, September). Automated identification of component state transition model elements from requirements. In *2017 IEEE 25th international requirements engineering conference workshops (REW)* (pp. 386-392). IEEE.
2) Madala, K., Piparia, S., Blanco, E., Do, H., & Bryce, R. (2021). Model elements identification using neural networks: a comprehensive study. *Requirements Engineering*, *26*, 67-96.

Tasks in RE
How AI can support?

- Requirements elicitation, prioritization, and negotiation; Capturing and understanding users' needs

- Traceability, evolution, reuse and management of requirements

- Requirements Engineering for Smart Cities, Cyber-Physical Systems, and Systems of Systems

- …

RE2021 - Research Papers - RE2021 (researchr.org)

Perini, A., Susi, A., & Avesani, P. (2012). A machine learning approach to software requirements prioritization. *IEEE Transactions on Software Engineering*, *39*(4), 445-461.

AI for Software Requirements

Types of Algorithms used in Requirement Phase:

1) NLP : For processing natural text requirements [3]
2) Neural networks : For generating automated requirement models [4]

How can we figure out whether requirements are complete or not? How AI can help in this process?

[3] Zeni, N., Kiyavitskaya, N., Mich, L., Cordy, J. R., & Mylopoulos, J. (2015). GaiusT: supporting the extraction of rights and obligations for regulatory compliance. *Requirements engineering*, *20*(1), 1-22.

[4] Madala, K., Piparia, S., Blanco, E., Do, H., & Bryce, R. (2021). Model elements identification using neural networks: a comprehensive study. *Requirements Engineering*, *26*, *67-96*.

Software Architecture

- A software system's architecture is the set of **<u>principal design decisions</u>** about the system during construction and evolution
  - Structure, Behavior, Interaction, Deployment
  - Qualities of software
  - Software archtiecture recovery, drift, erosion...

6. Jahić, J., & Roitsch, R. (2020, September). State of the practice survey: Predicting the influence of ai adoption on system software architecture in traditional embedded systems. In *European Conference on Software Architecture* (pp. 155-169). Springer, Cham.

Tasks in SA
How AI can support?

- Automatic extraction and generation of software architecture descriptions; Refactoring and evolving architecture design decisions and solutions;
- Software architecture for legacy systems and systems integration
- DevOps, Containerization; Microservices and event-driven architectures
- Architecting specific types of systems, such as Systems of Systems, IoT systems, AI/ML systems, CPSs, software ecosystems, self-adaptive systems, or autonomous systems

https://icsa-conferences.org/2021/call-for-papers/technical-papers/

AI in Software Design

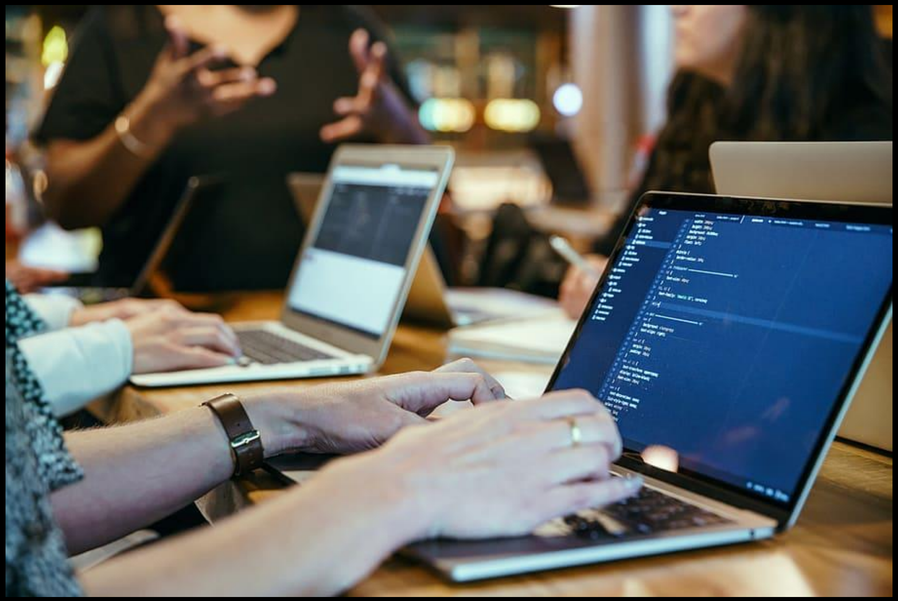
- Automatic verification of design of software systems, simplifies the process of software development [5]
- Identifying design model present from source code is of direct interest to developers for software maintenance
- Algorithms such as Decision Trees and neural networks are also used to identify the faults in the design

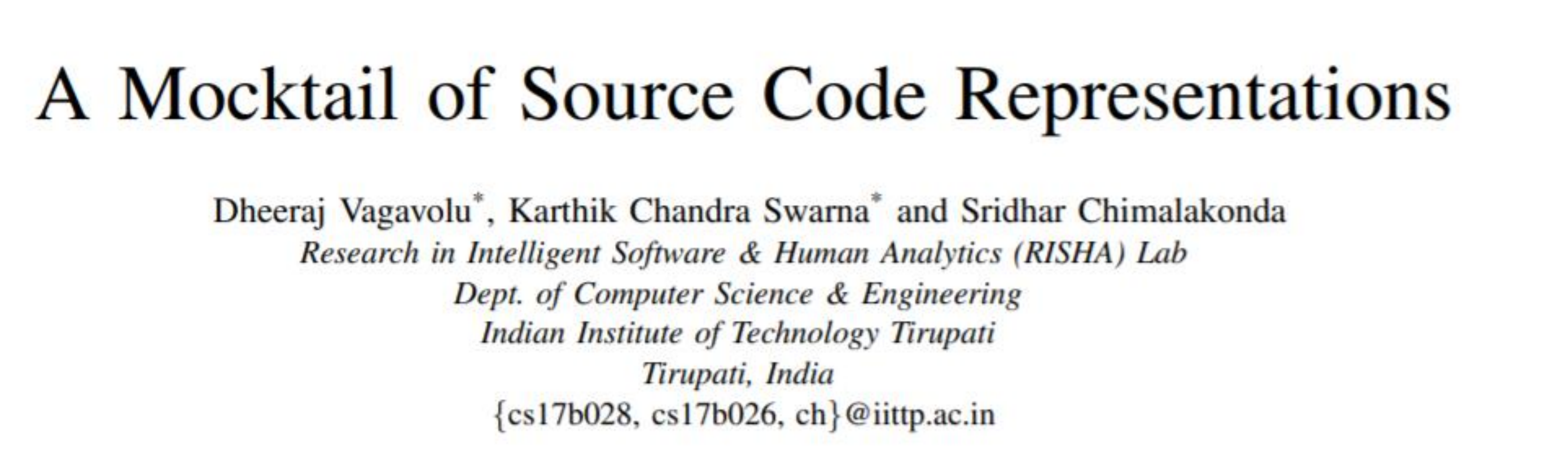
Cai, C. H., Sun, J., & Dobbie, G. (2019). Automatic B-model repair using model checking and machine learning. *Automated Software Engineering*, *26*(3), 653-704.

Tasks in Software Construction

- Code Completion
- API Recommendations
- Code Comment Suggestions
- Source Code Summarization
- Semantic Code Search
- Software Documentation

- Source code representations for AI: *code2vec*, *code2seq*, *code2graph*, *ASTNN*, *codeBERT*, Mocktail…





A Mocktail of Source Code Representations

Dheeraj Vagavolu*, Karthik Chandra Swarna* and Sridhar Chimalakonda
Research in Intelligent Software & Human Analytics (RISHA) Lab
Dept. of Computer Science & Engineering
Indian Institute of Technology Tirupati
Tirupati, India
{cs17b028, cs17b026, ch}@iittp.ac.in

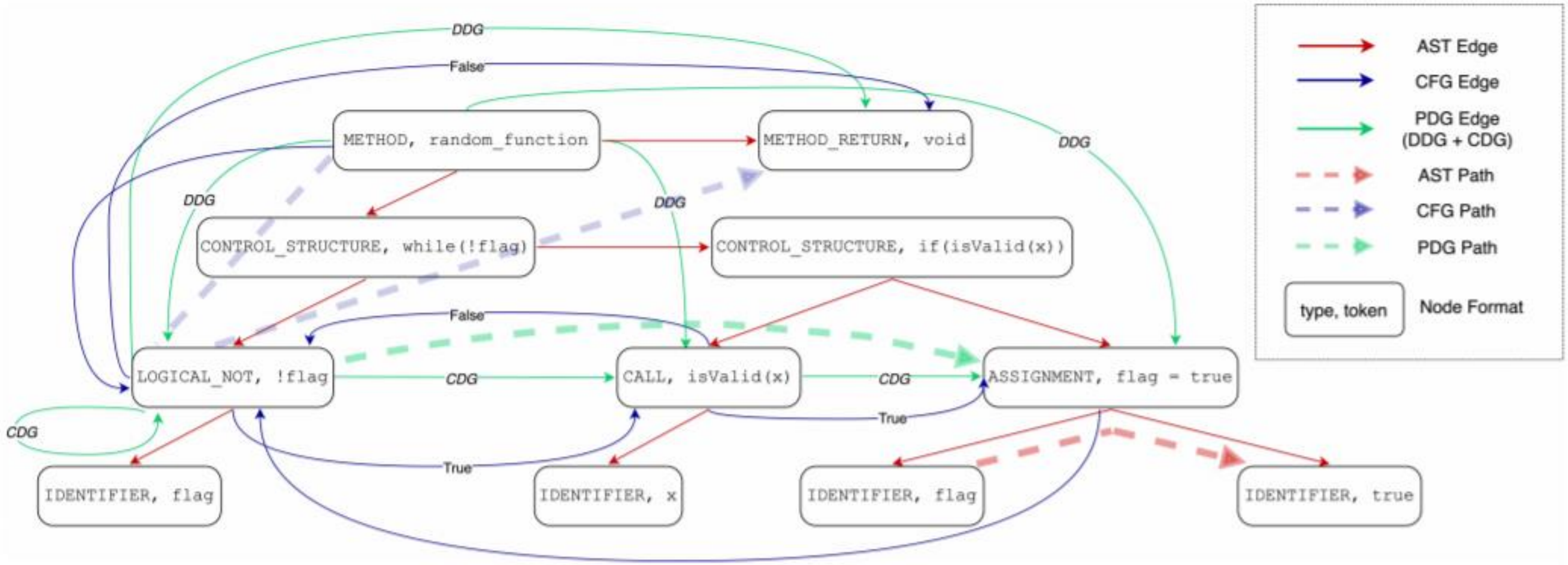


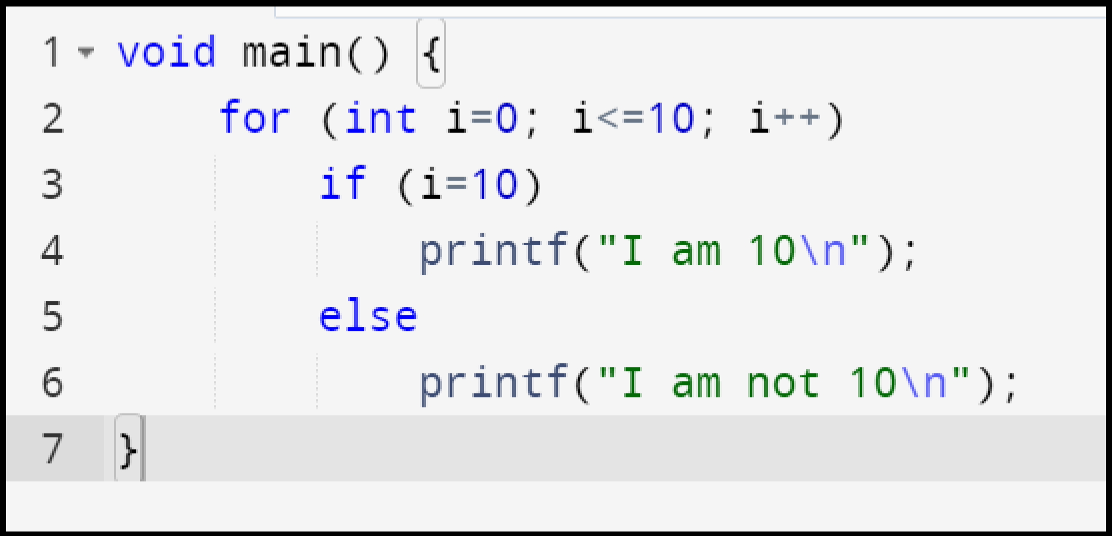
Fig. 2. Extracting AST, CFG, and PDG paths from the code snippet in Fig. 1

Code is often buggy!
Debugging is effort intensive!

- Locate and fix errors!
- Fundamental skill
- Requires practice
- Improves skill of problem handling

A program to print "I am 10" if *i* is 10, else "I am not 10"

What does it print?



```
void main() {
    for (int i=0; i<=10; i++)
        if (i=10)
            printf("I am 10\n");
        else
            printf("I am not 10\n");
}
```

https://www.edureka.co/blog/what-is-debugging/

Code Completion

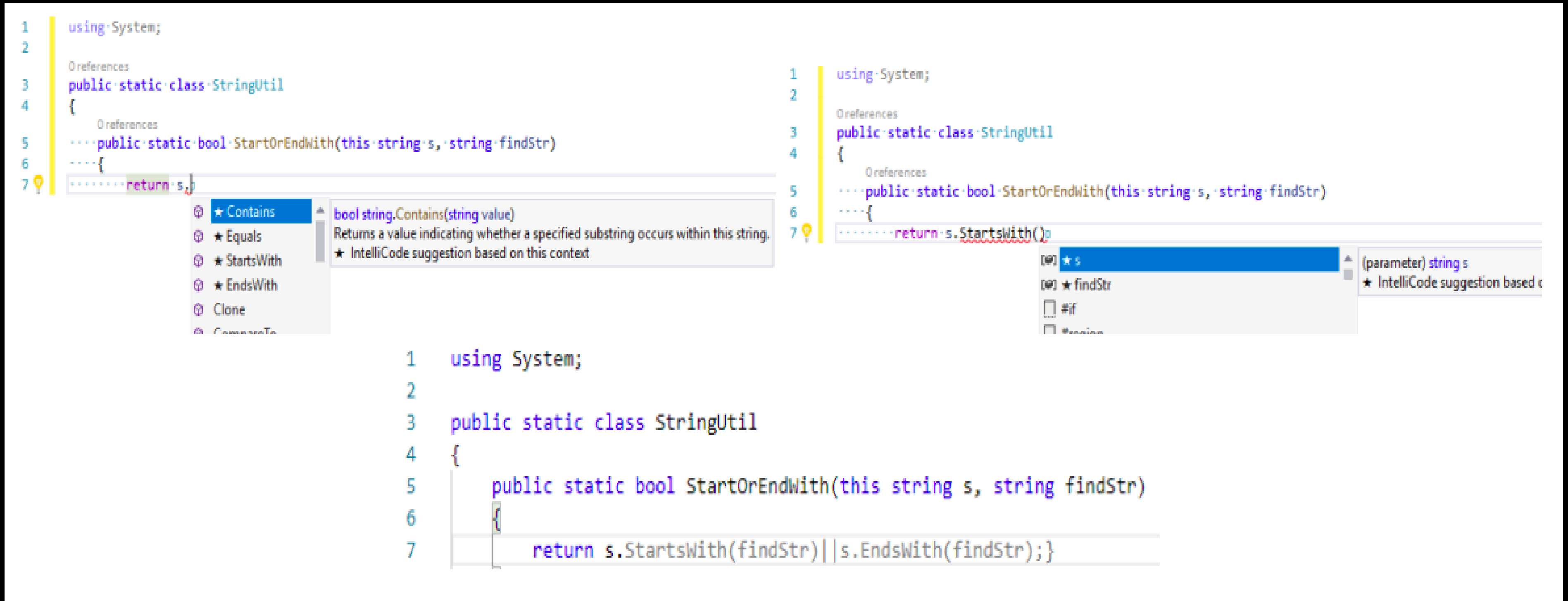
A set of suggestions provided by the IDEs for software developers, maintainers [10]:

- Increase the speed of coding
- Promote reuse
- Improve quality of the code

10. Allamanis, M., Barr, E. T., Devanbu, P., & Sutton, C. (2018). A survey of machine learning for big code and naturalness. *ACM Computing Surveys (CSUR)*, *51*(4), 1-37.

IntelliCode Compose: Code Generation using Transformer

(ESEC/FSE 2020)



```
using·System;

0 references
public·static·class·StringUtil
{
    0 references
····public·static·bool·StartOrEndWith(this·string·s,·string·findStr)
····{
········return·s.
```

bool string.Contains(string value)
Returns a value indicating whether a specified substring occurs within this string.
★ IntelliCode suggestion based on this context

```
using·System;

0 references
public·static·class·StringUtil
{
    0 references
····public·static·bool·StartOrEndWith(this·string·s,·string·findStr)
····{
········return·s.StartsWith()
```

(parameter) string s
★ IntelliCode suggestion based c

```
using System;

public static class StringUtil
{
    public static bool StartOrEndWith(this string s, string findStr)
    {
        return s.StartsWith(findStr)||s.EndsWith(findStr);}
```

Figure 1: Comparison of code completion scenarios. Top: method completion and argument completion served by *Intellicode.* Bottom: whole-line of code completion served by the *IntelliCode Compose.*

- General purpose code completion framework.
- Generate syntactically correct code for multiple programming languages [11].
- Uses GPT-C(generative transformer model for code) for suggesting a whole line of code.
- Developed by Microsoft.
- Supports C#, JavaScript, Python and TypeScript.

[11] Svyatkovskiy, A., Deng, S. K., Fu, S., & Sundaresan, N. (2020, November). Intellicode compose: Code generation using transformer. In *Proceedings of the 28th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering* (pp. 1433-1443).

kite

- AI powered code completion tool.
- Supports 16 programming languages and 16 editors.
- Its ML models is trained on 25 millions files.

tabnine

Code faster with AI code completions

- AI powered code completion tool, with GPT-2 model at its core.
- Supports 30 programing languages and 15 editors.
- Provided an option of private code model.
- Heavier application compare to kite

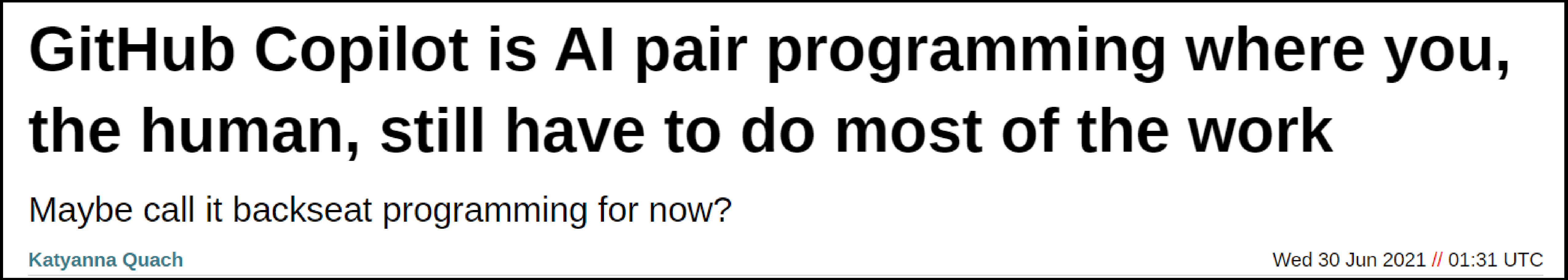
1. https://www.kite.com/
2. https://www.tabnine.com/
3. https://medium.com/swlh/kite-vs-tabnine-which-ai-code-autocomplete-should-you-choose-eb6eba85c3a6

Github Copilot - An AI Pair Programmer

- “GitHub Copilot draws context from the code you’re working on, suggesting whole lines or entire functions” - GitHub CEO Nat Friedman

- Can we support developers without having to navigate and search Stack Overflow, Google or the web to find relevant code?

- https://twitter.com/i/status/1411074516411764743



GitHub Copilot is AI pair programming where you, the human, still have to do most of the work

Maybe call it backseat programming for now?

Katyanna Quach

Wed 30 Jun 2021 // 01:31 UTC

- “The model generated correct code 43 per cent of the time on the first try, the PR rep said, and 57 per cent of the time when allowed 10 attempts.”

https://www.theregister.com/2021/06/30/github_ai_copilot/

API Recommendations

- The use of API in software development has been drastically increased.
- APIs simplify the process of development and maintenance of a software application.
- Real-time recommendations of APIs while writing a code is still a challenging task.

PyART: Python API Recommendation in Real-Time – A lightweight tool

(ICSE 2021)

- PyART (**Py**thon **A**PI **R**ecommendation in Real-**T**ime)[12]
- Recommend library APIs and APIs defined in the same project
- The tool considered three features, i.e., optimistic data-flow, token similarity along data-flow and token co-occurrence.
- These features are trained to create a predictive model based on **Random Forest**.

12. He, X., Xu, L., Zhang, X., Hao, R., Feng, Y., & Xu, B. (2021, May). PyART: Python API Recommendation in Real-Time. In *2021 IEEE/ACM 43rd International Conference on Software Engineering (ICSE)* (pp. 1634-1645). IEEE.

Recommending API Function Calls and Code Snippets to Support Software Development

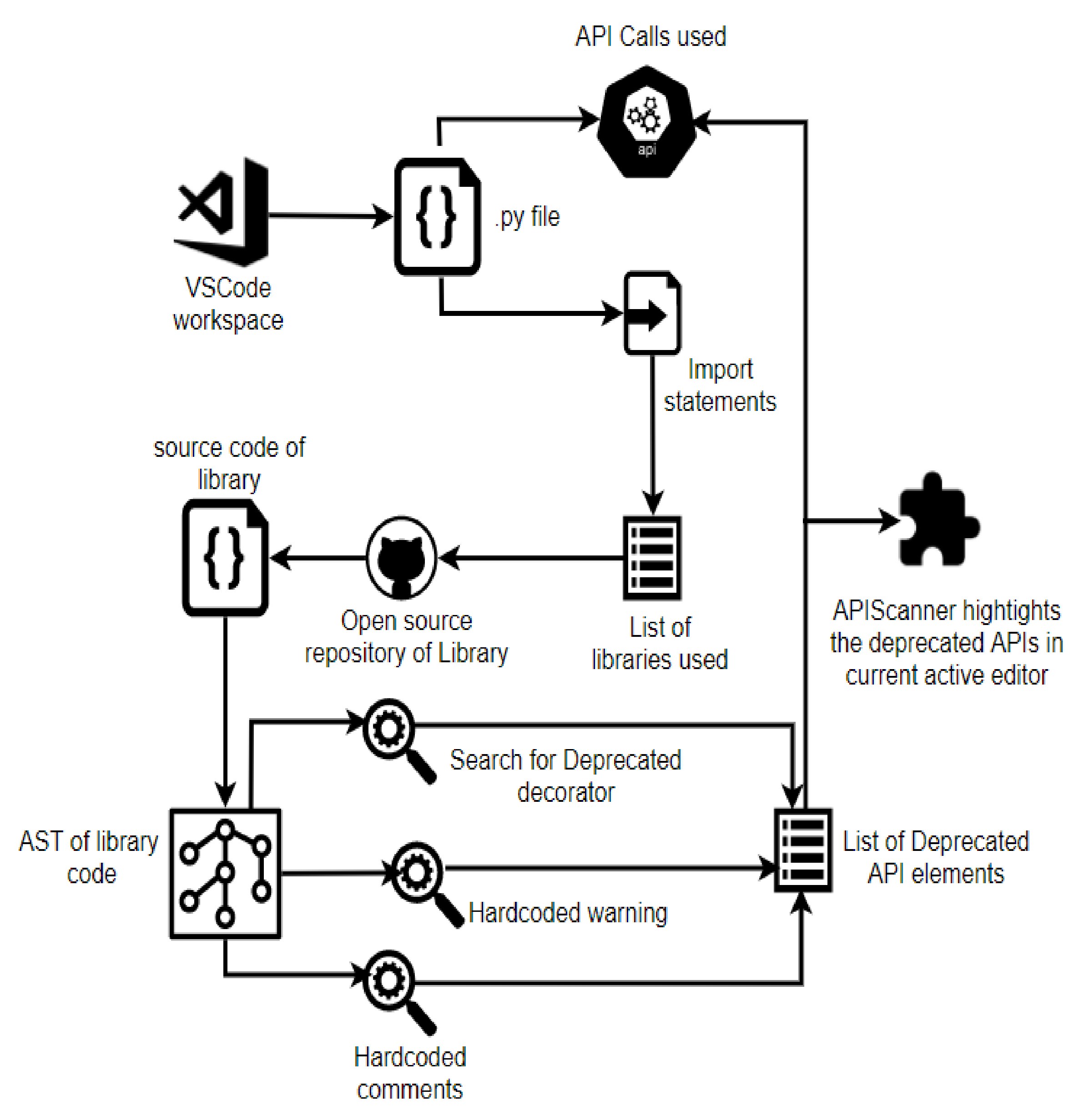
(IEEE TSE 2021)

- **FOCUS**: A tool which recommends API calls and source code during software development [13]
- It based on collaborative filtering recommender system
- Extract the API usages from open source software
- Tested on Android programming of 2600 mobile apps

13. Nguyen, P. T., Di Rocco, J., Di Sipio, C., Di Ruscio, D., & Di Penta, M. (2021). Recommending api function calls and code snippets to support software development. *IEEE Transactions on Software Engineering*.

APIScanner - Towards Automated Detection of Deprecated APIs in Python Libraries

(ICSE Demo 2021)


API Calls used
api
.py file
VSCode workspace
Import statements
source code of library
Open source repository of Library
List of libraries used
APIScanner hightights the deprecated APIs in current active editor
Search for Deprecated decorator
AST of library code
List of Deprecated API elements
Hardcoded warning
Hardcoded comments

- Automatically detects deprecated APIs in a dynamic language, i.e., Python[14].
- Most of existing work focuses on static languages such as Java and C#.
- A visual code extension.
- Uses ASTs of source code to identify the deprecated modules of the libraries.

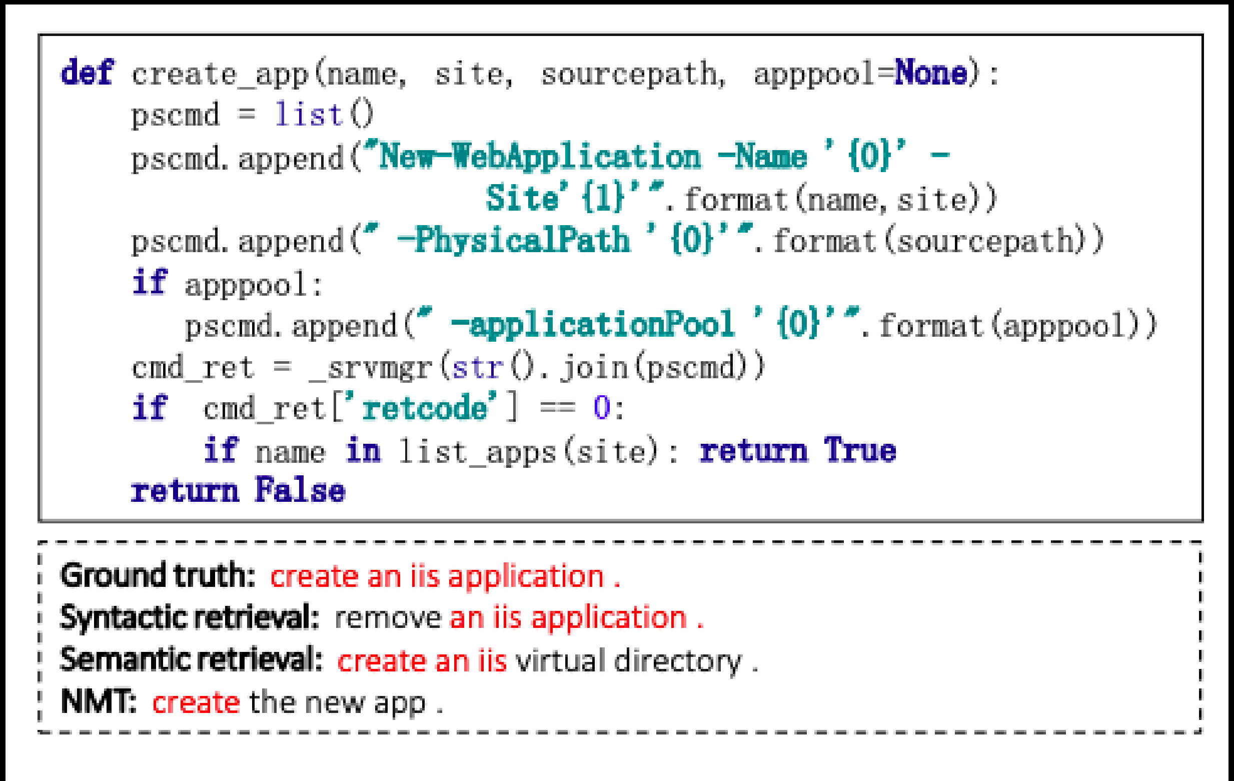
14. Vadlamani, A., Kalicheti, R., & Chimalakonda, S. (2021, May). APIScanner-Towards Automated Detection of Deprecated APIs in Python Libraries. In *2021 IEEE/ACM 43rd International Conference on Software Engineering: Companion Proceedings (ICSE-Companion)* (pp. 5-8). IEEE.

Code Summarization

- Automatically generating the natural language descriptions of a source code
- Aimed for improving the comprehension of source code for developers by providing concise summaries
- Legacy software systems without documentation

Retrieval-based
Neural Source Code Summarization

(ICSE 2020)



```
def create_app(name, site, sourcepath, apppool=None):
    pscmd = list()
    pscmd.append("New-WebApplication -Name '{0}' -
                    Site' {1}'".format(name, site))
    pscmd.append(" -PhysicalPath '{0}'".format(sourcepath))
    if apppool:
       pscmd.append(" -applicationPool '{0}'".format(apppool))
    cmd_ret = _srvmgr(str().join(pscmd))
    if  cmd_ret['retcode'] == 0:
        if name in list_apps(site): return True
    return False
```

Ground truth: create an iis application .

Syntactic retrieval: remove an iis application .

Semantic retrieval: create an iis virtual directory .

NMT: create the new app .

Figure 1: An example of NMT-based and IR-based source code summarization, where the correct words are marked in red

- Based on neural architecture called **Rencos** (**Re**trieval-based **N**eural Source **Co**de **S**ummarizer)[15].
- It is developed using Neural Machine Translation (NMT) and Information Retrieval Based Methods.
- Given an input code snippet, the model tries to obtain the similar code snippet from the training set.
- Two most similar code snippets are based on the syntax-level and semantics-level information based on Rencos model.

[15] Zhang, J., Wang, X., Zhang, H., Sun, H., & Liu, X. (2020, October). Retrieval-based neural source code summarization. In *2020 IEEE/ACM 42nd International Conference on Software Engineering (ICSE)* (pp. 1385-1397). IEEE.

Improved Code Summarization via a Graph Neural Network

(ICPC 2020)

- Summarizes the program subroutines.
- Used **graph2seq** encoder
- First the subroutines are converted into AST followed by using GNN (graph2seq) + RNN based encoder to generate a sequence of source code and AST tokens
- Uses attention mechanism to learn important information from source code followed by predicting the next token in the sequence

16. LeClair, A., Haque, S., Wu, L., & McMillan, C. (2020, July). Improved code summarization via a graph neural network. In *Proceedings of the 28th International Conference on Program Comprehension* (pp. 184-195).

How many software projects have proper documentation?

Can AI help in generating software documentation and keeping it up-to-date?

Tasks in Software Testing

- Defect prediction
- Mutation testing
- Test case generation

Defect Prediction

- The task of identifying the defect prone modules in the software system is defect prediction [17].

- Two types of defect prediction:
  - Within-Project Defect Prediction
  - Cross-Project Defect Prediction

- Identifying the defect prone modules will reduce the time of testing of the software systems.

[17] Zhou, Y., Yang, Y., Lu, H., Chen, L., Li, Y., Zhao, Y., ... & Xu, B. (2018). How far we have progressed in the journey? an examination of cross-project defect prediction. *ACM Transactions on Software Engineering and Methodology (TOSEM)*, *27*(1), 1-51.

DeepJIT: An End-To-End Deep Learning Framework for Just-In-Time Defect Prediction (MSR 2019)

- JIT (Just in time) : The term is used to define defect prediction techniques, which try to identifiy defects in the early stages of coding / whenever a new change is made to the codebase.

- DeepJIT is a JIT defect prediction techniques based on Convolution Neural Networks (CNNs)[18].

- DeepJIT uses code commit messages and code changes present in the commit to identify the defects.

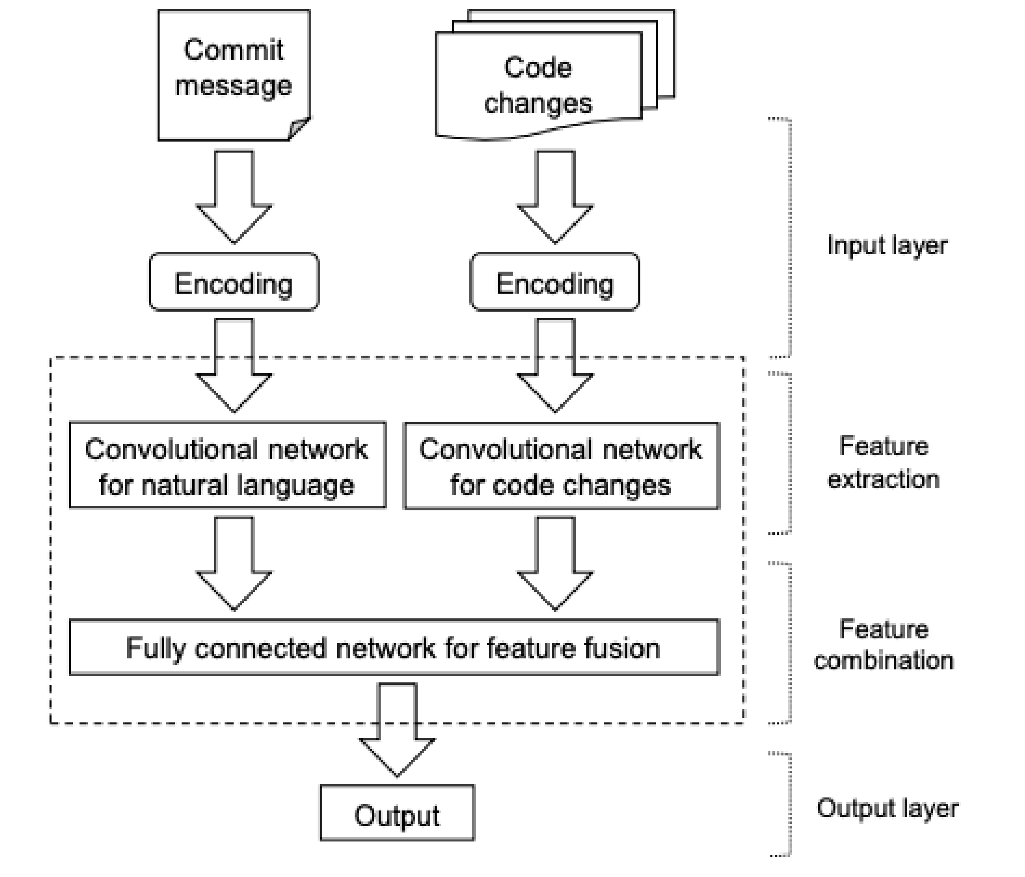

Commit message
Code changes
Encoding
Encoding
Input layer
Convolutional network for natural language
Convolutional network for code changes
Feature extraction
Fully connected network for feature fusion
Feature combination
Output
Output layer

Fig. 3: The general framework of the *Just-In-Time* defect prediction model.

18. Hoang, T., Dam, H. K., Kamei, Y., Lo, D., & Ubayashi, N. (2019, May). DeepJIT: an end-to-end deep learning framework for just-in-time defect prediction. In *2019 IEEE/ACM 16th International Conference on Mining Software Repositories (MSR)* (pp. 34-45). IEEE.

Software Visualization and Deep Transfer Learning for Effective Software Defect Prediction (ICSE 2020)

- In most of the defect prediction models, source code is converted into intermediary representation for processing.
- In this work, the **images of source code** are taken and trained on image classification models.
- Thus, semantic and structural similarity of programs are identified by visually comparing them.
- Built the image classification model on AlexNet platform[19].
- Used deep learning and attention mechanism models to improve the defect prediction w.r.t cross-project defect prediction.

[19] Chen, J., Hu, K., Yu, Y., Chen, Z., Xuan, Q., Liu, Y., & Filkov, V. (2020, June). Software visualization and deep transfer learning for effective software defect prediction. In *Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering* (pp. 578-589).

Tasks in Software Maintenance

- Bug Localization
- Clone Detection
- Feature Location
- Bug report summarization
- Malware detection

Bug Localization

- The task of identifying the bug’s location in the codebase from the input bug report file
- Large number of bug reports for large software systems and multiple versions
- Handling all bug reports and identifying the exact location of bug is a time-consuming task!
- With the help of existing bug reports, ML/DL models along with IR models are developed to automatically find the bug location

DRAST - A Deep Learning and AST Based Approach for Bug Localization

- The approach uses rVSM and DNNs to list out the most probable file for a given input report[23].

- DRAST is a framework, that is evaluated and tested on 7 C projects and 2 Java Projects.

- Its uses a novel source code representation, converting source code blocks into high level AST.

- Shows 90% accuracy in C projects and 70% accuracy in Java Projects.

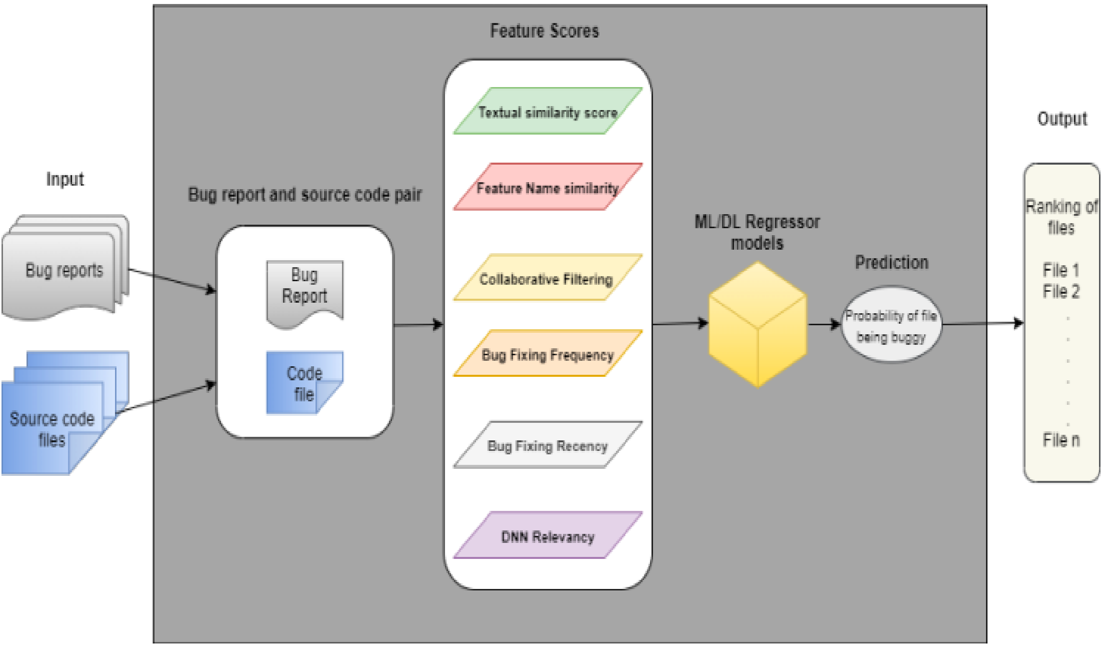


Figure 3: DRAST Pipeline

23. Sangle, S., Muvva, S., Chimalakonda, S., Ponnalagu, K., & Venkoparao, V. G. (2020). DRAST--A Deep Learning and AST Based Approach for Bug Localization. *arXiv preprint arXiv:2011.03449*.

Clone detection

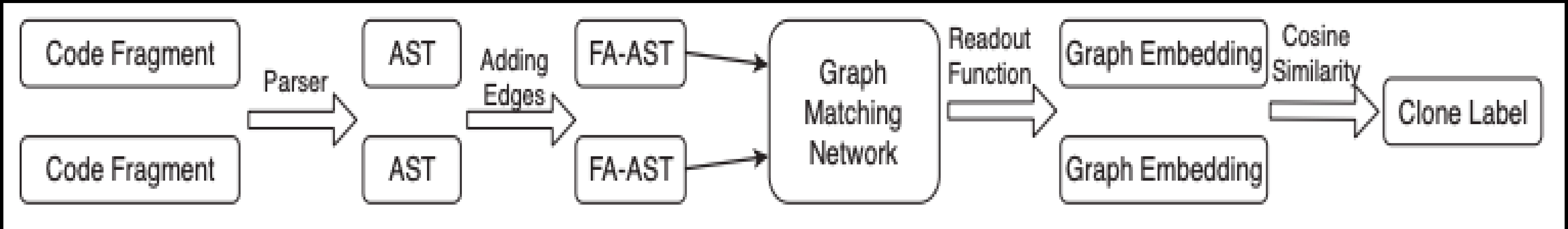
- The task is to identify similar code snippets with same functionalities in different software projects.

- Duplicated code snippets with bugs affects the software quality, leads to high maintenance costs.

- 4 Types of Code Clones:
  - Textual Clones (Type-I)
  - Lexical Clones (Type-II)
  - Syntactic Clones (Type-III)
  - Functional Clones (Type-IV)

Detecting Code Clones with Graph Neural Network and Flow-Augmented Abstract Syntax Tree (SANER 2020)

- The paper argues that considering AST for clone detection is not enough.
- Therefore, they combined AST with control and data flow edges, named as flow-augmented AST (FA-AST)[25].
- Applied two different Graph Neural Networks (GNNs) on FA-AST to compute similarities between code pairs.



[25] Wang, W., Li, G., Ma, B., Xia, X., & Jin, Z. (2020, February). Detecting code clones with graph neural network and flow-augmented abstract syntax tree. In *2020 IEEE 27th International Conference on Software Analysis, Evolution and Reengineering (SANER)* (pp. 261-271). IEEE.

DL-Droid: Deep learning based android malware detection using real devices

- DL-Droid: A deep-learning based system, to detect malicious Android application using dynamic analysis[26].
- It utilizes state-based input generation approach for increasing the code coverage during malware detection.
- DL-Droid achieved 97.8% detection rate (using dynamic analysis) and 99.6% detection rate using dynamic + static analysis.

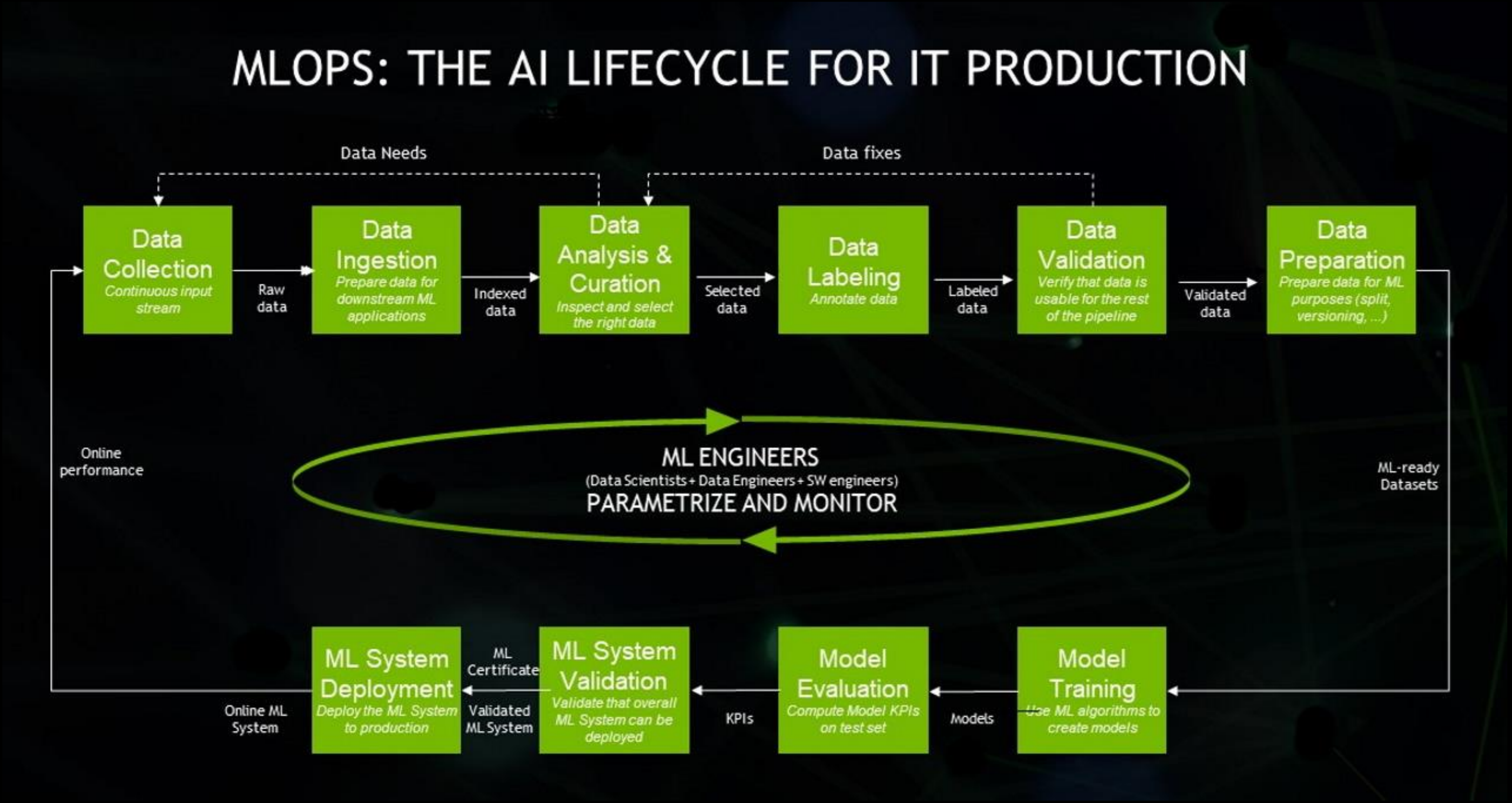
[26] Alzaylaee, M. K., Yerima, S. Y., & Sezer, S. (2020). DL-Droid: Deep learning based android malware detection using real devices. *Computers & Security*, *89*, 101663.

AI in Software Project Planning

Support project management tasks include[8]:

1) Task Assignment
2) Human resource allocation
3) Software Project Scheduling Problem

8. Zhang, W., Yang, Y., & Liu, X. (2020, September). Reducing delay penalty of multiple concurrent software projects based on overtime planning. In *2020 35th IEEE/ACM International Conference on Automated Software Engineering Workshops (ASEW)* (pp. 47-52). IEEE.



MLOPS: THE AI LIFECYCLE FOR IT PRODUCTION

Data Needs

Data fixes

Data Collection
Continuous input stream

Raw data

Data Ingestion
Prepare data for downstream ML applications

Indexed data

Data Analysis & Curation
Inspect and select the right data

Selected data

Data Labeling
Annotate data

Labeled data

Data Validation
Verify that data is usable for the rest of the pipeline

Validated data

Data Preparation
Prepare data for ML purposes (split, versioning, ...)

ML-ready Datasets

Model Training
Use ML algorithms to create models

Models

Model Evaluation
Compute Model KPIs on test set

KPIs

ML System Validation
Validate that overall ML System can be deployed

ML Certificate

Validated ML System

ML System Deployment
Deploy the ML System to production

Online ML System

Online performance

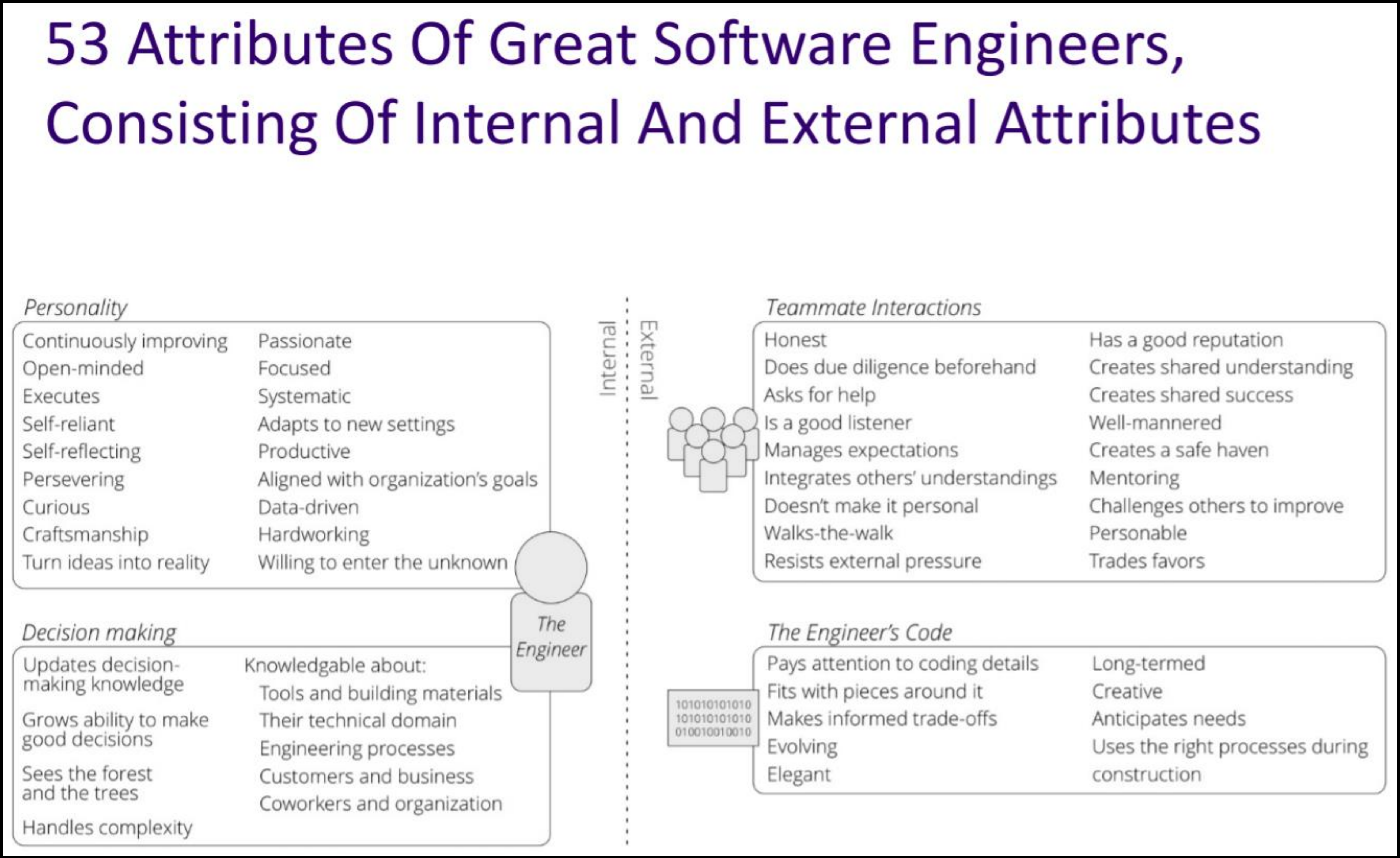
ML ENGINEERS
(Data Scientists + Data Engineers + SW engineers)
PARAMETRIZE AND MONITOR

https://blogs.nvidia.com/blog/2020/09/03/what-is-mlops/

What AI can do for SE?

- Improve software development process, quality and support software professionals throughout the lifecycle
- Decision making support
- 10x, 100x, 1000x productivity?

What Makes A Great Software Engineer? – The Implications for AI



53 Attributes Of Great Software Engineers, Consisting Of Internal And External Attributes

Internal

External

Personality

- Continuously improving
- Open-minded
- Executes
- Self-reliant
- Self-reflecting
- Persevering
- Curious
- Craftsmanship
- Turn ideas into reality
- Passionate
- Focused
- Systematic
- Adapts to new settings
- Productive
- Aligned with organization's goals
- Data-driven
- Hardworking
- Willing to enter the unknown

Decision making

- Updates decision-making knowledge
- Grows ability to make good decisions
- Sees the forest and the trees
- Handles complexity

Knowledgable about:

- Tools and building materials
- Their technical domain
- Engineering processes
- Customers and business
- Coworkers and organization

The Engineer

Teammate Interactions

- Honest
- Does due diligence beforehand
- Asks for help
- Is a good listener
- Manages expectations
- Integrates others' understandings
- Doesn't make it personal
- Walks-the-walk
- Resists external pressure
- Has a good reputation
- Creates shared understanding
- Creates shared success
- Well-mannered
- Creates a safe haven
- Mentoring
- Challenges others to improve
- Personable
- Trades favors

The Engineer's Code

- Pays attention to coding details
- Fits with pieces around it
- Makes informed trade-offs
- Evolving
- Elegant
- Long-termed
- Creative
- Anticipates needs
- Uses the right processes during construction

What AI cannot do for SE?

- Generalized vs specific software engineering tasks – Generalizability!
- Human-driven software engineering tasks
- Explainability, Fairness, Accountability, Transparency, Bias, Ethics, Legal, Social Issues!
- High Computation Resources
- Lack of consensus on privacy and security issues

Energy-Aware AI is distant future!

One NLP model has the same carbon footprint of 5 cars.

Strubell et al. [2019] have observed that training a single machine learning model can generate up to 626,155 pounds of CO2 emissions, which is alarming given the extensive use of machine learning in various applications today

Edge AI?

.poftut.com/what-is-server-farm/

Source: https://www.poftut.com/what-is-server-farm/

Emma Strubell, Ananya Ganesh, and Andrew McCallum. 2019. Energy and policy considerations for deep learning in NLP. *arXiv preprint arXiv:1906.02243* (2019).

SE4AI

Why SE for AI?

- AI systems are data-intensive systems! How do we look at requirments, architecture, code, testing and maintanance in this context?

- Accuracy largely varies when datasets and other environment settings are varied!!
- Can we guarantee
  - robust pipelines and regular updates?
  - updating models already in production?
  - reasonable trade-off between learning rate, updatability & interpretability?

27. Kästner, C., & Kang, E. (2020, October). Teaching Software Engineering for AI-Enabled Systems. In 2020 IEEE/ACM 42nd International Conference on Software Engineering: Software Engineering Education and Training (ICSE-SEET) (pp. 45-48). IEEE.
28. Shneiderman, B. (2020). Human-centered artificial intelligence: Three fresh ideas. AIS Transactions on Human-Computer Interaction, 12(3), 109-124.

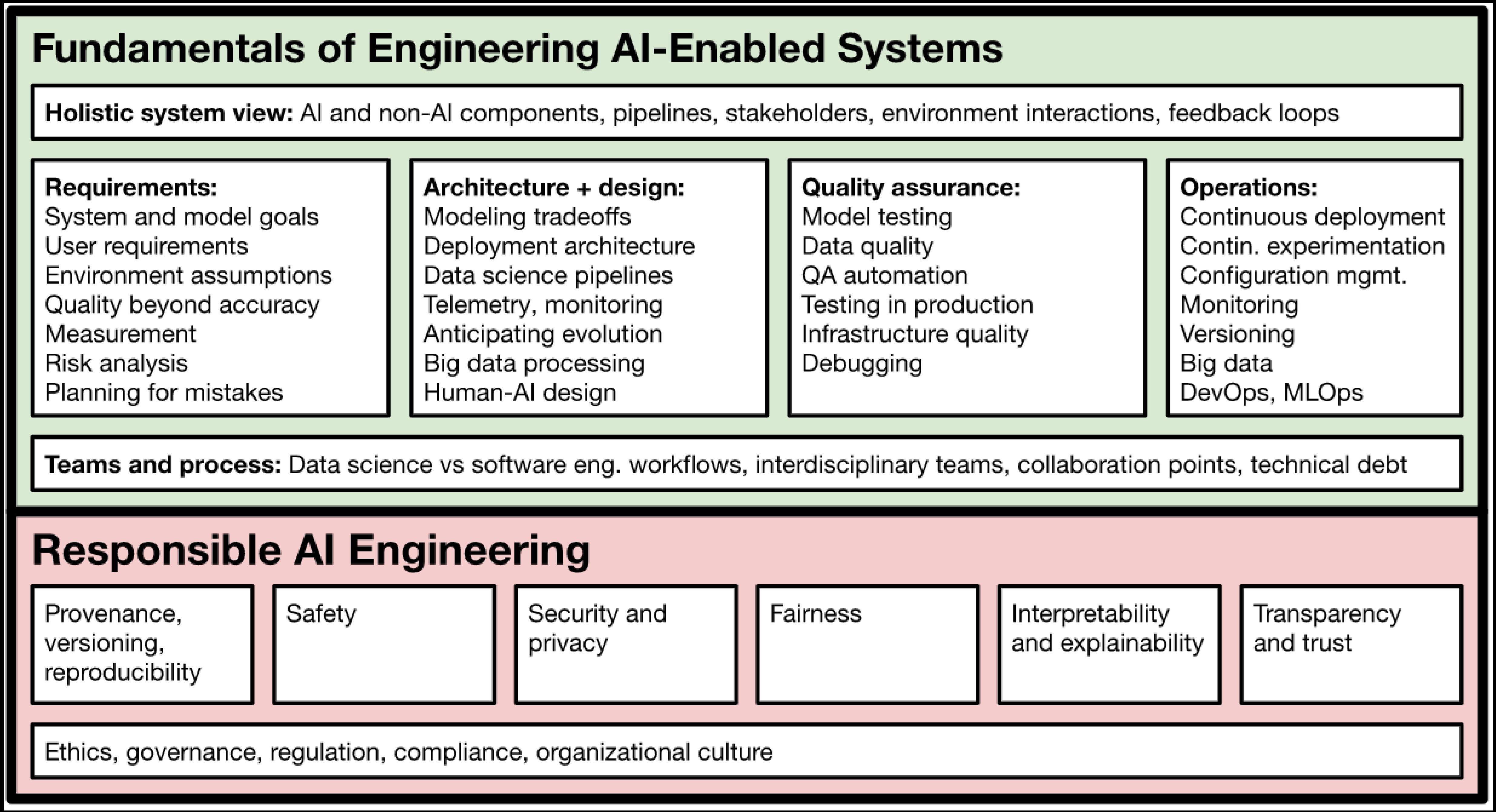
5 Laws of SE for AI

1. AI Software Mostly Isn't About AI
2. AI Software Needs Software Engineers
3. Poor SE Leads to Poor AI
4. Better SE Leads to Better AI
5. SE Needs Special Kinds of AI

29. Menzies, T. (2019). The five laws of SE for AI. *IEEE Software*, *37*(1), 81-85.

SE for AI

A CMU course on SE4AI on how to build, deploy, assure, and maintain ML models, responsible AI systems and so on.



Fundamentals of Engineering AI-Enabled Systems

Holistic system view: AI and non-AI components, pipelines, stakeholders, environment interactions, feedback loops

Requirements:
System and model goals
User requirements
Environment assumptions
Quality beyond accuracy
Measurement
Risk analysis
Planning for mistakes

Architecture + design:
Modeling tradeoffs
Deployment architecture
Data science pipelines
Telemetry, monitoring
Anticipating evolution
Big data processing
Human-AI design

Quality assurance:
Model testing
Data quality
QA automation
Testing in production
Infrastructure quality
Debugging

Operations:
Continuous deployment
Contin. experimentation
Configuration mgmt.
Monitoring
Versioning
Big data
DevOps, MLOps

Teams and process: Data science vs software eng. workflows, interdisciplinary teams, collaboration points, technical debt

Responsible AI Engineering

Provenance, versioning, reproducibility

Safety

Security and privacy

Fairness

Interpretability and explainability

Transparency and trust

Ethics, governance, regulation, compliance, organizational culture

Course available on Github - https://github.com/ckaestne/seai/

An Important Takeaway

- “If you have a hammer, you tend to see every problem as a nail” – Abraham Maslow
- Can we use the hammer of AI only if necessary?

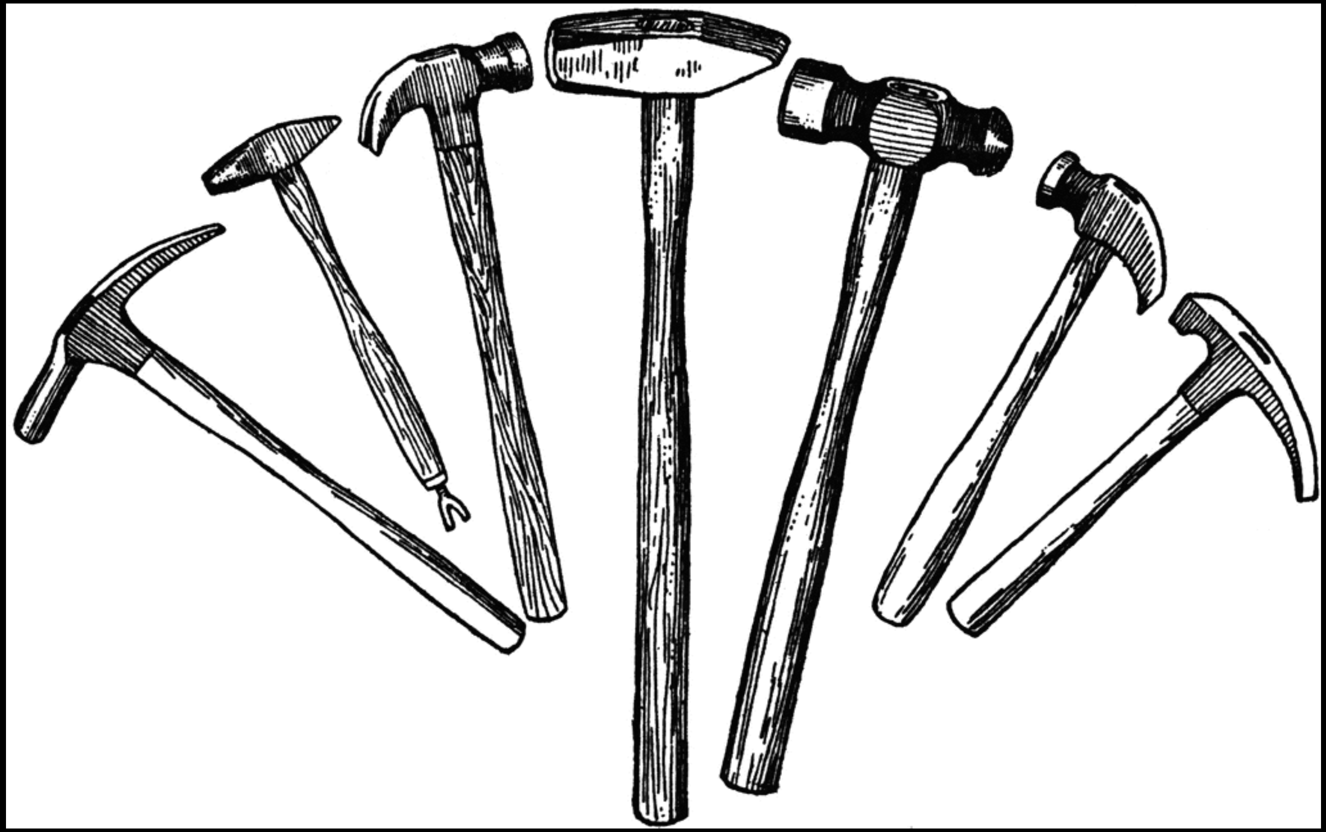
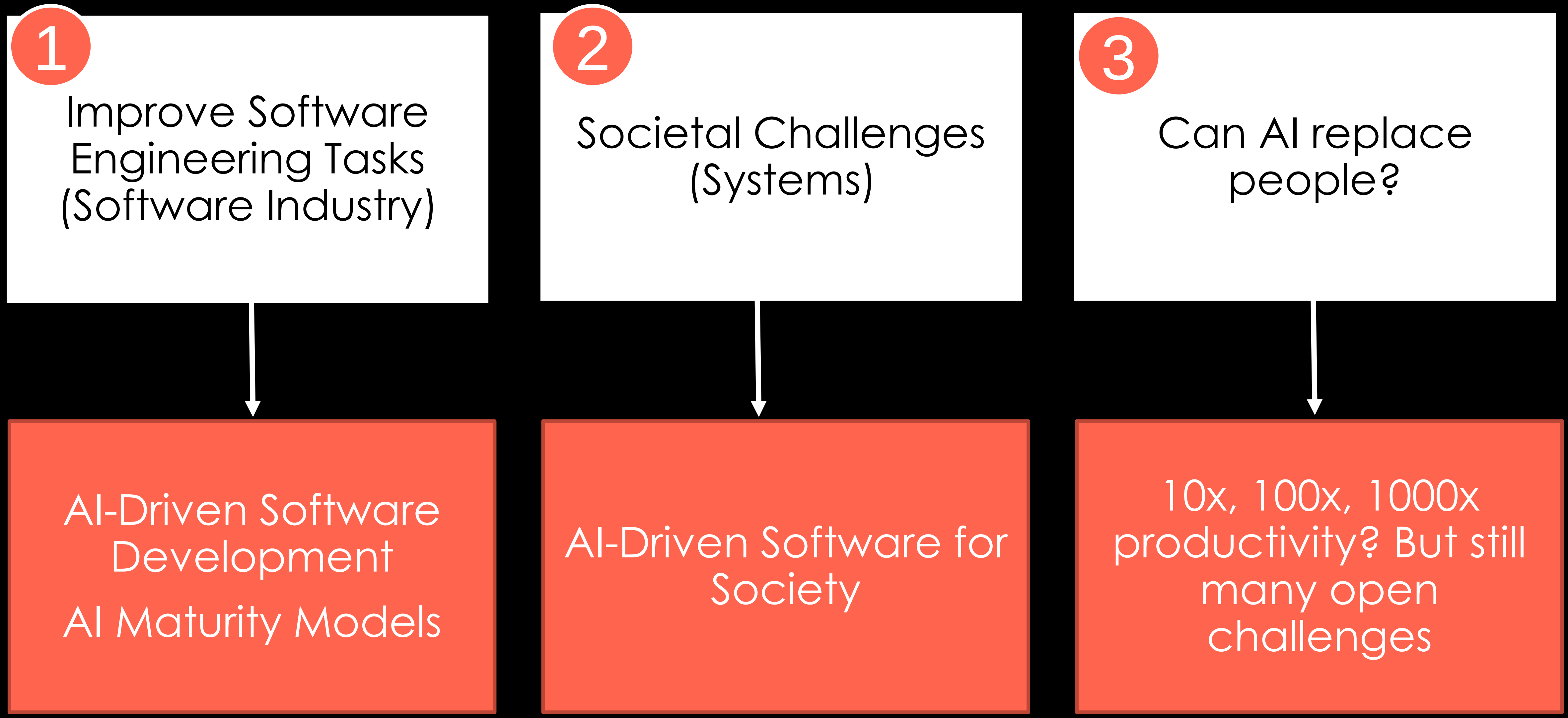


Image Credit: https://www.quora.com/What-is-meant-by-the-Mark-Twain-quote-To-a-man-with-a-hammer-everything-looks-like-a-nail, https://etc.usf.edu/clipart/56300/56373/56373_hammer_types.htm

What are the key take-aways?



1 Improve Software Engineering Tasks (Software Industry)

AI-Driven Software Development

AI Maturity Models

2 Societal Challenges (Systems)

AI-Driven Software for Society

3 Can AI replace people?

10x, 100x, 1000x productivity? But still many open challenges

Can we design Responsible Software?

Can we design Responsible AI Systems?

Can we design Responsible and Ethical AI+SE?

AI that does not harm society
Software that does not harm society

"Necessity is the mother of invention"

Creativity is the father

Passion, Curiosity & Originality are siblings

Capability & Copability are cousins

Inventions & Innovations are heirs!!!

[while luck is the best friend]

भारतीय प्रौद्योगिकी संस्थान तिरुपति

TIRUPATI

Thank you

☺ ☺ ☺ ☺

RISHA ↠

Research in Intelligent Software & Human Analytics Lab

https://rishalab.in/

Questions, Comments, Criticism, Collaborations

ch@iittp.ac.in