Corso di Laurea Magistrale in Informatica

Technical Debt in Sistemi di Intelligenza Artificiale: uno Studio Empirico sullo Stato dell'Arte e della Pratica

Prof. Fabio Palomba
Dott. Fabiano Pecorelli

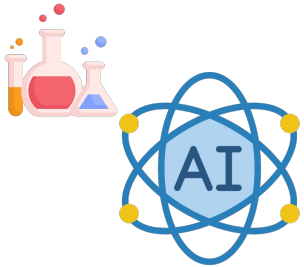
Gilberto Recupito
Matricola: 0522500842

g.recupito@studenti.unisa.it
gilbertrec.github.io
@gilrec

SCAN ME

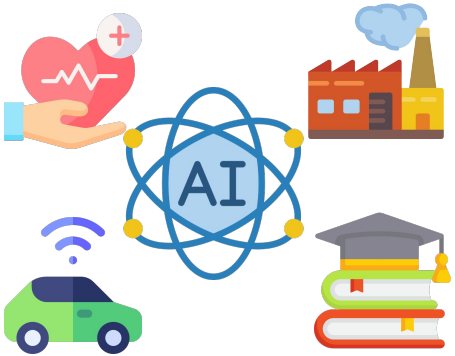
Introduzione

L'intelligenza artificiale ha subito un'evoluzione integrando i modelli AI al sistema software creando un prodotto industriale.



Modelli di AI

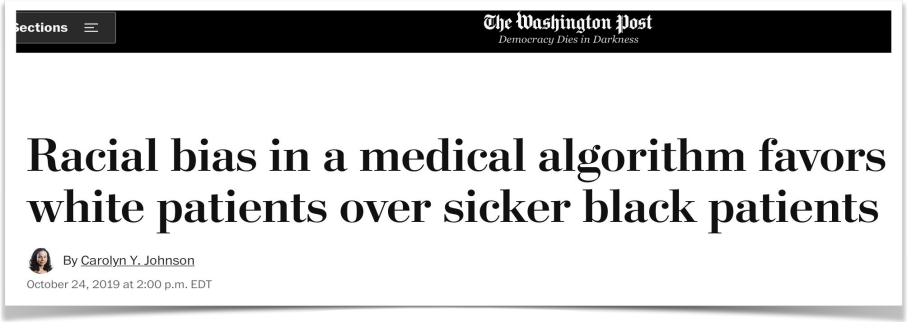
Sistema Software

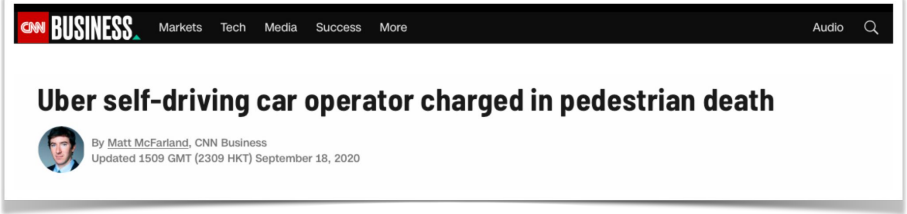


Sistema AI

Problema

Effettuare manutenzione e controllo della qualità su un sistema AI è complicato... ma **fondamentale**.





Un sistema AI con una **scarsa qualità** può causare problemi all'infrastruttura, problemi sociali e all'ambiente circostante, ma soprattutto provocare **danni a persone.**

Problema

Technical Debt

*“un insieme di scelte di design subottimali o soluzioni implementative che possono **influenzare negativamente i dati e la qualità del codice**.”*
-Ward Cunningham, 1992.

Problema

Technical Debt

*"un insieme di scelte di design subottimali o soluzioni implementative che possono **influenzare negativamente i dati e la qualità del codice**."*
-Ward Cunningham, 1992.

Dagli studi presenti in letteratura si riscontra che i sistemi AI sono particolarmente esposti al Technical Debt, in particolare su due tipologie:

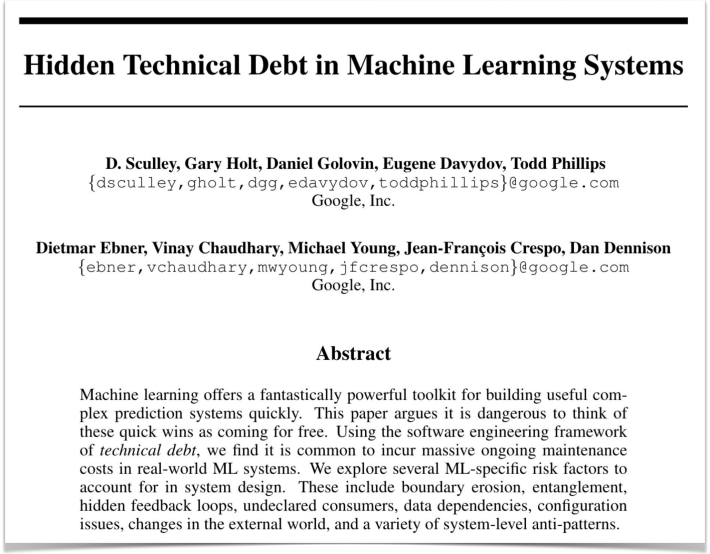
Traditional Technical Debt

Scelte di design subottimali riscontrabili in qualsiasi sistema software.

AI-Technical Debt

Scelte di design subottimali e specifiche per i sistemi AI applicati al codice, ai dati, al modello e all'architettura.

Problema



Hidden Technical Debt in Machine Learning Systems

D. Sculley, Gary Holt, Daniel Golovin, Eugene Davydov, Todd Phillips
{dsculley,gholt,dgg,edavydov,toddphillips}@google.com
Google, Inc.

Dietmar Ebner, Vinay Chaudhary, Michael Young, Jean-François Crespo, Dan Dennison
{ebner,vchaudhary,mwyoung,jfcrespo,dennison}@google.com
Google, Inc.

Abstract

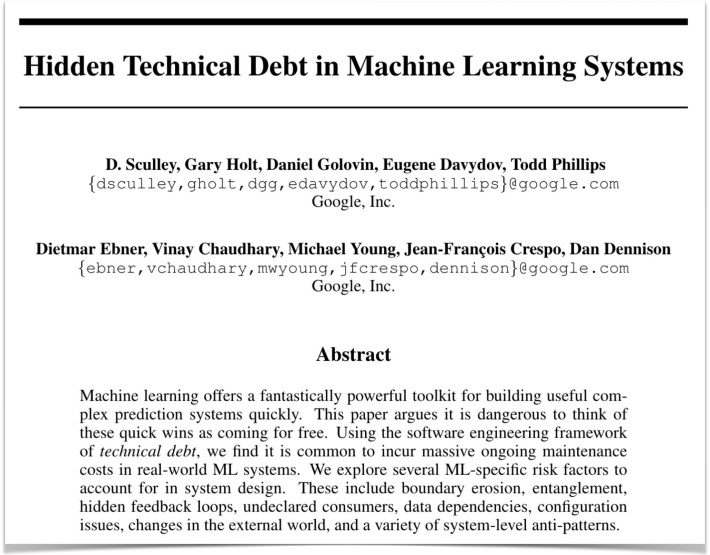
Machine learning offers a fantastically powerful toolkit for building useful complex prediction systems quickly. This paper argues it is dangerous to think of these quick wins as coming for free. Using the software engineering framework of *technical debt*, we find it is common to incur massive ongoing maintenance costs in real-world ML systems. We explore several ML-specific risk factors to account for in system design. These include boundary erosion, entanglement, hidden feedback loops, undeclared consumers, data dependencies, configuration issues, changes in the external world, and a variety of system-level anti-patterns.

Sculley et. Al hanno effettuato uno studio al fine di esplorare e avviare la definizione delle istanze di Technical Debt per i sistemi di Machine Learning.

Definizione

Pipeline Jungles: Le Pipeline jungles possono crescere ogni volta che viene aggiunta informazione o nuovi dati al sistema. Queste nuove informazioni richiedono di inserire all'interno del processo nuove operazioni di elaborazione dei dati. In questo modo il processo incrementa e la difficoltà di effettuare manutenzione cresce in modo proporzionale alla complessità del sistema.

Problema



Hidden Technical Debt in Machine Learning Systems

D. Sculley, Gary Holt, Daniel Golovin, Eugene Davydov, Todd Phillips
{dsculley,gholt,dgg,edavydov,toddphillips}@google.com
Google, Inc.

Dietmar Ebner, Vinay Chaudhary, Michael Young, Jean-François Crespo, Dan Dennison
{ebner,vchaudhary,mwyoung,jfcrespo,dennison}@google.com
Google, Inc.

Abstract

Machine learning offers a fantastically powerful toolkit for building useful complex prediction systems quickly. This paper argues it is dangerous to think of these quick wins as coming for free. Using the software engineering framework of *technical debt*, we find it is common to incur massive ongoing maintenance costs in real-world ML systems. We explore several ML-specific risk factors to account for in system design. These include boundary erosion, entanglement, hidden feedback loops, undeclared consumers, data dependencies, configuration issues, changes in the external world, and a variety of system-level anti-patterns.

Tuttavia la categorizzazione proposta non è ancora sufficiente per il professionista di sistemi AI per poter effettuare pratiche di gestione di Technical Debt in quanto:

- Esistono ulteriori tipologie di Technical Debt.
- Non è presente uno studio sulla natura e gli effetti che possono causare la presenza di Technical Debt nel sistema.
- Non sono presenti attuali strategie di identificazione e mitigazione delle istanze di Technical Debt.

Obiettivo di Ricerca

Contribuire alla definizione della tassonomia di AI Technical Debt, al fine di fornire agli sviluppatori:

Catalogo delle diverse istanze di AI Technical Debt.

Definizioni sugli effetti che ogni istanza di Technical Debt può causare alla qualità.

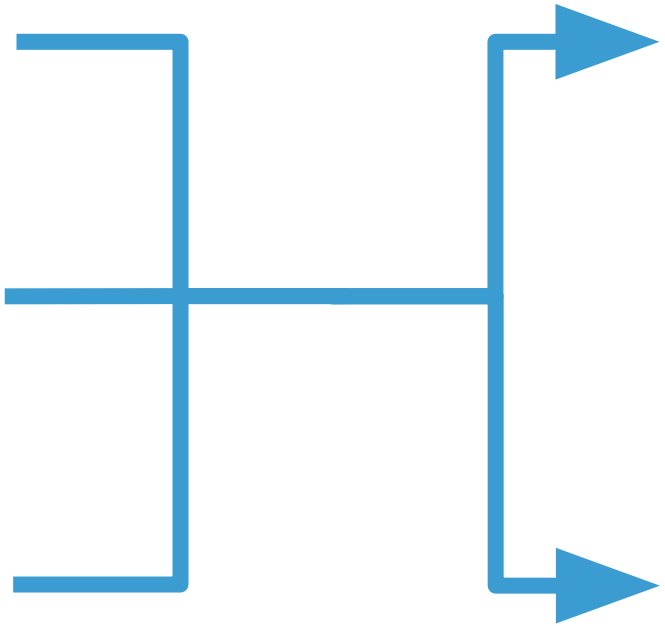
Guida su come identificare e mitigare il Technical Debt.

Soluzioni

Catalogo delle diverse istanze di AI Technical Debt.

Definizioni sugli effetti che ogni istanza di Technical Debt può causare alla qualità.

Guida su come identificare e mitigare il Technical Debt.



Analisi dello stato dell'Arte.

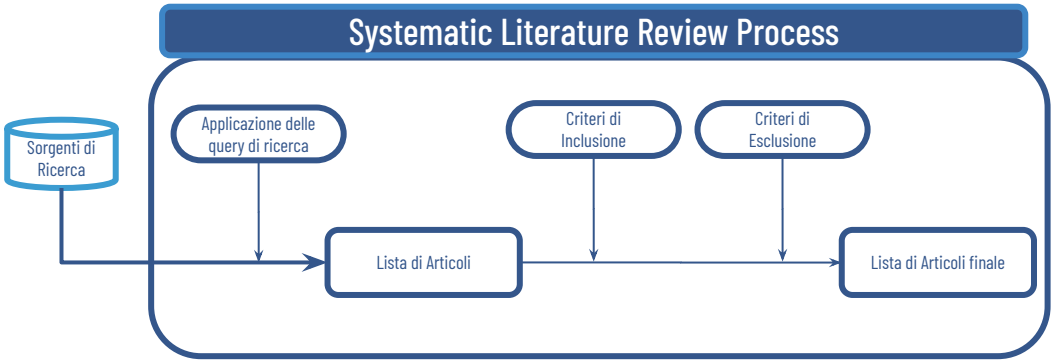
Survey per analizzare il punto di vista degli sviluppatori.

SLR – Research Map

RQ 1.1: Quali tipologie di Technical Debt sono presenti nei sistemi AI?

RQ 1.2: Quali sono gli approcci o i tool per identificare e mitigare TD nei sistemi AI?

SLR Metodologia



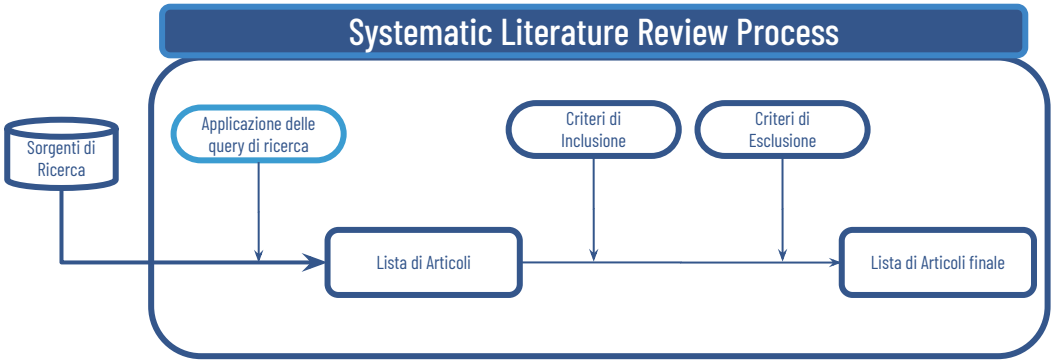
ACM DL DIGITAL LIBRARY

IEEE Xplore® Digital Library

Scopus

Clarivate Web of Science™

SLR Metodologia



Technical Debt Keywords

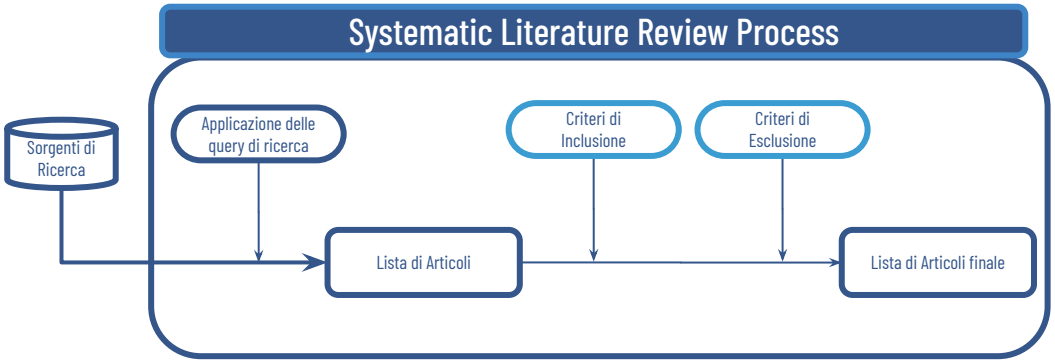
technical debt,
tech debt,
requirements debt,
data debt,
data smell*,
code debt...

AND

Artificial Intelligence Keywords

Deep learning,
Artificial intelligence,
AI,
ML,
Machine learning...

SLR Metodologia



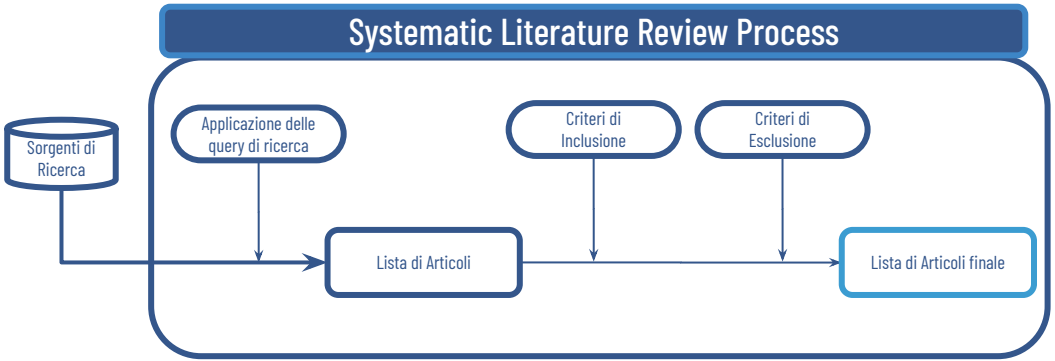
Inclusion Criteria

Tipologie di TD
Tool/strategie di identificazione
Tool/strategie mitigazione

Exclusion Criteria

Articoli non scritti in lingua inglese
Articoli non soggetti a peer review
Articoli duplicati
Position paper
Pubblicazioni non accessibili

SLR Metodologia



Coding Schema

- Scopo o topic
- Tipo di Risorsa (Conferenza, Journal...)
- Istanze di TD descritte
- Definizione delle istanze di TD
- Strategie o tool di identificazione
- Strategie o tool di mitigazione

Survey — Research Map

RQ 1.1: Quali sono le tipologie di code debt e architectural debt più **frequenti** secondo la percezione degli sviluppatori AI?

RQ 1.2: Quali sono le tipologie di code debt e architectural debt più **problematiche** secondo la percezione degli sviluppatori AI?

RQ 1.3: Qual è **l'impatto** causato dai code debt e architectural debt secondo la percezione degli sviluppatori AI?

RQ 1.4: Quali sono le strategie utilizzate dagli sviluppatori AI per l'identificazione e la mitigazione di code debt e architectural debt?

Survey – Oggetti Sperimentali

Architectural Debt

Pipeline Jungles
Correction Cascades
Undeclared Consumers
Jumbled Model Architecture

Code Debt

Glue Code
Multiple Language Smell
Scattered Use of ML Libraries
Unwanted Debugging Code,
Deep God File

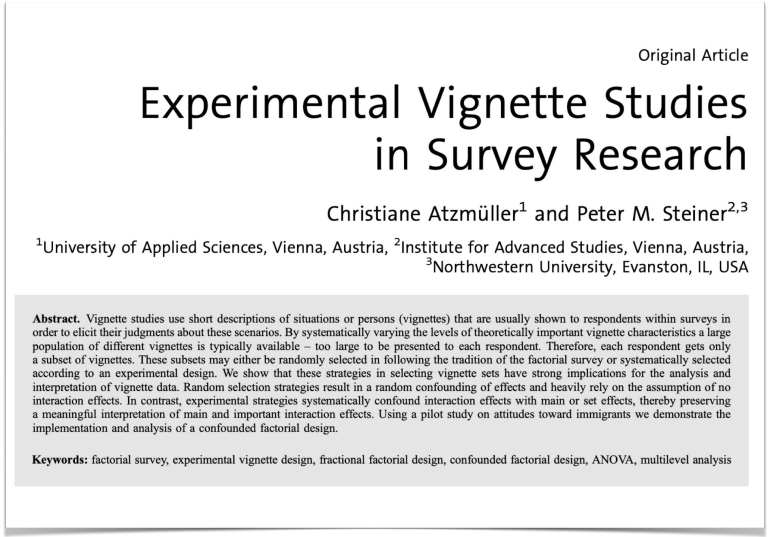
Survey – Vignetta

Definizione

Pipeline Jungles: Pipeline jungles can evolve each time a new input data or information is added. This new information requires adding to the process joins, sampling, and intermediate steps to process data. So in this way the pipeline increases, and the cost to detect it and maintain it is high.

Vignetta

Pipeline Jungles: Suppose that **you are working** on a machine learning pipeline application. During the pipeline definition of the model, **you see that each** part of the team adds several steps for each phase of the pipeline, and **you notice that** the entire pipeline starts growing more.



Original Article

Experimental Vignette Studies in Survey Research

Christiane Atzmüller[1] and Peter M. Steiner[2,3]

[1]University of Applied Sciences, Vienna, Austria, [2]Institute for Advanced Studies, Vienna, Austria, [3]Northwestern University, Evanston, IL, USA

Abstract. Vignette studies use short descriptions of situations or persons (vignettes) that are usually shown to respondents within surveys in order to elicit their judgments about these scenarios. By systematically varying the levels of theoretically important vignette characteristics a large population of different vignettes is typically available – too large to be presented to each respondent. Therefore, each respondent gets only a subset of vignettes. These subsets may either be randomly selected in following the tradition of the factorial survey or systematically selected according to an experimental design. We show that these strategies in selecting vignette sets have strong implications for the analysis and interpretation of vignette data. Random selection strategies result in a random confounding of effects and heavily rely on the assumption of no interaction effects. In contrast, experimental strategies systematically confound interaction effects with main or set effects, thereby preserving a meaningful interpretation of main and important interaction effects. Using a pilot study on attitudes toward immigrants we demonstrate the implementation and analysis of a confounded factorial design.

Keywords: factorial survey, experimental vignette design, fractional factorial design, confounded factorial design, ANOVA, multilevel analysis

Survey - Processo

Reclutamento Partecipanti | Prescreening | Mapping Partecipante/Gruppo Sperimentale | Test Pilota | Diffusione del Survey

- **500 partecipanti reclutati** tramite la piattaforma Prolific, effettuando una selezione specifica su **esperti** che hanno i seguenti prerequisiti:

Prolific

-Esperienza in sistemi di **Intelligenza Artificiale**
-Conoscenza delle tecniche di **sviluppo e ingegneria del software.**

Survey - Processo

Reclutamento Partecipanti

Prescreening

Mapping Partecipante/Gruppo Sperimentale

Test Pilota

Diffusione del Survey

- Ai partecipanti è stato sottoposto un questionario iniziale per valutare l'idoneità delle competenze, analizzando:

—Competenze in **Programmazione**
—Competenze in **Intelligenza Artificiale**
—Competenze in **Ingegneria del Software**

Survey - Processo

Reclutamento Partecipanti → Prescreening → **Mapping Partecipante/Gruppo Sperimentale** → Test Pilota → Diffusione del Survey

- I partecipanti sono stati **assegnati ai gruppi sperimentali** per analizzare le istanze di AI Technical Debt:

Gruppo Partecipanti	Gruppo Sperimentale
Gruppo 1	Gruppo A+B
Gruppo 2	Gruppo B+C
Gruppo 3	Gruppo A+C

Gruppo Sperimentale	Istanze di Ai Technical Debt
Gruppo A	Glue Code, Multiple Language Smell, Undeclared Consumers.
Gruppo B	Pipeline Jungles, Correction Cascades, Scattered Use of ML Libraries.
Gruppo C	Jumbled Model Architecture, Unwanted Debugging Code, Deep God File.

Survey - Processo

Reclutamento Partecipanti → Prescreening → Mapping Partecipante/Gruppo Sperimentale → **Test Pilota** → Diffusione del Survey

- Il questionario progettato è stato **inviato a un sottoinsieme di 6 esperti** del settore al fine di **migliorare la qualità e la comprensibilità delle vignette e delle domande sottoposte.**

Survey - Processo

Reclutamento Partecipanti → Prescreening → Mapping Partecipante/Gruppo Sperimentale → Test Pilota → **Diffusione del Survey**

- Infine, il survey è stato sottomesso ai candidati dell'esperimento al fine di raccogliere le seguenti informazioni sulle istanze di AI Technical Debt:

(1) Frequenza

(2) Problematicità

(3) Strategie di identificazione e mitigazione

Risultati Preliminari - SLR

80

Articoli Analizzati

Collezionati dalle sorgenti consultate.

14

Articoli Inclusi

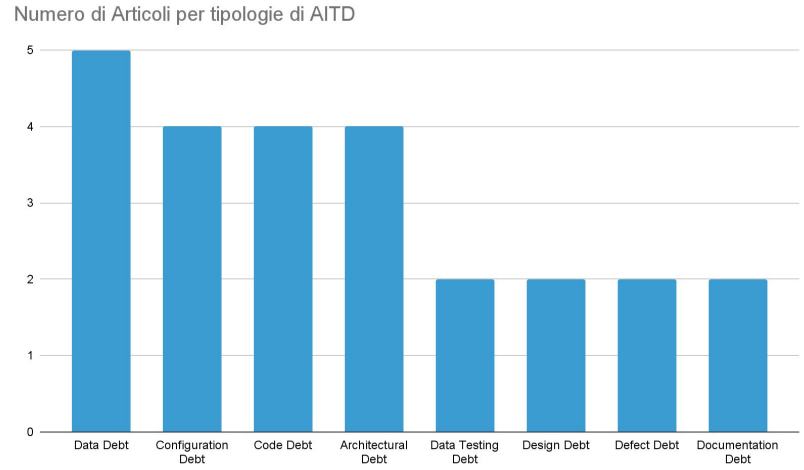
Tramite il processo di analisi preliminare su titolo e abstract.

8

Articoli Finali

Estratti dal processo di analisi completa dell'articolo.

Risultati SLR
RQ1.1: Tipologie



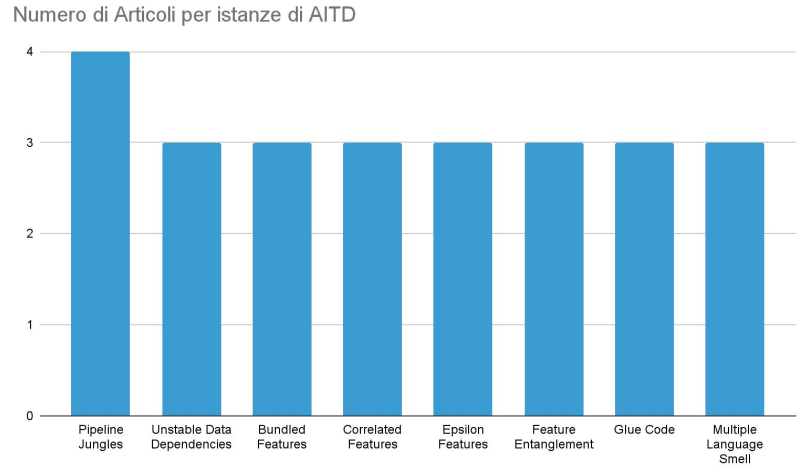
01

Data Debt più riscontrato

02

Alto riscontro per Configuration, Code e Architectural Debt

Risultati SLR
RQ1.1: Istanze



01

Pipeline Jungles è l'istanza più riscontrata

02

Alto riscontro per le istanze relative ai dati

03

Alto riscontro per alcune istanze relative al codice e all'architettura

Risultati SLR
RQ1.2 : Tecniche

- Attualmente, la letteratura non presenta un insieme di tecniche utili a identificare e mitigare le istanze di AITD nel software AI.
- I professionisti di intelligenza artificiale potenzialmente **concentrano i loro sforzi** al fine di **massimizzare le performance del modello**, piuttosto che la manutenzione e l'evoluzione del codice AI.

Definizione

Epsilon Feature: introduzione di un attributo dell'osservazione all'interno dello schema dei dati che garantisce un basso guadagno di performance al modello

Identificazione

Leave One Out Feature Evaluation: Viene creato un insieme di attributi del modello *F* che non contiene un determinato attributo sotto analisi. Se le differenze di performance del modello non sono significative, allora l'attributo sotto analisi è un'istanza di *Epsilon Feature*

Risultati Preliminari - Survey

500

Partecipanti Reclutati

Dalla piattaforma Prolifc.

54

Partecipanti idonei

Alla conduzione dell'investigazione tramite survey.

36

Risposte

In media raccolte per ogni istanza di AI Technical Debt.

Risultati Survey
RQ2.1: Frequenza

AITD	Mediana	Media	Dev.Std.	Moda
Glue Code	3.0	2.45	1.11	3.0
Pipeline Jungles	2.5	2.69	1.17	2.0
Jumbled Model Architecture	2.0	2.53	1.02	2.0

01

Pipeline Jungles, Glue Code e Jumbled Model Architecture sono le più diffuse secondo gli sviluppatori.

02

In totale, nessuna istanza è stata riscontrata dagli sviluppatori con alta frequenza nei sistemi AI.

Risultati Survey
RQ2.2: Pericolosità

AITD	Mediana	Media	Dev.Std.	Moda
Undeclared Consumers	4.0	3.73	1.32	5.0
Pipeline Jungles	4.0	3.47	1.13	4.0
Jumbled Model Architecture	4.0	3.47	1.0	4.0

01

Undeclared Consumers risulta l'istanza più pericolosa di AI Technical Debt.

02

In totale, le istanze Undeclared Consumers, Pipeline Jungles e Jumbled Model Architecture risultano essere altamente pericolose.

Risultati Survey – RQ2.3 Impatto

Di seguito sono riportate le istanze che, secondo la percezione degli sviluppatori, riportano un alto effetto sugli aspetti di qualità del software.

AITD	Mediana della distribuzione delle risposte					
	Effetto sulle altre componenti	Comprensibilità	Evoluzione	Performance	Accoppiamento	Manutenibilità
Undeclared Consumers	**4.0**	**4.0**	**4.0**	**4.0**	**4.0**	3.0
Pipeline Jungles	**4.0**	**4.0**	**4.0**	**4.0**	**4.0**	**4.0**
Jumbled Model Architecture	3.0	**4.0**	**4.0**	3.0	**4.0**	**4.0**

Risultati Survey – RQ2.4 Tecniche

Di seguito sono riportati gli approcci di identificazione e mitigazione delle minacce più pericolose.

AITD	Percentuale delle risposte			
	Manual Inspection	Automatic Inspection	Professional Team	Nessun approccio
Undeclared Consumers	**32.26%**	9.68%	22.58%	**35.48%**
Pipeline Jungles	**46.34%**	12.2%	**24.39%**	17.07%
Jumbled Model Architecture	**55%**	10%	15%	20%

Conclusioni

AITD non diffuso ma pericoloso

Anche se è stata rilevata una bassa diffusione, gli sviluppatori confermano la pericolosità di AITD.

AITD impatta la qualità

Pipeline Jungles, Jumbled Model Architecture e Undeclared Consumers sono percepite impattanti sugli aspetti di qualità per i sistemi AI.

Nessuna soluzione automatica

Attualmente non esistono tecniche automatiche di identificazione e mitigazione.

Sviluppi Futuri

Concentrare gli studi su data debt.

Comprendere se ulteriori tipologie, come quella riscontrata, possono ulteriormente danneggiare il sistema AI.

Definire tecniche di identificazione e mitigazione

Per sostenere gli sviluppatori durante la gestione di AITD.

Analisi delle cause di AITD

Studio sulle possibili cause che possono portare gli sviluppatori a introdurre AITD nei sistemi.

Soluzioni

Catalogo delle diverse istanze di AI Technical Debt.

Definizioni sugli effetti che ogni istanza di Technical Debt può causare alla qualità.

Guida su come identificare e mitigare il Technical Debt.

Analisi dello stato dell'Arte.

Survey per analizzare il punto di vista degli sviluppatori.

Stato della Pratica

Reclutamento Partecipanti → Prescreening → Mapping Partecipante/Gruppo Sperimentale → Test Pilota → **Diffusione del Survey**

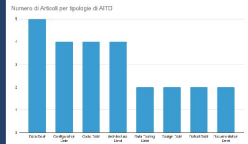
- Infine, il survey è stato sottomesso ai candidati dell'esperimento al fine di raccogliere le seguenti informazioni sulle istanze di AI Technical Debt:

(1) Frequenza

(2) Problematicità

(3) Strategie di identificazione e mitigazione

Risultati SLR - Tipologie



01 **Data Debt più riscontrato**

02 **Alto riscontro per Configuration, Code e Architectural Debt**

Risultati Survey - Pericolosità

AITD	Mediana	Media	Dev.Std.	Moda
Undeclared Consumers	4,0	3.73	1.32	5,0
Pipeline Jungles	4,0	3.47	1.13	4,0
Jumbled Model Architecture	4,0	3.47	1.0	4,0

01 **Undeclared Consumers risulta l'istanza più pericolosa di AI Technical Debt.**

02 **In totale, le istanze Undeclared Consumers, Pipeline Jungles e Jumbled Model Architecture risultano essere altamente pericolose.**

g.recupito@studenti.unisa.it
gilbertrec.github.io
@gilrec

Grazie per l'attenzione!

SCAN ME