UNIVERSITÀ DEGLI STUDI DELL'AQUILA
DISIM Dipartimento di Ingegneria e Scienze dell'Informazione e Matematica
Bộ Giáo dục và Đào tạo
VIỆN NGHIÊN CỨU CAO CẤP VỀ TOÁN

Applications of Recommender Systems and Machine Learning in Software Engineering

Dr. Phuong Nguyen
University of L'Aquila, Italy

Seminar, August 11th 2022

Short Biography

Nguyễn Thanh Phương
Đại học Tổng hợp L'Aquila, Cộng hòa Ý
Email: phuong.nguyen@univaq.it

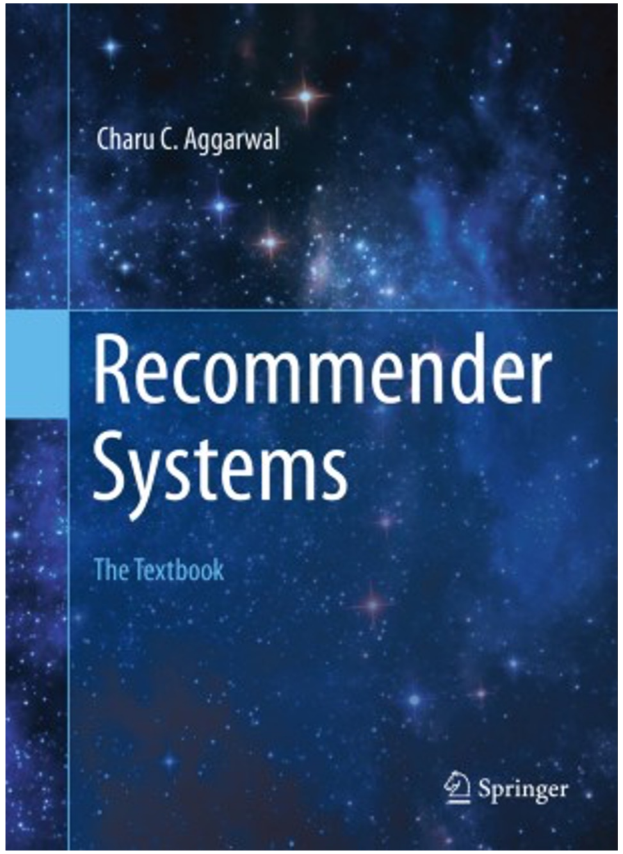
- Diploma and MSc in Information Technology, HUST
- PhD in Computer Science, University of Jena (Germany)
- Lecturer at FPT University, Duy Tan University
- Postdoctoral researcher, Polytechnic University of Bari (Italy)
- Postdoctoral researcher, University of L'Aquila (Italy)
- From January 2022 - now: Assistant Professor, University of L'Aquila

Agenda

- Introduction
- Recommender Systems
- Machine Learning and Deep Learning
- Notable applications in Software Engineering
- Ongoing research issues
- Questions and answers

Recommender Systems

Recommender Systems



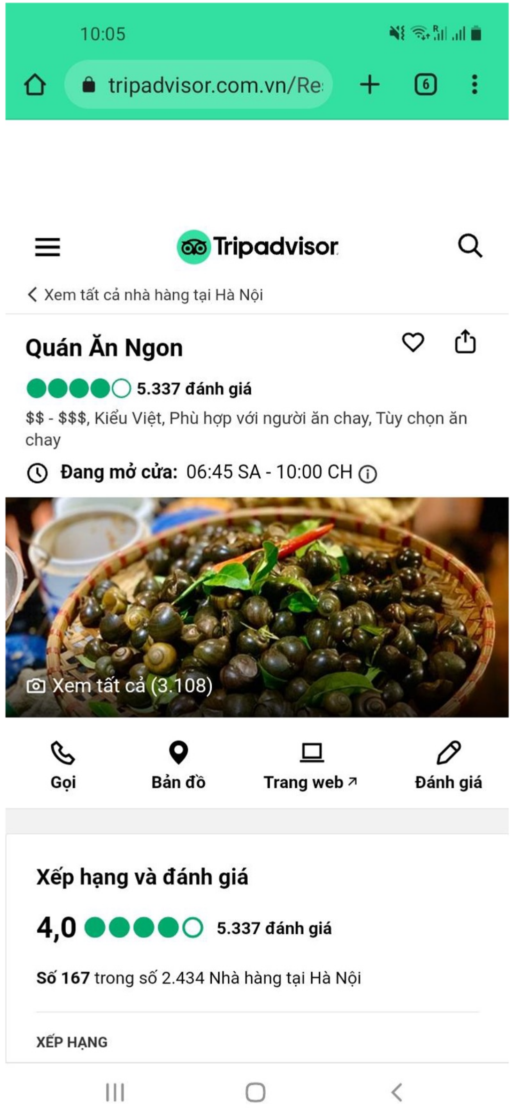
"The basic principle of recommendations is that significant dependencies exist between user and item-centric activity."

Charu C. Aggarwal, ***Recommender Systems***, Springer

User-Item Matrix

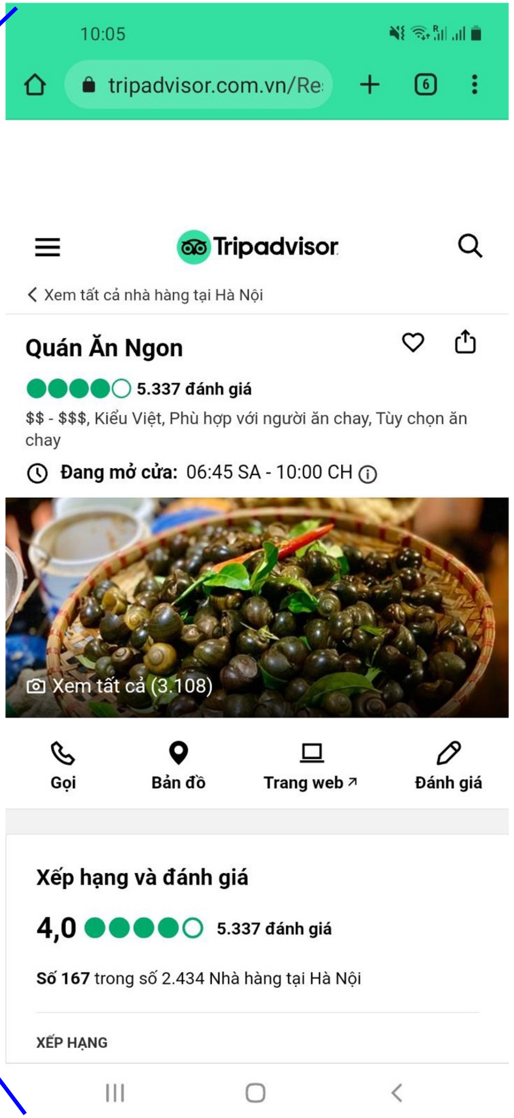
	Lã Vọng	Phở 24	Quán Ăn Ngon
Việt	3	5	5
Minh	3	4	5
Hà	4	5	?

- User-item matrix: Ratings given to restaurants by customers



User-Item Matrix

	Lã Vọng	Phở 24	Quán Ăn Ngon
Việt	3	5	5
Minh	3	4	5
Hà	4	5	**5**



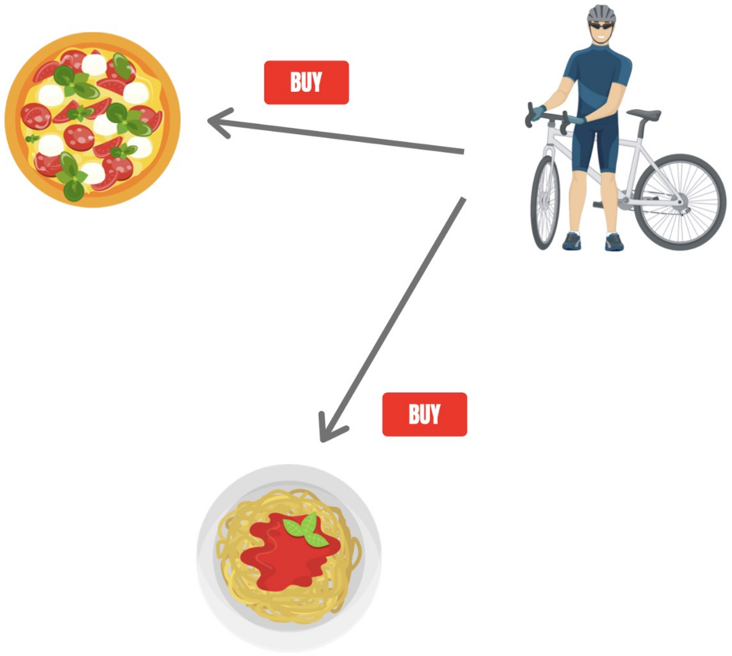
- User-item matrix: Ratings given to restaurants by customers

Recommender Systems

NETFLIX amazon YouTube

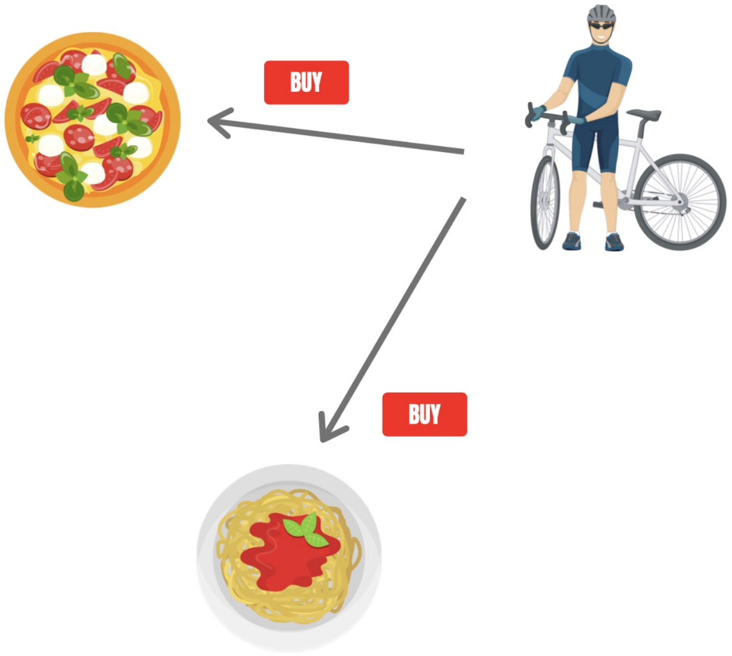
- provide users with suggestions based on their preferences
- recommend items that normally fit users' taste/need
- have been widely used in online systems, e.g., YouTube, Booking.com, TripAdvisor to customize recommendations

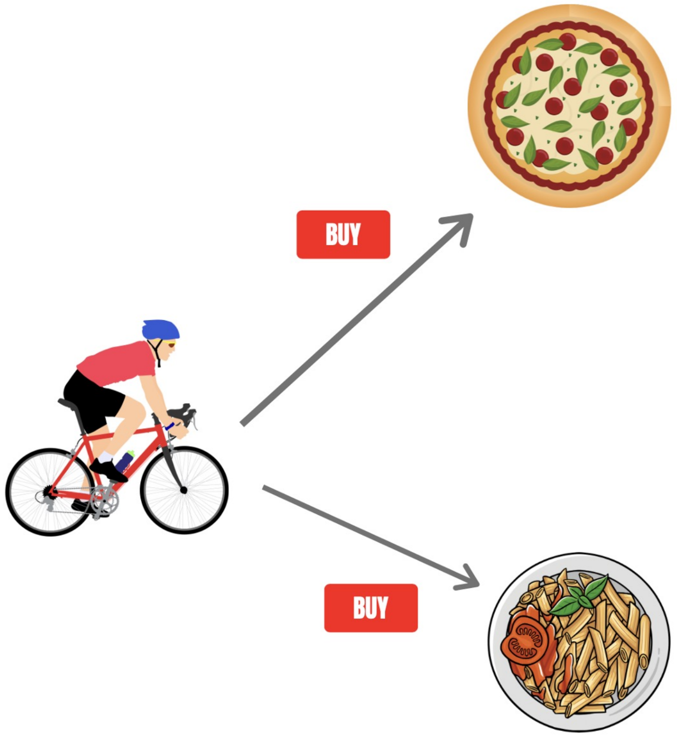
Collaborative-Filtering RecSys



- Providing users with suggestions that fit their taste/need

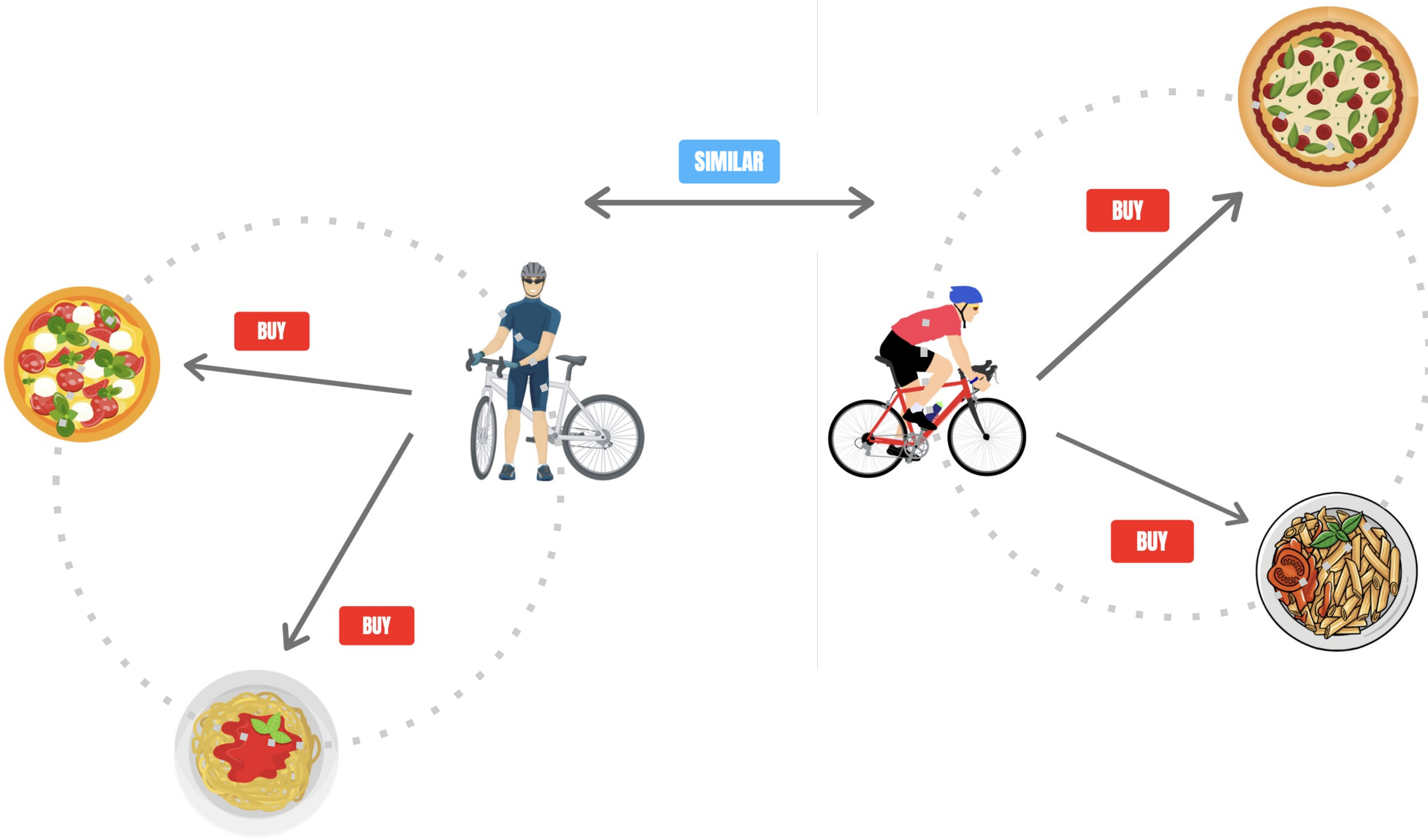
Collaborative-Filtering RecSys





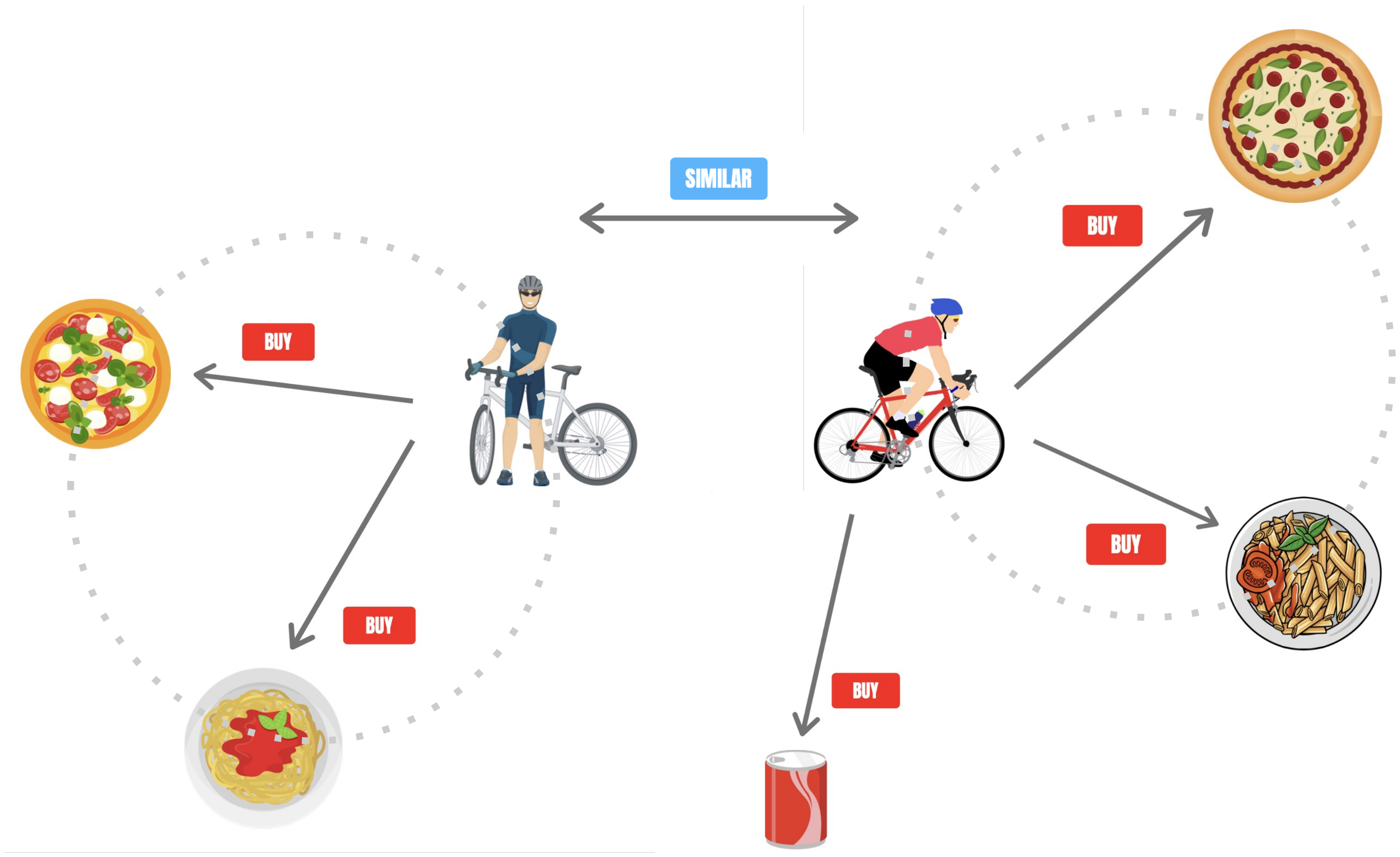
- Providing users with suggestions that fit their taste/need
- Being based on the preferences of similar users

Collaborative-Filtering RecSys



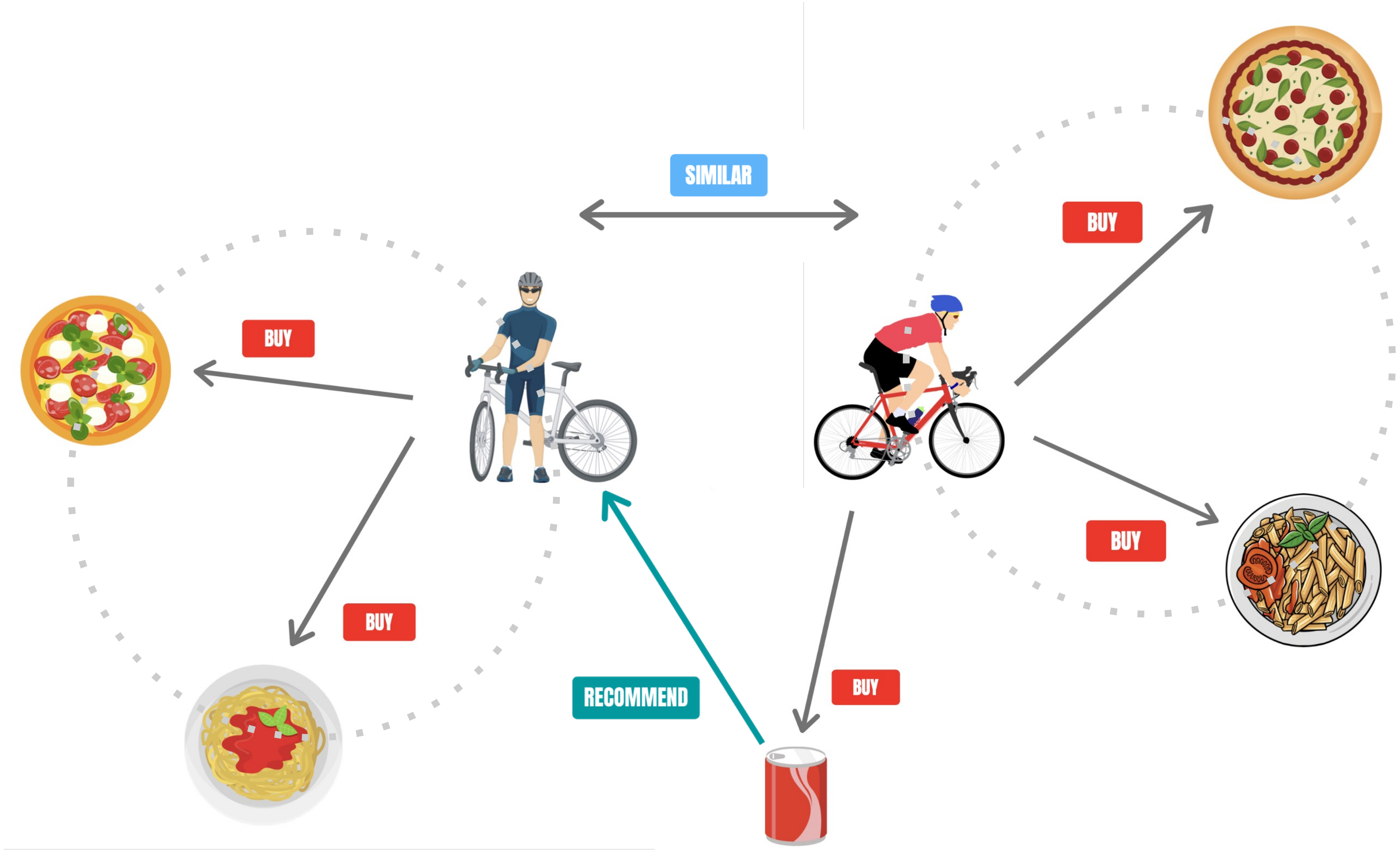
- Providing users with suggestions that fit their taste/need
- Being based on the preferences of similar users

Collaborative-Filtering RecSys



- Providing users with suggestions that fit their taste/need
- Being based on the preferences of similar users

Collaborative-Filtering RecSys



- Providing users with suggestions that fit their taste/need
- Being based on the preferences of similar users

Collaborative-Filtering RecSys

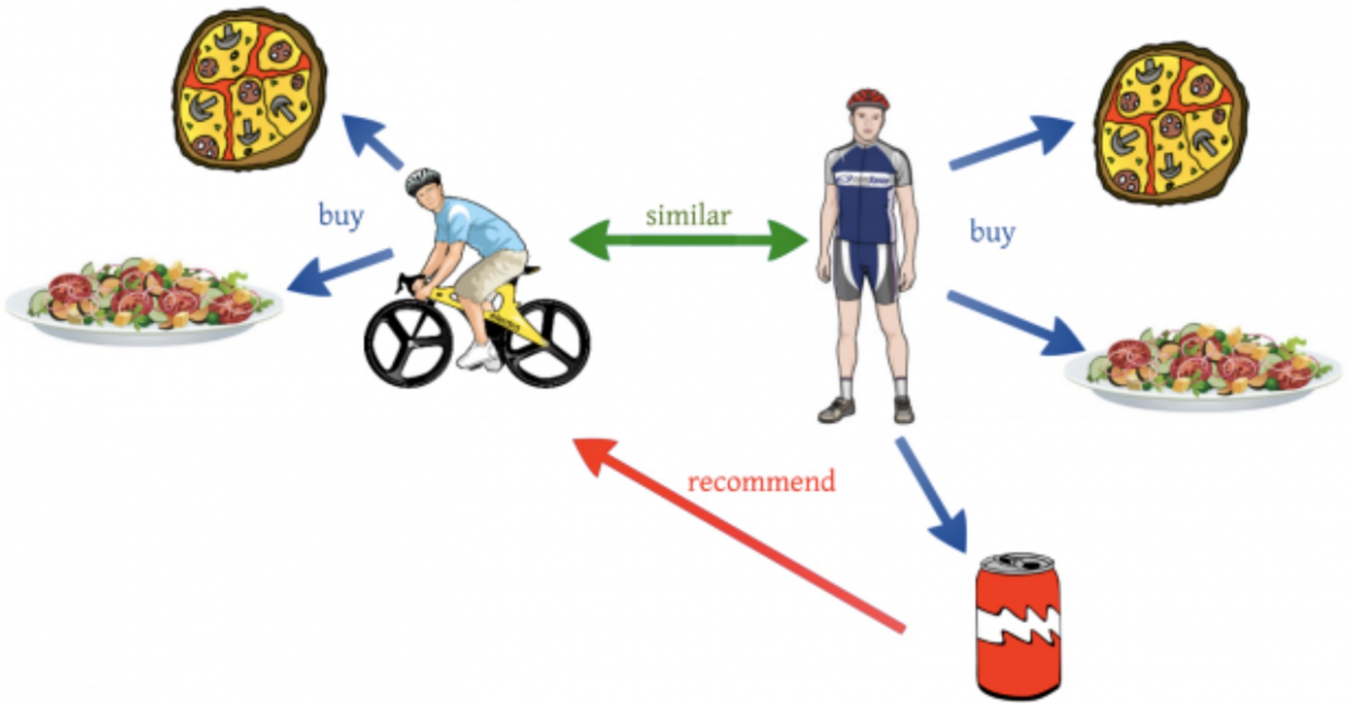


Image source: https://towardsdatascience.com/various-implementations-of-collaborative-filtering-100385c6dfe0

- Providing users with suggestions that fit their taste/need
- Being based on the preferences of similar users

Example: Movie Recommendation

Image source: https://usa.newonnetflix.info/info/70140403

Image source: https://www.whats-on-netflix.com/news/what-csi-titles-are-on-netflix/

... because you have watched you should also watch this

NETFLIX

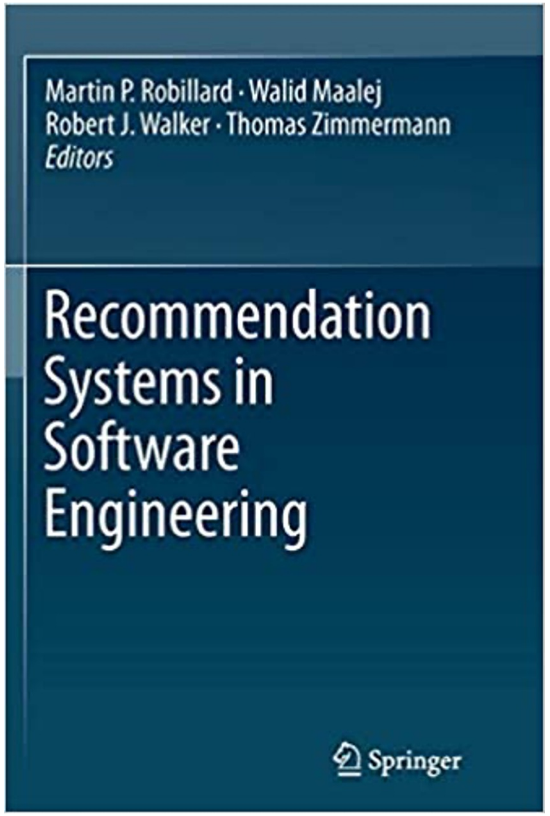
Recommender Systems

A Machine Learning algorithm walks into a bar.
The bartender asks, "What'll you have?"
The algorithm says, "What's everyone else having?"

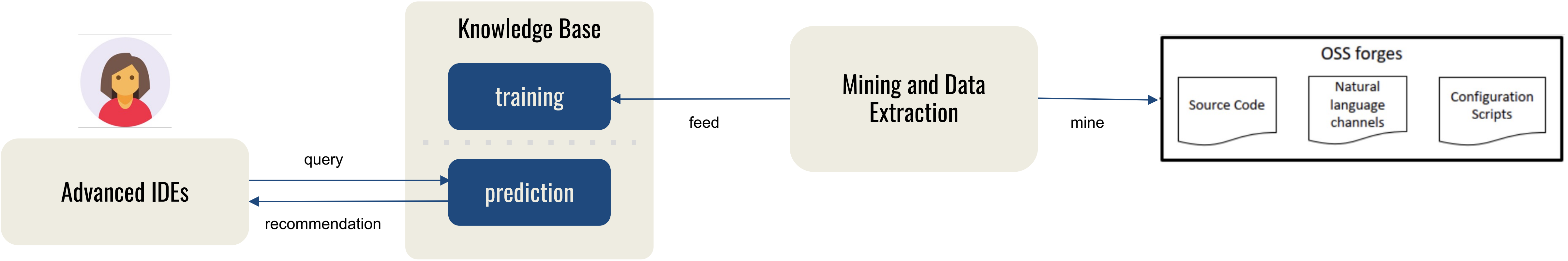
Recommender Systems in Software Engineering (RSSE)

"a software application that provides information items estimated to be valuable for a software engineering task in a given context."

- Martin Robillard, Walid Maalej, Robert Walker, Thomas Zimmermann, ***Recommendation Systems in Software Engineering***, Springer



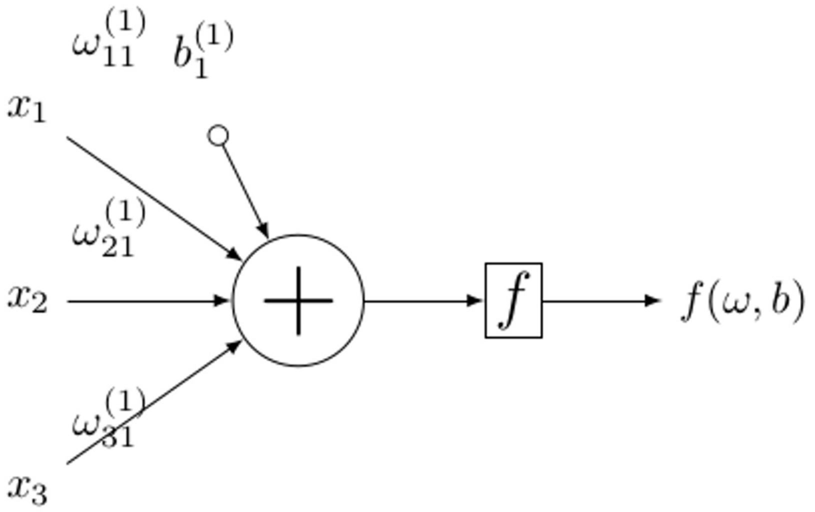
Intelligent RecSys Architecture



- Incorporating various recommendation and Machine Learning techniques
- Aiming to efficiently and effectively mine the existing open source software repositories

Machine Learning and Deep Learning

Neural Network: Perceptron

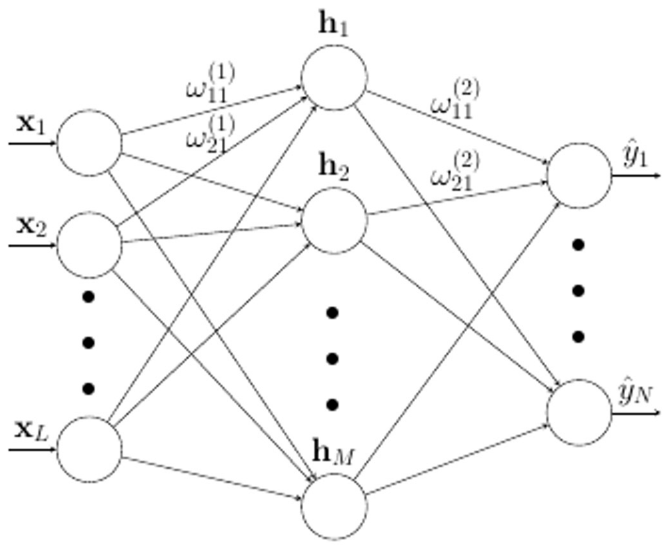


(a) A simple perceptron with 3 inputs

$$output = f(\sum_{i=1}^{3} \omega_{i1}^{(1)} \cdot x_i + b^{(1)})$$

- A decision making unit, taking into consideration different inputs

Feed-forward Neural Network



(b) A three-layer neural network

$$C(w,b) = \frac{1}{2n}\sum_x \|y(x) - (w.x + b)\|^2$$

$$\Delta y \approx \sum_j \frac{\partial y}{\partial w_j}\Delta w_j + \frac{\partial y}{\partial b}\Delta b$$

- Learning is actually the process of modifying the weights so that we can produce the desired outputs given the inputs
- The more input data you feed the network, the better accuracy you will obtain

Minimizing the cost function

- Find a set of weights that minimize the cost function between the predicted value and the real one using Stochastic Gradient Descent (SGD)

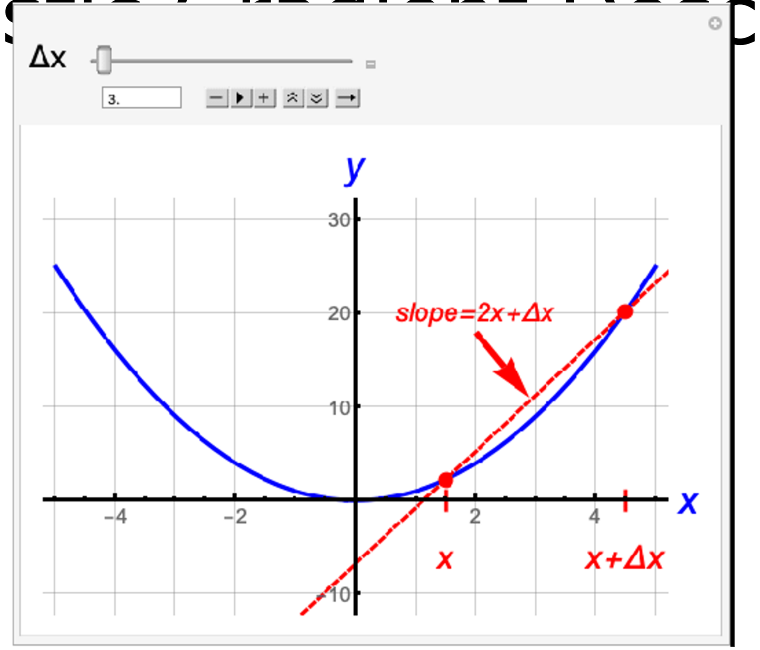
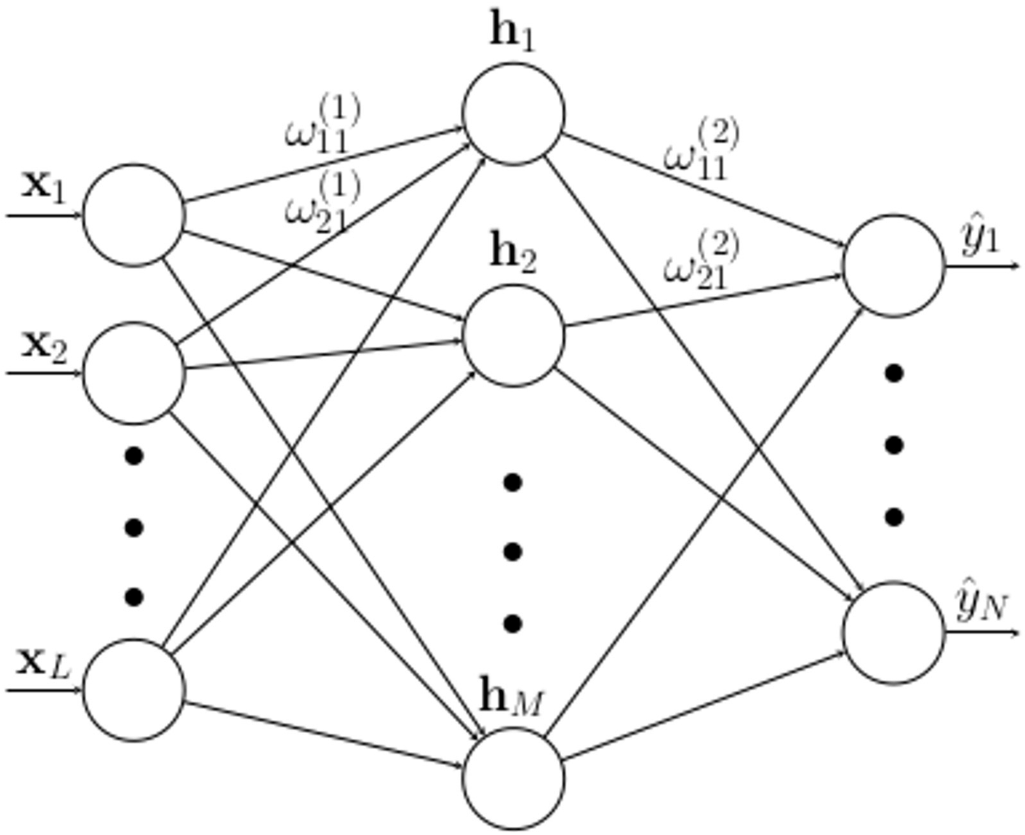
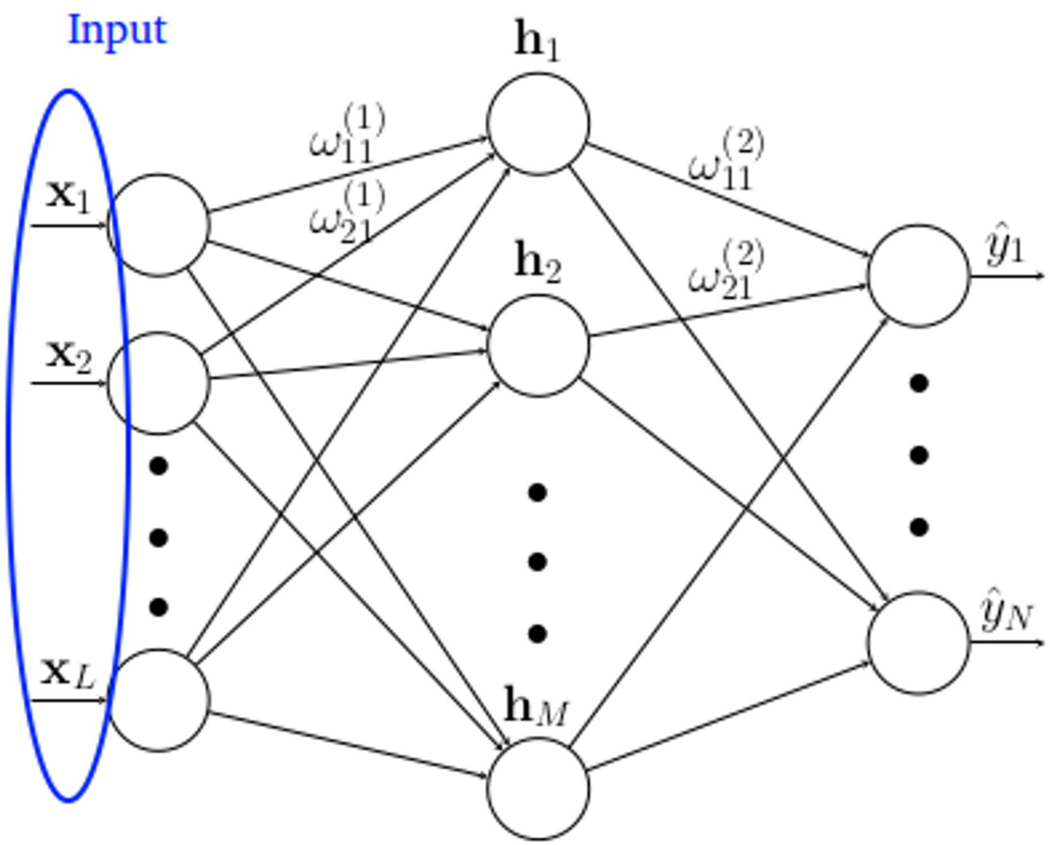


Image source: https://infinityisreallybig.com/2019/09/20/the-derivative-of-y-x2/

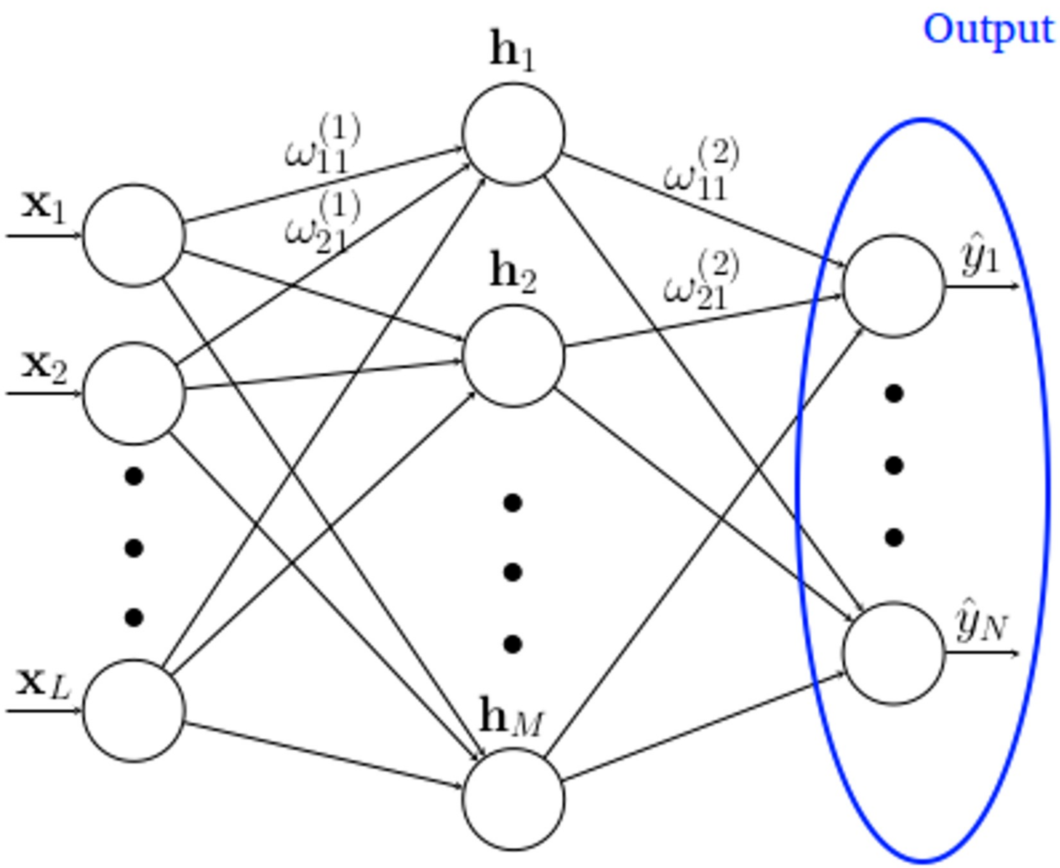
A real neural network



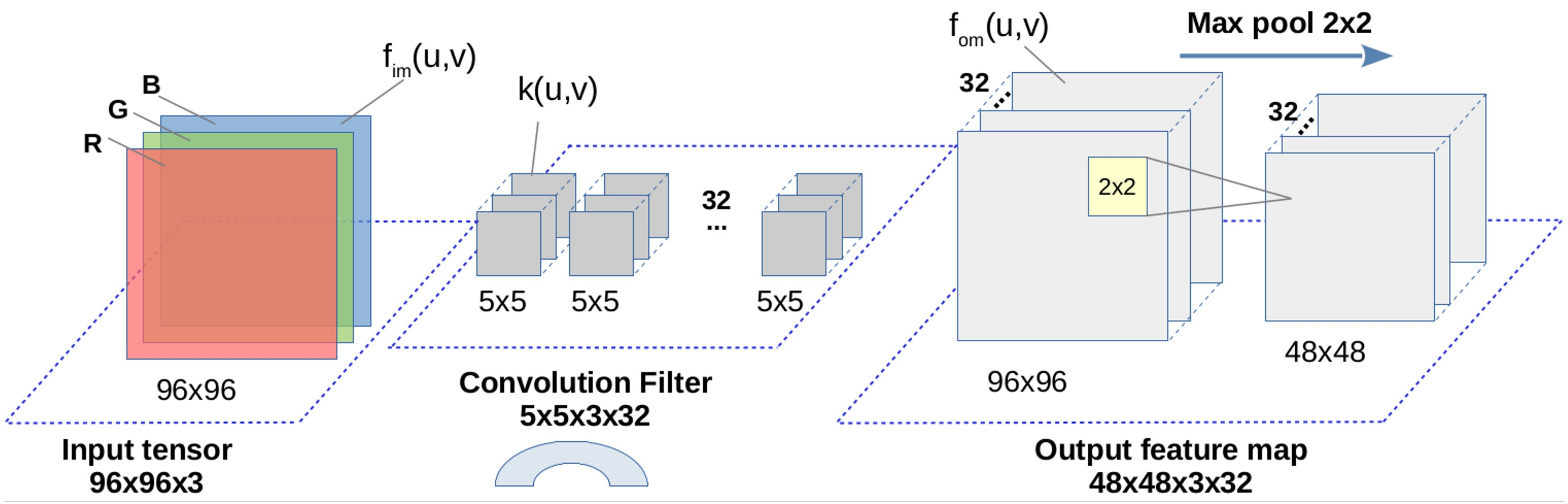
A real neural network



A real neural network

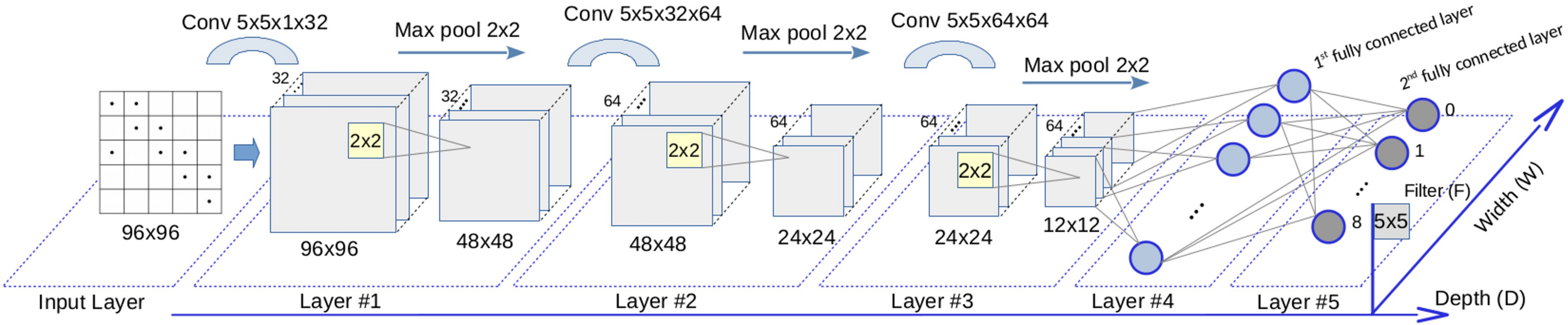


Convolutional neural networks



Phuong T. Nguyen, Davide Di Ruscio, Juri Di Rocco, Alfonso Pierantonio, Ludovico Iovino, "*Convolutional neural networks for enhanced classification mechanisms of metamodels*," Journal of Systems and Software, 2020, DOI: [10.1016/j.jss.2020.110860](https://doi.org/10.1016/j.jss.2020.110860).

A CNN



Phuong T. Nguyen, Davide Di Ruscio, Juri Di Rocco, Alfonso Pierantonio, Ludovico Iovino, "*Convolutional neural networks for enhanced classification mechanisms of metamodels*," Journal of Systems and Software, 2020, DOI: [10.1016/j.jss.2020.110860](https://doi.org/10.1016/j.jss.2020.110860).

Machine Learning frameworks

Keras

TensorFlow

PyTorch

colab

- ML frameworks provide a convenient way to program and run ML code

Train a neural network

Train a dog

- Input feature: commands: "sit down", "shake hand"
- Label: you press him, or grab his hand to instruct the corresponding action
- Deployment: you speak out the command, and the dog does exactly what he was taught

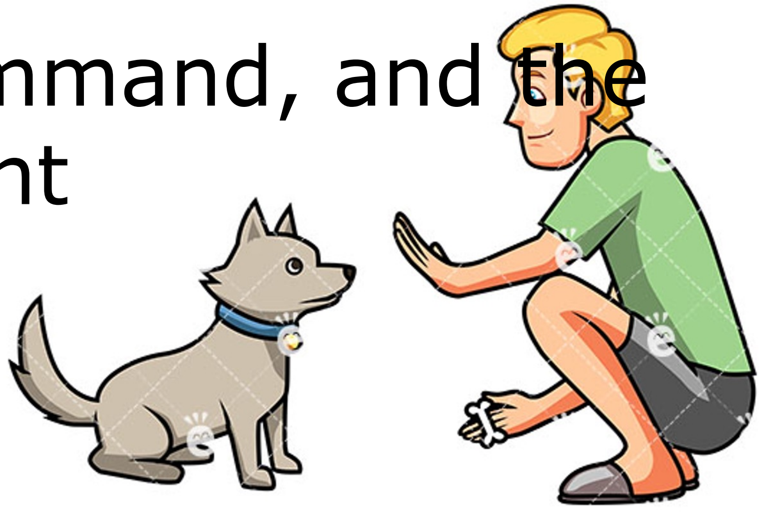


Image source: https://friendlystock.com/product/man-training-his-dog-to-stay-giving-treat/

Train a neural network

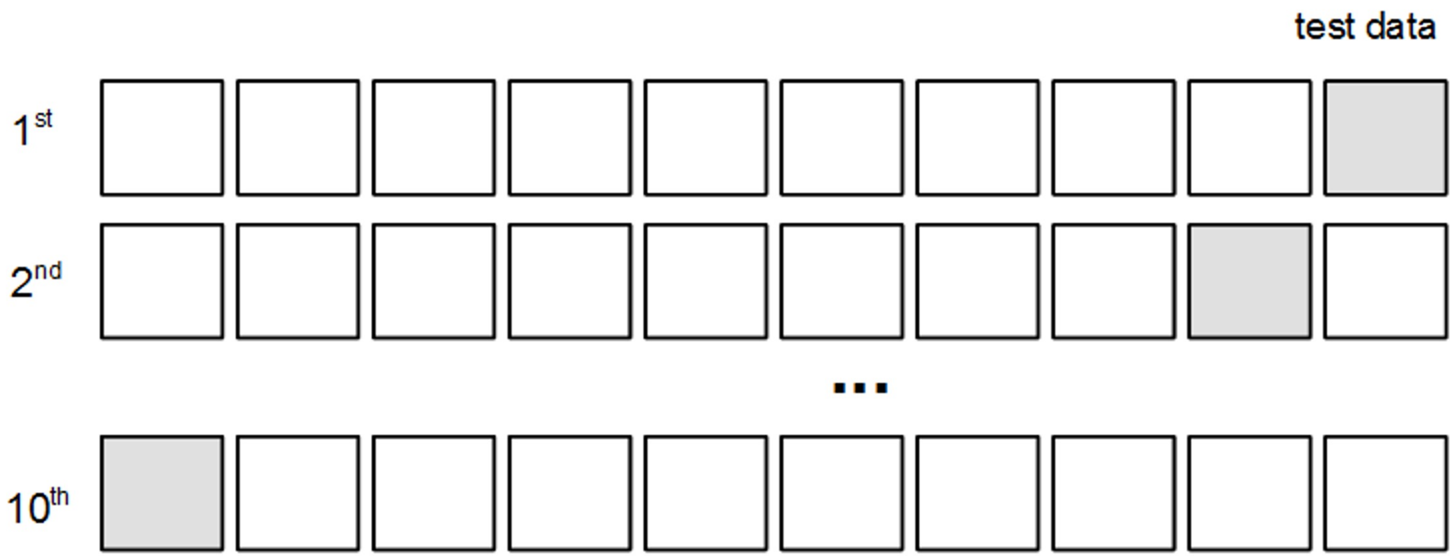
- Learning is actually the process of modifying the weights so that we can produce the desired outputs given the inputs
- The more (good) input data you feed the network, the better accuracy you will obtain

How to test a neural network

K-fold cross validation

- Given a dataset with labels, you split into k equal parts
- k-1 parts are used for training, 1 part is used for testing
- For the testing part, you remove the labels (used as ground-truth data)
- The labels returned by the neural network are compared with the ground-truth labels

Ten-fold cross validation



- The dataset is split into 10 equal parts
- Nine parts are training data, 1 part is testing data
- The evaluation is performed in 10 rounds

More information: https://www.kdnuggets.com/2018/01/training-test-sets-cross-validation.html

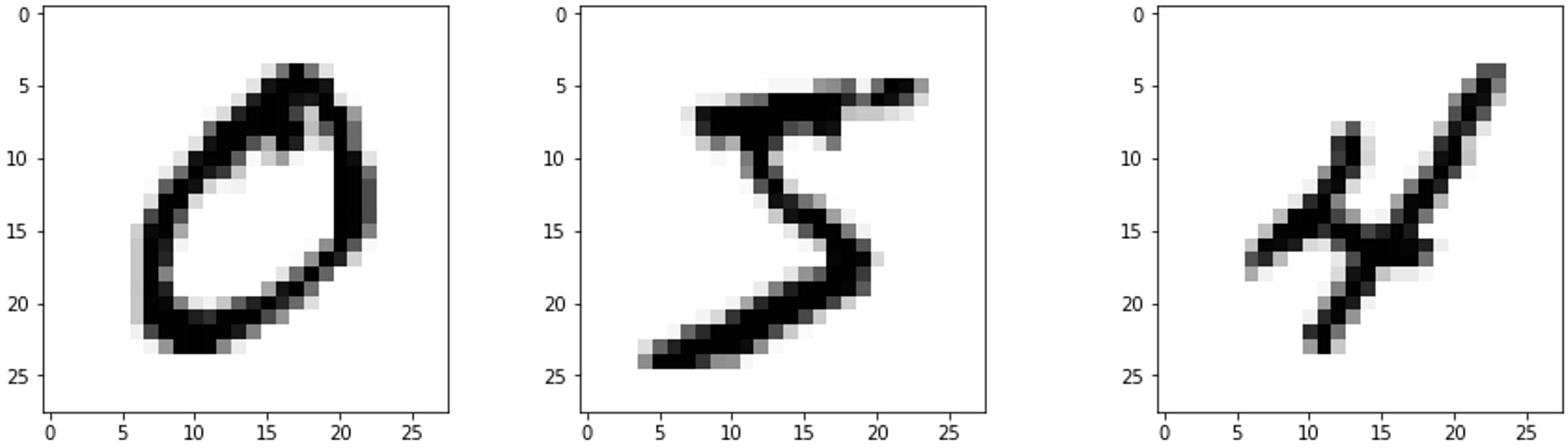
Spam Detection

- Training: a set of emails with two labels: Spam or Not Spam
- Deployment: an email is fed into the network, which returns the desired label of the email

Recognition of handwritten numbers

- Training: a set of images with 10 labels: from 0 to 9
- Deployment: an image containing a handwritten number is fed into the network, which computes and returns the predicted number

Handwritten numbers



■ Image source: https://www.python-course.eu/neural_network_mnist.php

■ MNIST: a dataset with 60,000 training and 10,000 testing images

More information: https://www.python-course.eu/neural_network_mnist.php

Recognition of handwritten numbers

- Each image is a 28 x 28 pixel matrix
- Each image is represented in a row, the first column is the label, and the remaining 784 columns contain grayscale level of one pixel (from 0-255)

Notable applications in Software Engineering

Notable applications of RSSE

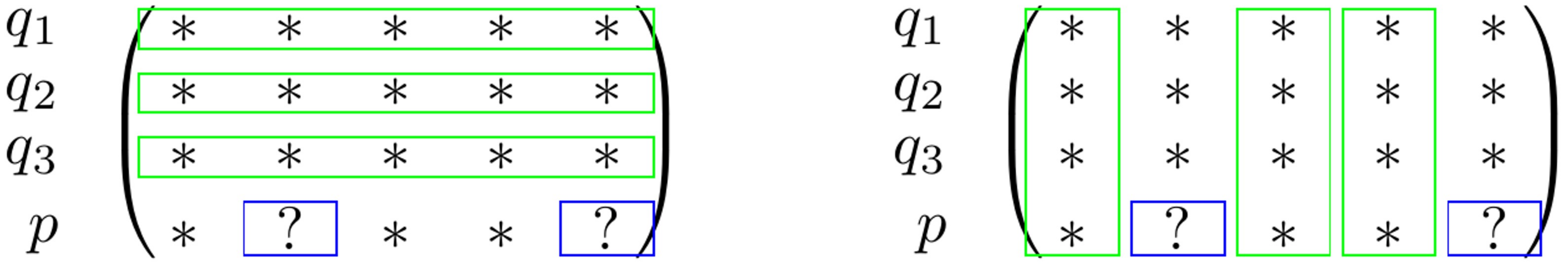
- Recommendation of third-party libraries (TPLs)
- Recommendation of API function calls, code snippets
- Recommendation of new versions of libraries (upgrading)

CrossRec: Recommending TPLs

$$
\begin{array}{c|ccccc}
 & lib_1 & lib_2 & lib_3 & lib_4 & lib_5 \\
\hline
p_1 & 1 & 1 & 0 & 0 & 0 \\
p_2 & 1 & 0 & 1 & 0 & 0 \\
p_3 & 1 & 0 & 1 & 1 & 1 \\
p_4 & 1 & 1 & 0 & 1 & 1
\end{array}
$$

- Representing the project-library relationships using a user-item ratings matrix
- Predict the inclusion of additional libraries

Predict the inclusion of TPLs



(a) User-based CF (b) Item-based CF

- Missing "ratings" can be predicted using collaborative-filtering techniques
- The row-wise and column-wise relationships are exploited to compute missing ratings

The Journal of Systems and Software 161 (2020) 110460

Contents lists available at ScienceDirect

The Journal of Systems and Software

journal homepage: www.elsevier.com/locate/jss

ELSEVIER

CrossRec: Supporting software developers by recommending third-party libraries

Phuong T. Nguyen[a], Juri Di Rocco[a], Davide Di Ruscio[a,*], Massimiliano Di Penta[b]

[a] Università degli studi dell'Aquila, L'Aquila 67100, Italy
[b] Università degli Studi del Sannio, Benevento 82100, Italy

Check for updates

JSS's Best Papers for 2020

Diamond Best Paper Award

https://bit.ly/3bZi5cx

DOI: 10.1016/j.jss.2019.110460

FOCUS: Recommending APIs

UNIVERSITÀ DEGLI STUDI DELL'AQUILA
DISIM Dipartimento di Ingegneria e Scienze dell'Informazione e Matematica
VIASM
Bộ Giáo dục và Đào tạo
VIỆN NGHIÊN CỨU CAO CẤP VỀ TOÁN

```java
public List<Boekrekening> findBoekrekeningen() {
   CriteriaBuilder cb = entityManager.getCriteriaBuilder();
   CriteriaQuery<Boekrekening> criteriaQueryBoekrekening =
   cb.createQuery(Boekrekening.class);

                                   ??
}
```

(a) Initial version

```java
public List<Boekrekening> findBoekrekeningen() {
   CriteriaBuilder cb = entityManager.getCriteriaBuilder();
   CriteriaQuery<Boekrekening> criteriaQueryBoekrekening =
   cb.createQuery(Boekrekening.class);

   Root<BoekrekeningPO> boekrekeningFrom =
   criteriaQueryBoekrekening.from(BoekrekeningPO.class);
   criteriaQueryBoekrekening.select(boekrekeningFrom);
   criteriaQueryBoekrekening.
     orderBy(cb.asc(boekrekeningFrom.get(BoekrekeningPO_.rekeningnr)));
   return entityManager.createQuery(criteriaQueryBoekrekening).getResultList();
}
```

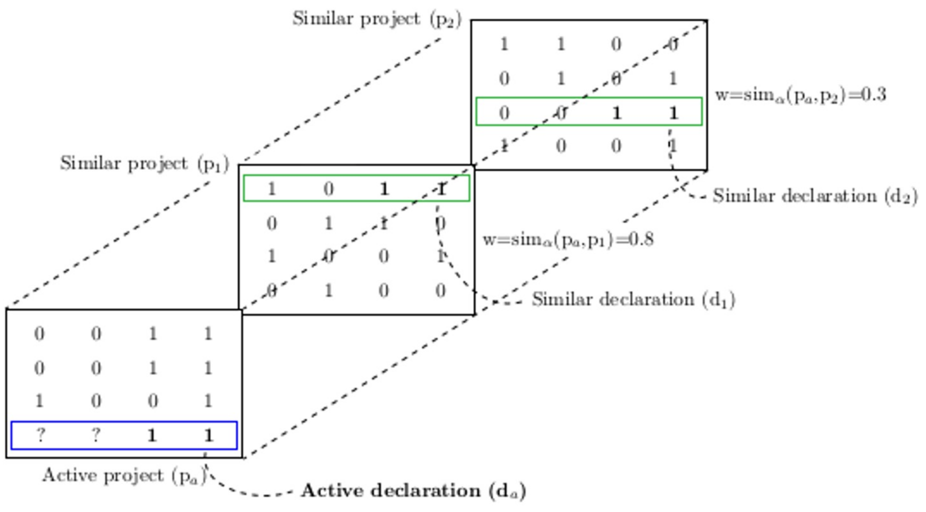
(b) Final version

```java
public List<QuestionsStaged> findByIdentifier(String identifier) {
  log.fine("getting Session instance by identifier: " + identifier);
  try {
     CriteriaBuilder cb = entityManager.getCriteriaBuilder();
     CriteriaQuery<QuestionsStaged> criteria = cb.createQuery(QuestionsStaged.class);
     Root<QuestionsStaged> qs = criteria.from(QuestionsStaged.class);
     criteria.select(qs).where(cb.equal(qs.get("identifier"), identifier));
     log.fine("get identifier successful");
     return entityManager.createQuery(criteria).getResultList();

  } catch (RuntimeException re) {
     log.severe("get identifier failed" + re);
     throw re;
  }
}
```

Recommended source code

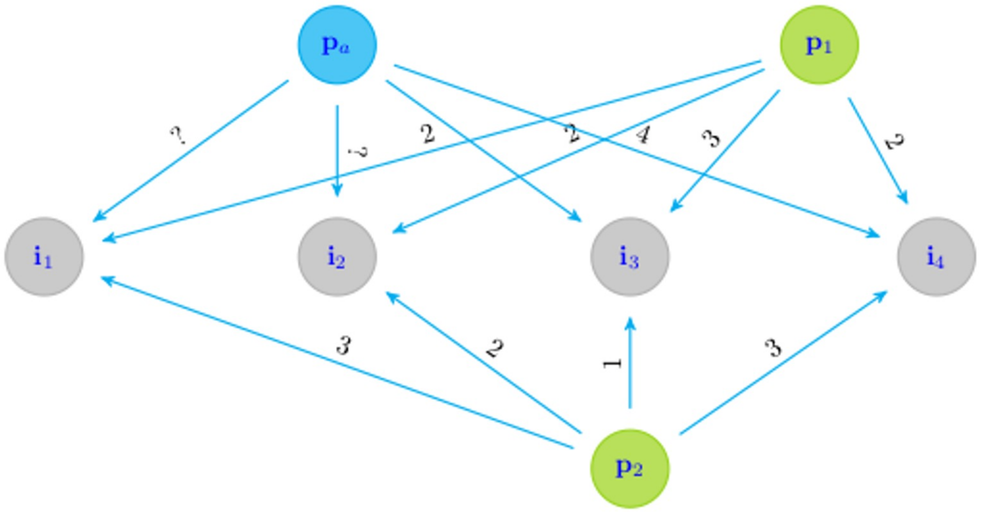
Context-aware collaborative-filtering



(a) 3D context-based rating matrix

- Representing the project-declaration-invocation relationships using a 3D user-item ratings matrix
- Predicting the inclusion of additional APIs

FOCUS: Mining APIs



- Exploiting graph to compute similarities among projects
- Using context-aware collaborative filtering techniques to mine APIs

FOCUS: Mining APIs

UNIVERSITÀ DEGLI STUDI DELL'AQUILA

DISIM Dipartimento di Ingegneria e Scienze dell'Informazione e Matematica

VIASM Bộ Giáo dục và Đào tạo
VIỆN NGHIÊN CỨU CAO CẤP VỀ TOÁN

2019 IEEE/ACM 41st International Conference on Software Engineering (ICSE)

Artifacts Available acm

Artifacts Evaluated acm Reusable

FOCUS: A Recommender System for Mining API Function Calls and Usage Patterns

Phuong T. Nguyen, Juri Di Rocco, Davide Di Ruscio
Università degli Studi dell'Aquila
L'Aquila, Italy
{firstname.lastname}@univaq.it

Lina Ochoa, Thomas Degueule
Centrum Wiskunde & Informatica
Amsterdam, Netherlands
{firstname.lastname}@cwi.nl

Massimiliano Di Penta
Università degli Studi del Sannio
Benevento, Italy
dipenta@unisannio.it

DOI: [10.1109/ICSE.2019.00109](https://doi.org/10.1109/ICSE.2019.00109)

DeepLib: Upgrading TPLs

current versions (input features)

Client	lib_1	lib_2	lib_3	lib_4
2.0.0	1.2.15	1.6.1	0	1.4
2.0.2	1.2.17	1.6.1	0	1.4
2.3.0	1.2.17	1.7.5	4.2.5	1.4
2.4.1	0	1.7.5	4.2.5	1.4
2.5.1	0	0	4.2.5	1.4
2.6.0	1.2.17	0	4.3.1	0

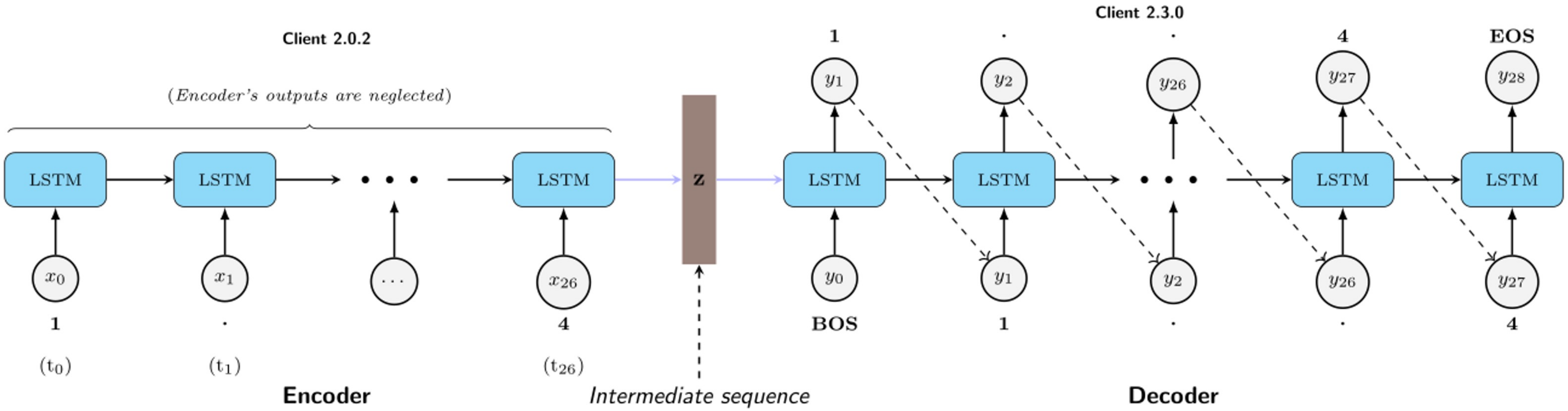
future versions (labels)

Client	lib_1	lib_2	lib_3	lib_4
2.0.2	1.2.17	1.6.1	0	1.4
2.3.0	1.2.17	1.7.5	4.2.5	1.4
2.4.1	0	1.7.5	4.2.5	1.4
2.5.1	0	0	4.2.5	1.4
2.6.0	1.2.17	0	4.3.1	0
*	?	?	?	?

(a) Migration matrices for a set of libraries

- Using matrices to represent the TPLs upgrading

DeepLib: Upgrading TPLs



- Library upgrading is formulated as a machine translation task
- Encoder-Decoder neural networks can be used to

DeepLib: Upgrading TPLs

Expert Systems With Applications 202 (2022) 117267

Contents lists available at ScienceDirect

Expert Systems With Applications

journal homepage: www.elsevier.com/locate/eswa

DeepLib: Machine translation techniques to recommend upgrades for third-party libraries

Phuong T. Nguyen, Juri Di Rocco, Riccardo Rubei, Claudio Di Sipio, Davide Di Ruscio *

Department of Information Engineering, Computer Science and Mathematics, Università degli studi dell'Aquila, 67100 L'Aquila, Italy

Ongoing research issues

Ongoing research issues

- Dealing with time-series data in Software Engineering with deep learning
  - Recommending third-party libraries migration for Android apps
  - Predicting code insertion

Ongoing research issues (2)

- Adversarial Machine Learning
  - Manipulating training data to perturb recommendations
  - Understanding attacks to recommender systems
  - Finding decent countermeasures

Ongoing research

- Federated learning
  - Training with heterogeneous datasets
  - Preserving privacy for the participating platforms

Adversarial Machine Learning

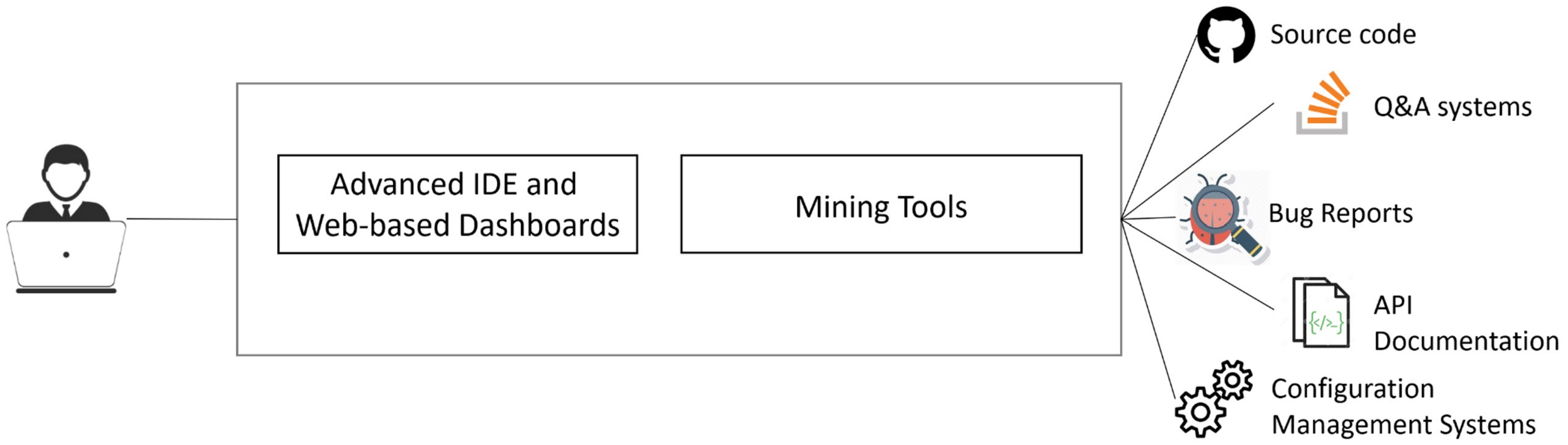
- Adversarial Machine Learning (AML) is a field of study that focuses on security issues in ML systems and recommender systems
- The aim of adversarial attacks is to manipulate target items, thus creating either a negative or positive impact on the final recommendations

AML in Image classification

Image source:
https://bit.ly/2RXhMsU

- Non-random noise added to an image can fool DL algorithms

Adversarial Machine Learning in RSSE



- Recommender systems for software engineering (RSSE) work with data from OSS platforms
- They are susceptible to crafted data

Classification of attacks

- **Evasion attacks** attempt to avoid being detected by hiding malicious contents, which then will be classified as legitimate by ML models
- **Poisoning attacks** spoil an ML model by falsifying the input data, aiming to perturb the final outcomes

Classification of attacks

- **Evasion attacks** attempt to avoid being detected by hiding malicious contents, which then will be classified as legitimate by ML models
- **Poisoning attacks** spoil an ML model by falsifying the input data, aiming to perturb the final outcomes

Poisoning attacks

- **Push attacks** favor the targeted items, to increase the possibility of being recommended
- In contrast, **nuke attacks** try to downgrade/defame the targeted items, forcing them to disappear from the recommendation list

Poisoning attacks

- **Push attacks** favor the targeted items, to increase the possibility of being recommended
- In contrast, **nuke attacks** try to downgrade/defame the targeted items, forcing them to disappear from the recommendation list

Risk of being exploited for malicious purposes

Table 1: Notable RSSE for mining libraries and APIs.

	System	Venue	Year	Data source
Library rec.	LibRec [33]	WCRE	2013	GitHub
	LibCUP [28]	JSS	2017	GitHub
	LibD [15]	ICSE	2017	Android markets
	LibFinder [24]	IST	2018	GitHub
	CrossRec [21]	JSS	2020	GitHub
	LibSeek [12]	TSE	2020	Google Play, GitHub, MVN
API rec.	MAPO [38]	ECOOP	2009	SourceForge
	UP-Miner [35]	MSR	2013	Microsoft Codebase
	DeepAPI [10]	ESEC/FSE	2016	GitHub
	PAM [8]	ESEC/FSE	2016	GitHub
	FINE-GRAPE [29]	EMSE	2017	GitHub
	FOCUS [22, 23]	ICSE	2019	GitHub, MVN

- The systems leverage open sources, e.g., GitHub or Android markets, for training
- They mine libraries using similarity-based measures, either a similarity function, or a clustering technique

Proof of concept

- Method: Manipulate the input data used to feed two third-party recommender systems
- Object: A malicious library named lib*
- Aim: Check if these systems provide lib* to software developers

Attacks to Library RecSys

$$\overbrace{\boxed{lib_1}\boxed{lib_2}\boxed{\ \dots\ }\boxed{\ \dots\ }\boxed{\ \dots\ }\boxed{lib_F}}^{\text{Filler libraries } (\mathbf{F})}\boxed{lib^*}$$

- "*Which libraries should be chosen as fillers, so that the fake project will be incorporated into the recommendation?*"
- We boost the popularity of the malicious library by embedding it to several projects

Hit ratio on the recommendation of LibRec

Table 2: Hit ratio @N for LibRec.

		HR@10					HR@15				
		γ=3	γ=4	γ=6	γ=8	γ=10	γ=3	γ=4	γ=6	γ=8	γ=10
α = 5%	**k=5**	–	0.154	0.177	0.242	0.219	–	0.154	0.189	0.252	0.237
	k=10	–	0.198	0.273	0.428	0.410	–	0.198	0.317	0.455	0.442
	k=15	–	0.199	0.307	0.541	0.546	–	0.199	0.368	0.580	0.580
	k=20	–	0.177	0.316	0.608	0.637	–	0.177	0.388	0.658	0.670
α = 10%	**k=5**	–	0.247	0.240	0.242	0.210	–	0.247	0.256	0.251	0.237
	k=10	–	0.380	0.418	0.428	0.320	–	0.380	0.449	0.455	0.442
	k=15	–	0.439	0.505	0.541	0.390	–	0.439	0.554	0.580	0.580
	k=20	–	0.443	0.547	0.608	0.431	–	0.443	0.605	0.657	**0.670**

- Hit ratio is always larger than 0, implying that LibRec recommends the malicious libraries to developers

Hit ratio on the recommendation of LibRec

Table 2: Hit ratio @N for LibRec.

		HR@10					HR@15				
		γ=3	γ=4	γ=6	γ=8	γ=10	γ=3	γ=4	γ=6	γ=8	γ=10
α = 5%	**k=5**	—	0.154	0.177	0.242	0.219	—	0.154	0.189	0.252	0.237
	k=10	—	0.198	0.273	0.428	0.410	—	0.198	0.317	0.455	0.442
	k=15	—	0.199	0.307	0.541	0.546	—	0.199	0.368	0.580	0.580
	k=20	—	0.177	0.316	0.608	0.637	—	0.177	0.388	0.658	0.670
α = 10%	**k=5**	—	0.247	0.240	0.242	0.210	—	0.247	0.256	0.251	0.237
	k=10	—	0.380	0.418	0.428	0.320	—	0.380	0.449	0.455	0.442
	k=15	—	0.439	0.505	0.541	0.390	—	0.439	0.554	0.580	0.580
	k=20	—	0.443	0.547	0.608	0.431	—	0.443	0.605	0.657	**0.670**

- Hit ratio is always larger than 0, implying that LibRec recommends the malicious libraries to developers

Hit ratio on the recommendation of CrossRec

Bộ Giáo dục và Đào tạo
VIỆN NGHIÊN CỨU CAO CẤP VỀ TOÁN

Table 3: Hit ratio @N for CrossRec.

		HR@10					HR@15				
		γ=3	γ=4	γ=6	γ=8	γ=10	γ=3	γ=4	γ=6	γ=8	γ=10
α = 5%	**k=5**	0.159	0.158	0.145	0.113	0.103	0.178	0.177	0.163	0.138	0.120
	k=10	0.140	0.141	0.165	0.149	0.140	0.184	0.190	0.201	0.185	0.174
	k=15	0.154	0.163	0.188	0.198	0.189	0.200	0.227	0.235	0.250	0.229
	k=20	0.112	0.135	0.207	0.188	0.194	0.168	0.191	0.268	0.235	0.250
α = 10%	**k=5**	0.356	0.346	0.267	0.235	0.210	0.386	0.373	0.291	0.265	0.248
	k=10	0.440	0.430	0.348	0.347	0.320	0.496	0.478	0.388	0.393	0.357
	k=15	0.427	0.445	0.427	0.407	0.390	0.487	0.497	0.488	0.475	0.438
	k=20	0.455	0.424	0.457	0.454	0.431	0.515	0.525	**0.530**	0.514	0.486

- CrossRec is affected by the crafted input data, it recommends the fake library to developers by all the configurations

Hit ratio on the recommendation of CrossRec

Table 3: Hit ratio @N for CrossRec.

		HR@10					HR@15				
		γ=3	γ=4	γ=6	γ=8	γ=10	γ=3	γ=4	γ=6	γ=8	γ=10
α = 5%	**k=5**	0.159	0.158	0.145	0.113	0.103	0.178	0.177	0.163	0.138	0.120
	k=10	0.140	0.141	0.165	0.149	0.140	0.184	0.190	0.201	0.185	0.174
	k=15	0.154	0.163	0.188	0.198	0.189	0.200	0.227	0.235	0.250	0.229
	k=20	0.112	0.135	0.207	0.188	0.194	0.168	0.191	0.268	0.235	0.250
α = 10%	**k=5**	0.356	0.346	0.267	0.235	0.210	0.386	0.373	0.291	0.265	0.248
	k=10	0.440	0.430	0.348	0.347	0.320	0.496	0.478	0.388	0.393	0.357
	k=15	0.427	0.445	0.427	0.407	0.390	0.487	0.497	0.488	0.475	0.438
	k=20	0.455	0.424	0.457	0.454	0.431	0.515	0.525	**0.530**	0.514	0.486

- CrossRec is affected by the crafted input data, it recommends the fake library to developers by all the configurations

Future work: Countermeasures

- It is important to devise proper countermeasures to this type of attacks
- Fake pattern recognition with association rule mining strategies
- Profile classification: Supervised classifiers are trained to detect fake projects from generated data

Summary

Automated library recommendation

Publisher: IEEE   Cite This   PDF

Ferdian Thung ; David Lo ; Julia Lawall   All Authors

37 Paper Citations
808 Full Text Views

ELSEVIER
Journal of Systems and Software
Volume 161, March 2020, 110460

CrossRec: Supporting software developers by recommending third-party libraries

Phuong T. Nguyen [a], Juri Di Rocco [a], Davide Di Ruscio [a], Massimiliano Di Penta [b]

Show more
Add to Mendeley   Share   Cite

https://doi.org/10.1016/j.jss.2019.110460
Get rights and content

Table 1: Notable RSSE for mining libraries and APIs.

	System	Venue	Year	Data source
Library rec.	LibRec [33]	WCRE	2013	GitHub
	LibCUP [28]	JSS	2017	GitHub
	LibD [15]	ICSE	2017	Android markets
	LibFinder [24]	IST	2018	GitHub
	CrossRec [21]	JSS	2020	GitHub
	LibSeek [12]	TSE	2020	Google Play, GitHub, MVN
API rec.	MAPO [38]	ECOOP	2009	SourceForge
	UP-Miner [35]	MSR	2013	Microsoft Codebase
	DeepAPI [10]	ESEC/FSE	2016	GitHub
	PAM [8]	ESEC/FSE	2016	GitHub
	FINE-GRAPE [29]	EMSE	2017	GitHub
	FOCUS [22, 23]	ICSE	2019	GitHub, MVN

- Training data can be manipulated for malicious purposes
- Both the considered systems are prone to adversarial attacks
- Many more RSSE are supposed to be affected by AML
- There is an urgent need for suitable countermeasures

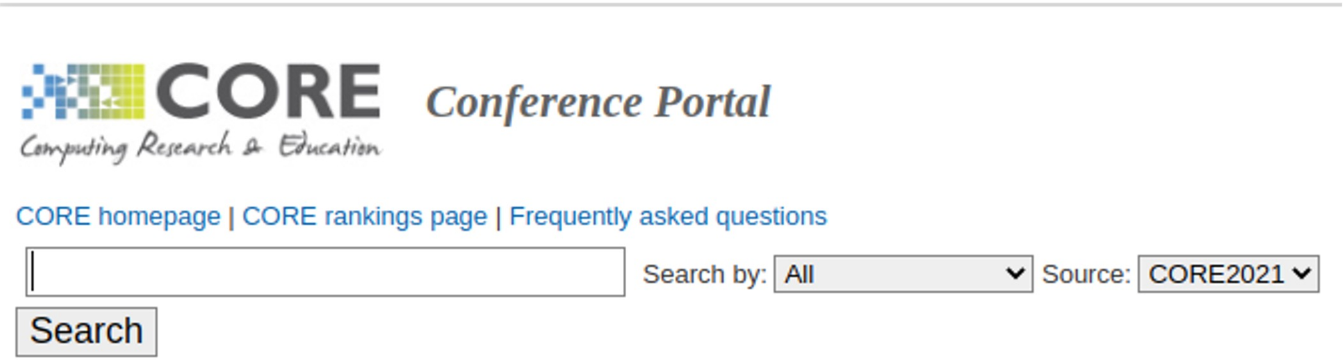
Venues for Software Engineering

	Publication	h5-index	h5-median
1.	ACM/IEEE International Conference on Software Engineering	76	113
2.	IEEE Transactions on Software Engineering	66	108
3.	Journal of Systems and Software	61	102
4.	Information and Software Technology	59	83
5.	ACM SIGSOFT International Symposium on Foundations of Software Engineering	57	93
6.	Empirical Software Engineering	56	85
7.	Proceedings of the ACM on Programming Languages	55	71
8.	IEEE Software	47	79
9.	IEEE/ACM International Conference on Automated Software Engineering (ASE)	47	77
10.	ACM SIGPLAN Conference on Programming Language Design and Implementation (PLDI)	47	66
11.	Mining Software Repositories	45	68
12.	Symposium on Operating Systems Principles	42	102
13.	International Conference on Software Analysis, Evolution, and Reengineering (SANER)	42	68
14.	Software & Systems Modeling	40	64
15.	International Symposium on Software Testing and Analysis	39	72
16.	IEEE International Conference on Software Maintenance and Evolution	36	52

Link: https://bit.ly/3SCfauR

Check ranking

Conferences



Link: http://portal.core.edu.au/conf-ranks/

Journals

Link: https://www.scimagojr.com/

Summary

- ❖ The proliferation of deep learning (DL) algorithms paves the way for a plethora of applications
- ❖ Machine learning and deep learning have been widely used in Software Engineering
- ❖ Open research issues

▪ Machine Learning algorithms attempt to mimic the ability of humans to learn from data ▪ Learning: the process of finding a set of weights and biases to produce the output, given an input	▪ RecSys have been widely used to mine OSS repositories ▪ There are ongoing research topics pertinent to RecSys, e.g., adversarial learning, federated learning
▪ Federated Learning: Preserving users' privacy, while maintaining the robustness of DL models by distributing the computation to several platforms	▪ Adversarial Machine Learning: The possibility to manipulate training data, causing malfunctions to systems based on deep learning

Thank you for your attention!

Questions and Answers