CY. center for algorithmic transparency

Document Title	**Process focused solutions**
Project Title and acronym	Cyprus Center for Algorithmic Transparency (CyCAT)
H2020-WIDESPREAD-05-2017-Twinning	Grant Agreement number: 810105 — CyCAT
Deliverable No.	D4.4
Work package No.	WP4
Work package title	Promoting Algorithmic Transparency
Authors (Name and Partner Institution)	Tsvi Kuflik (UH)
Contributors (Name and Partner Institution)	Alan Hartman, Avital Shulner-Tal, Veronika Bogina (UH)
Reviewers	Fausto Giunchiglia (UNITN) Frank Hopfgartner (USFD)
Status (D: draft; RD: revised draft; F: final)	F
File Name	D4.4_Paper-Based_Developer_Side_Solutions_M18
Date	31 March 2020

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No. 810105.

Draft Versions - History of Document				
Version	**Date**	**Authors / contributors**	**e-mail address**	**Notes / changes**
v1.0		Tsvi Kuflik	tsvikak@is.haifa.ac.il	Initial version
v2.0	19/3/20	Tsvi Kuflik	tsvikak@is.haifa.ac.il	Final version

Abstract	
D4.4-Process-focused solutions documents various process-focused "solutions on paper." Solutions generated will concern the role of processes (e.g., the choice of the training data used, the choice of the interface design) as well as the developer him-/herself (e.g., the developer's own background and set of beliefs, the assumptions he or she makes about the user) and the user-system feedback loop, in promoting algorithmic transparency. Based on the solutions developed, the OUC team shall create a tailored Developers' seminar for computer science students in Cyprus (T5.2 / D5.3).	
Keyword(s):	Algorithmic bias, Developers training, Developers' training guide

Contents

1. Executive Summary

D4.4 uses the results of the analysis of D4.1 and presents a variety of "paper-based-solutions" for the development process of an algorithmic system as well as its ongoing usage, focusing on the process-related aspects where different stakeholders (developer, owner, observer, educator) may be involved. The aim is to make stakeholders aware of the risks and suggesting ways of training them to prevent/discover discrimination risks and for providing explanations about the reasoning process during the process of developing and while using an algorithmic system. These aspects should be integrated into the regular system/software development and usage processes, starting from system/software requirements as another set of non-functional requirements, throughout the development and during ongoing system operation. The document presents a wide range of training activities aimed at training developers to be aware of FAT related aspects as part of their basic and advanced training and points to existing examples/cases. It is worth noting that this area attracts a lot of research and public attention and new information is becoming available by the day, hence with respect to examples and cases, the document provides only a snapshot of the current state of the art. This deliverable is tightly linked to D4.3, as both are complementary and to a certain extent overlapping in nature. When considering a course/training about FAT, there is a need to consider both data aspect (D4.3) and process aspect (D4.4). In addition, this deliverable serves as the basis for T5.2 / D5.3, in which the OUC team shall develop a Developers' Seminar in Algorithmic Transparency to be given to computer science students in Cyprus.

The rest of the document is structured as follows: Section 2 provide a general description of methods/techniques that can be applied for training developers, Section 3 provides a detailed description of each and every approach described in section 2 + a suggested list of concepts to be introduced, section 4 provides a list of tools and datasets that can be used for training, Section 5 provides a list of courses/tutorials/training material given/available on the web, section 6 provides a few additional resources, Section 7 summarizes the document, Section 8 provides a list of references and the appendix includes an example for a syllabus for a graduate seminar.

2. Training methods/techniques

Concluding from issues raised in other deliverables of WP 4 we argue that there is a need to educate parties involved in the development of algorithmic systems (from now on referred as developers for simplicity) about the potential risks of bias and discrimination in such systems. Hence, the issue of FAT needs to be an integral part of the education of anyone involved in the development of an algorithmic system. There is a need to consider not only educating new developers, but also supporting organizations in training their current workers. Considering current curricula of software/system development/management, we found out that there is a diverse set of ways for training these parties. We argue that the most important aspect in ensuring the fairness and transparency of an algorithmic system is to ensure that the system developers and operators are aware of the risks of discrimination and take measures to avoid it.

Educating can be done by training developers to be aware of the risk of biases, to ensure that fairness is an integral part of the system's requirements and to ensure throughout the development that no biases are unintentionally introduced whether as part of the design and implementation of the algorithms or as part of the training data or system configuration.

Still, while there is a need to train stakeholders about FAT, in general, it should be noted that algorithms by themselves are not biased, as described in D 4.3. Bias may be introduced by developers that develop the algorithms and/or by users that train and configure the algorithms or by the input used (if the algorithm improves its performance over time). Hence FAT-related requirements need to be part of the requirements engineering process that will lead to FAT-related tests as part of V&V process, integrated into system testing.

Based on our analysis of best practices of IT education, we suggest that new developers' training (students during their studies) may take place in one or more of the following forms:

2.1. It may be integrated into relevant classes of software developers in higher education (usually taking about 2-4 hours' sessions)

- As part of Artificial Intelligence/Machine Learning courses
- As part of Information Retrieval courses
- As part of Human-Computer Interaction courses
- As part of Software Engineering courses.
- As part of any other advanced courses like User Modeling or Recommender Systems (or any other course that involves developing and using machine learning techniques)

2.2. It may be integrated into relevant classes of other disciplines in higher education (Law, Social sciences, Business etc., usually taking about 2-4 hours' sessions)

- As part of any introductory Artificial Intelligence/Machine Learning Courses
- As part of more generic course about application of Information System in a specific domain
- As part of any other related course - where the use of ICT is discussed, e.g., focusing on responsible and ethical use of computer systems.

2.3. It may be introduced as dedicated, specific tutorials (usually taking about 3-6 hours' sessions)

- Introducing the challenges
- Hands-on/demonstration of the challenges
- Hands-on Experience of available tools

2.4. It may be introduced in a dedicated course (usually taking about 28-56 hours)

- In depth presentation of the challenges
- Case studies and demonstration
- Tools and techniques to discover/prevent discrimination

2.5. It may be presented as a graduate seminar (may take up to 60 hours)

- Covering the state of the art research
- In depth reviews of specific topics by the students

2.6. It may be presented as a one-day workshop + hackathon (about 30 additional hours)

- Covering the state of the art research
- Students/developers participation in a hackathon (building a transparent and fair algorithmic system)

When considering existing organizations, training their workers may be a bit challenging, as training interrupts regular development activities. In this case, the practical options are those described above in sections 2.1-2.3: An introduction to increase awareness or a more in depth tutorial (depending upon the selected option - whether a one-two hours talk or a half/full day tutorial). A more demanding option may be a concentrated course that may take about a week (equivalent to 2.4).

3. Detailed description of training sessions

When considering the "process", we may refer to two aspects:

1. The process of an algorithmic system, where we may consider the process and the outcome with respect to black box and white box (e.g. applying discrimination discovery techniques)
2. The process of the development of an algorithmic system where stakeholders need to be aware of their own personal and cultural biases and their potential impact.

It is worth noting that WP 4.3 and 4.4 are complementing each other, as both cover the whole system - the data it uses and the process it implements.

3.1. A single session integrated into regular class (see 2.1 and 2.2)

A single session may only be used for raising awareness. It may serve both "process" related goals noted above, but especially may be important for non-developers - managers and other stakeholders of organizations that use algorithmic systems that need to worry about their fairness and may be held accountable for unfair reasoning.

It is worth noting that machine learning algorithms by themselves are not biased. Bias may be introduced by the developers/users in the way they configure the algorithm, select the training data, define reasoning rules.

Such a session may take between one to three hours of lecture integrated into a relevant course. It can be based on the integrated model presented in D 4.1 followed by several case studies that can be presented as a lecture focusing on FAT as part of a specific course at any discipline.

With respect to fairness testing, existing measurements and tools that can be applied need to be presented (see Table 1 for pointers to lists of tools and measurements) and as time permits, demonstrate one of them using a biased dataset.

In D 3.2, we summarise several case studies on algorithmic bias. An overview can be found at http://www.cycat.io/case-studies/. Below there are three examples that can be used. They are presented in an order that reflects an increased complexity from the trivial and easily understood to the more complex one.

An introductory relevant case study may be the case of race discrimination in AirBnB, described by Edelman et al. 2017, which is easily understood, even though it is caused by Human bias and not by any algorithm, but it nicely demonstrates the general problem where users' perception/biases get into play - Use it to trigger a discussion about human diversity and personal biases.

Then use the example of Google Photos tagging African-Americans as Gorillas when using a facial recognition software (Zhang 2015) to demonstrate the importance of carefully selecting the right training data, considering also the expected as well as possibly unexpected use of the system, ensuring its applicability to the users' population. While the problem is not in any algorithm, but in the training data, it is an opportunity to discuss the integration of FAT-related non-functional requirements engineering and then introduce appropriate tests into system testing, as part of regular V&V - to ensure the non-functional FAT requirements are met.

Finally, consider using the ProPublica report about the COMPASS system for recidivism prediction that was found to discriminate Afro Americans (Feller et al. 2016, Dieterich et al. 2016, Washington 2018). - This example may be used to present and discuss the risks and challenges of “black box” decision support systems. The same issue of FAT non-functional requirements and testing may be discussed again, emphasizing the risk to human rights when these aspects are missed/ignored.

Obviously, there are many additional examples in the literature. that can be used and the lecturer is invited to add more to emphasize specific points. Examples are provided in Section 5.

<u>Sum up</u>
Terms to be introduced during the discussion:

- Algorithmic system
- Algorithmic transparency
- Group fairness
- Individual fairness
- Discrimination discovery

3.2. A Tutorial (see 2.3)

A tutorial may take between half a day and a day and it is intended for stakeholders that need not only to be aware of the risks but also to take action in preventing and detecting discrimination in algorithmic systems, mainly regulators and system testers / evaluators.
Whoever takes this tutorial needs to have a basic knowledge of artificial intelligence/machine learning.
A tutorial may start with a short introductory session (a shortened version of section 3.1) and then, present the model described in D 4.1 in detail. First of all the abstract algorithmic model, depicted in Fig. 1 and described in D 4.1, will be presented.

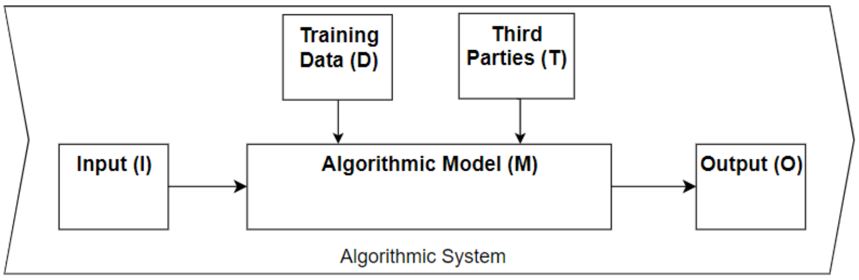

Figure 1. A schematic model of an Algorithmic System. It includes an algorithm (generic or one specially developed) that may be trained on training data selected by its developer or user (3rd party) and configured by its user (3rd party) given an input the system produces an output according to its training.

Using this model, it is possible to present possible sources of potential biases and risks that need to be considered:

- The algorithm may be biased intentionally or unintentionally by its developers (consider cultural and personal background and biases),
- The training data may be biased, the configuration may be biased and the input itself may be a cause of bias (in cases of reinforced learning)

Figure 2 provides a hierarchical presentation of problems and challenges that may be manifested in an algorithmic system.

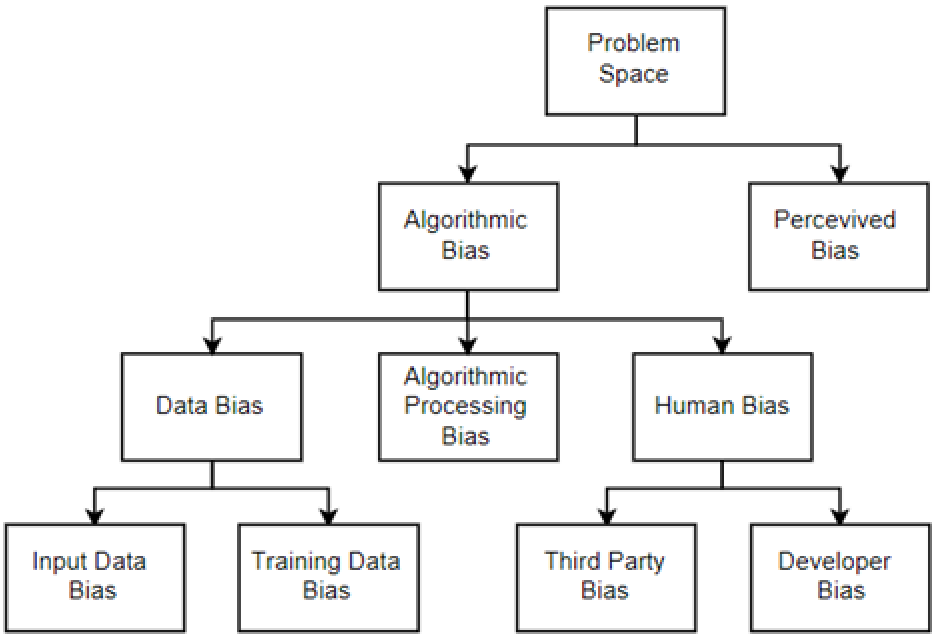

Figure 2: The problem space - system related biases (problems) classification.

Once the basic idea is clear, there is a need to present also the challenge of user perception - perceived bias: - The user perception about the difference between the input data and the system's results, compared with what the user thinks.

Now, Figure 3 may be used to summarise the discussion as it presents both the components of the algorithmic system, together with the allocation of biases/risks of discrimination to the individual sections.

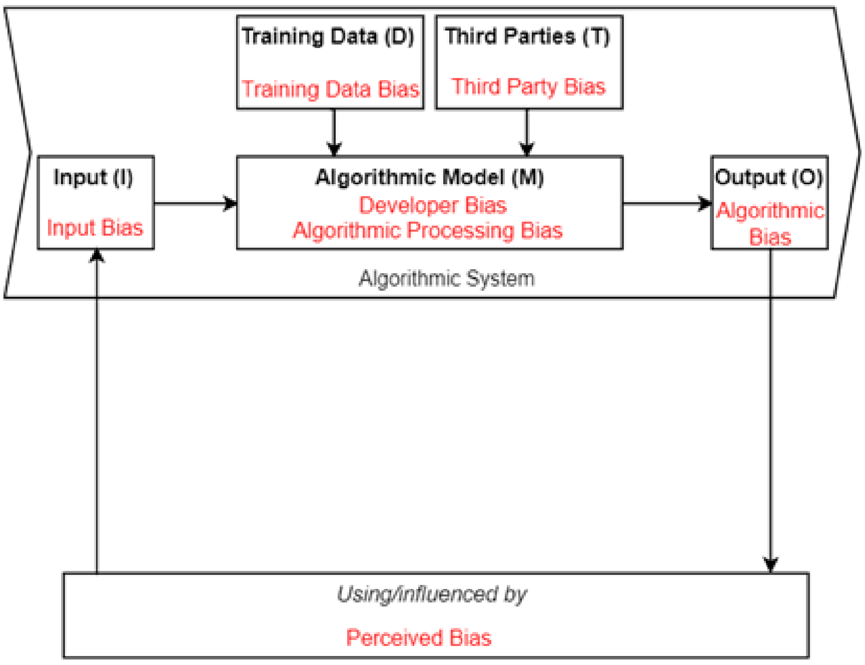

Figure 3. A schematic model of an Algorithmic System that includes also potential biases and the user perception aspect.

Once the challenges are presented, it is time to turn to possible solutions. The solutions need to be assigned to the challenges presented earlier, hence Figure 4 presents a hierarchical diagram of the solution space. Once the different solutions are discussed, Figure 5 may be used to integrate everything - the algorithmic system, the challenges and the possible solutions - as a complete overall framework.

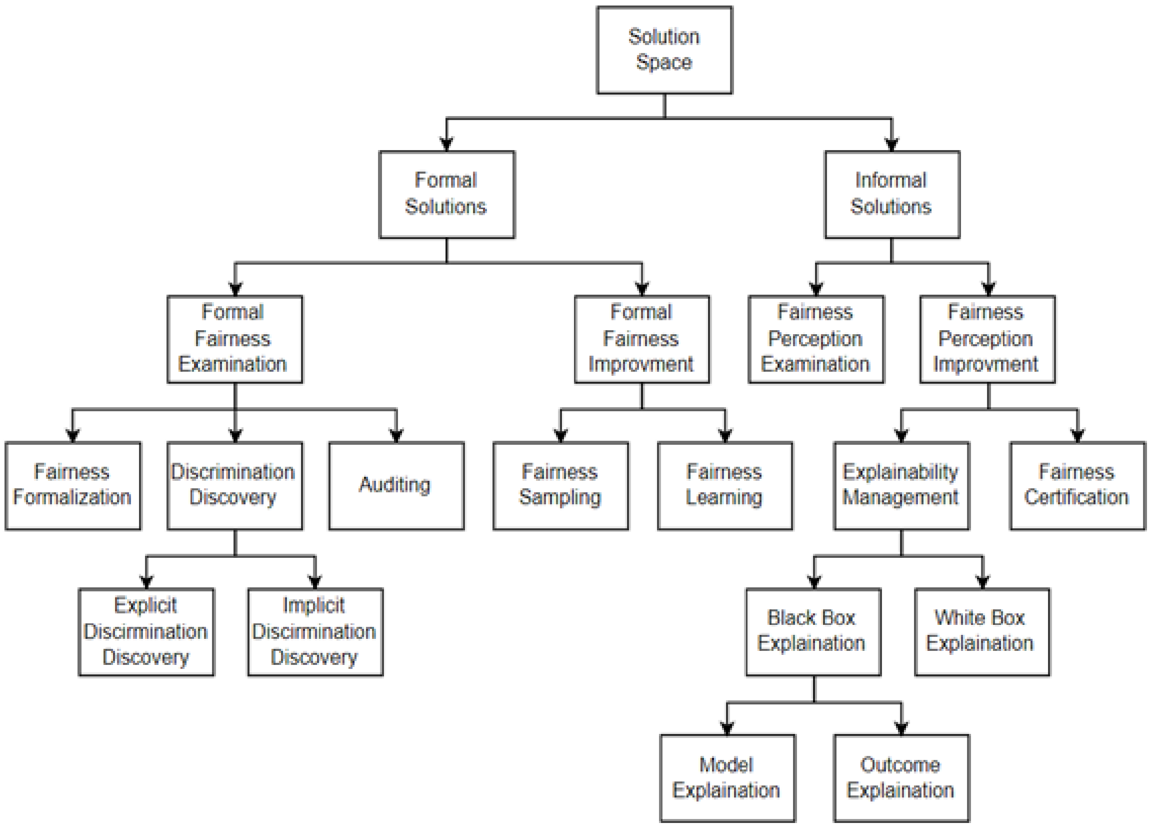

Figure 4. A hirarchical diagram of the solutions space - potential solutions to the challenges of biases and discrimination of an algorithmic system.

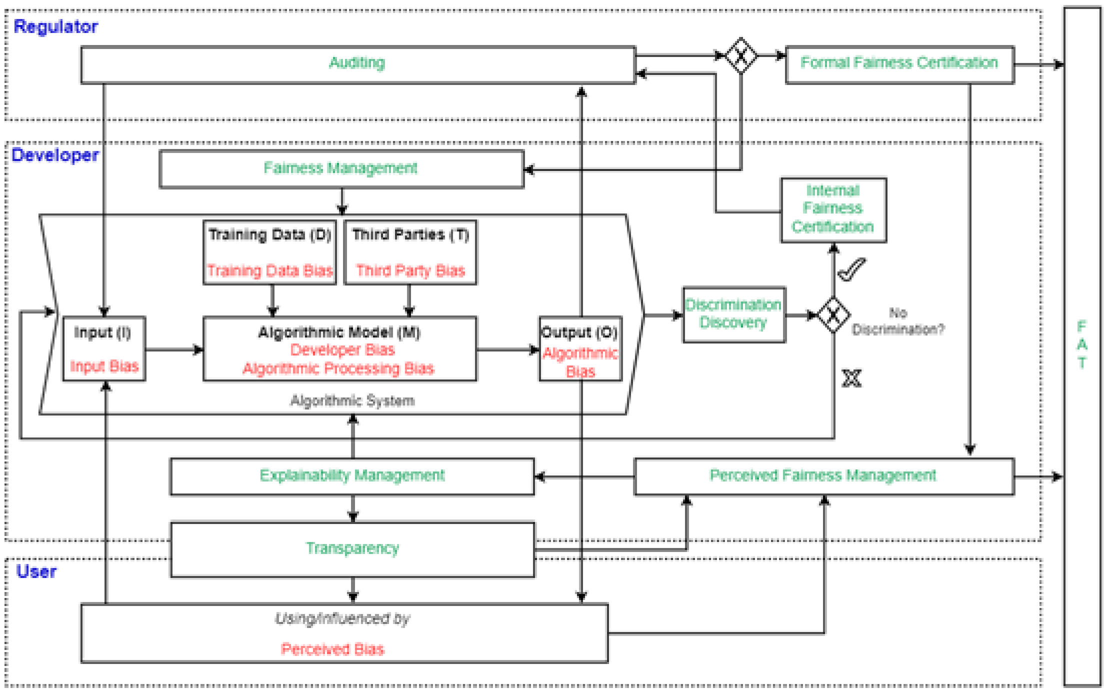

Figure 5. Comprehensive algorithmic system true fairness (ASTF) framework. The flow starts from the left hand side to the right. The framework considers different stakeholders (in blue) and their influence on the system (dashed boxes), the components of the algorithmic system (in bold black), the possible biases and their occurrences (in red), the suggested solutions (in green). Arrows describe the set of effects between different parts of the chart. The system is considered to be truly fair if both fairness certification from the regulator and perceived fairness management are applied and approved.

The theoretical part may take about half a day (given the relevant prior knowledge).
Then the hands on part should enable the users to experiment with a biased data set, use relevant tools to discover the discrimination and apply de-biasing techniques to overcome it.

Here the users may experience one aspect of data bias while additional techniques/approaches may be discussed

One example is: simulating the reasoning process of a system that evaluates CV of job applicants and recommends to accept a candidate that is clearly inappropriate or to reject a top-of-class candidate. For such demonstration there is no need to have a real system working.

Consider also introducing datasets and tools listed in section 4.

Sum up

Terms to be introduced during the discussion:

- Algorithmic system
- Algorithmic transparency
- Group fairness
- Individual fairness
- Discrimination discovery
- Biasses (all)
- Discrimination discovery (all)
- Fairness promotion
- Explainability

3.3. A Dedicated course (see 2.4)

When considering a dedicated course, then the focus may be more on hands-on experience and in-depth discussion - extending the tutorial to something between 28 and 56 hours of lectures and hands-on experience.

The course may follow the tutorial - presentation and discussion of the theoretical framework, however, in much greater details - looking at several examples of algorithmic systems - those that are based on what may be considered “white box” systems - systems that are not based on any machine learning algorithms and systems that are based on “transparent” machine learning algorithms - Decision Trees, Decision Rules, Baysean Reasoning - algorithms that their reasoning behavior can be traced and explained, those that are based on “Black Box” algorithms, where the reasoning process is untraceable.

A few relevant case known studies may be presented and analysed.

Then techniques for “opening the black box” may be presented.

After introducing the two types of systems families and their potential risks, tools for discrimination discovery may be presented and then tools for fairness assurance.

Participants may be given case studies that are intentionally biased that may lead them to develop rule based systems that consider protected attributes. Then, gender/racial/cultural bias introduced by the background story that reflects personal biases may be discussed, followed by the application of tools for bias removal/discrimination mitigation.

The course may conclude with an abstract discussion of fairness and it becoming a non-functional requirement of a system and how this requirement may be achieved during development and validated during system testing.

<u>Sum up</u>

- Algorithmic system
- Algorithmic transparency
- Group fairness
- Individual fairness
- Discrimination discovery
- Biasses (all)
- Discrimination discovery (all)
- Fairness promotion
- Explainability
- Auditing

3.4. A Graduate seminar (see 2.5)

In a graduate seminar, there may be a mix of frontal, introductory lectures of the lecturer, followed by a literature review and presentations by the students about their literature review. In such a seminar it is suggested to involve also invited speakers from different domains: Law, Philosophy, Economics, Sociology etc, as the issue of FAT is far beyond CS/IS alone.
A suggested structure/content for such seminar is:

1. An introduction to FAT
2. Incremental presentation of the general algorithmic system model

At this stage the students select topics for literature survey (abstract topics taken from the model)

3. Introduction of the different biases and their challenges
   Including illustrative case studies
4. Introduction of the various possible solutions
   Including illustrative case studies

Students presentations:

Topics may include: An overview of diversity and biases (where it all starts); Bias in training data, bias in input data (especially when reinforced learning is applied); 3rd party biases; Discrimination discovery (direct, indirect, white box, black box); Discrimination measurements; Unbiasing data; Explanations and explainability (for white box and black box systems); availavble tools;

Sum up

Terms to be introduced during the discussion:

- Algorithmic system
- Algorithmic transparency
- Group fairness
- Individual fairness
- Discrimination discovery
- Biasses (all)
- Discrimination discovery (all)
- Fairness promotion
- Explainability
- Auditing

3.5. A course for the developers (not students)

A dedicated course for already employed developers is more challenging than for the students. Therefore, an education could be done through a one-day workshop (with both theoretical and practical parts). Or as a one-month course with lectures and hands-on tutorials once a week. Specially, providing tools on collecting the training data and mitigating bias.

Recently, with the growth of Machine Learning, new challenges, as well as new Data Science related positions, were raised (Holstein, 2019). *Designers* of the AI systems should be taught to design interfaces that consider system explanations: what (the model and a prediction) and how (visual/voice/textual explanations). *Testers* of such systems should think of suitable testbeds to cover various populations (gender, demographic, etc.) and to be able to discover bias and unfairness.

Development and product managers should think about providing an appropriate auditing process for AI systems (Raji, 2020). Therefore, such courses will be brainstorming sessions that mix people, filling different positions in the company, to come up with novel ideas, satisfying their specific company requirements and constraints.

Sum up

Terms to be introduced during the discussion:

- Algorithmic system
- Algorithmic transparency
- Group fairness

- Individual fairness
- Discrimination discovery
- Biasses (all)
- Discrimination discovery (all)
- Fairness promotion
- Explainability
- Auditing

4. Available tools and datasets

Google's pair-code tool (https://pair-code.github.io/what-if-tool/) allows machine learning models inspection, including algorithmic fairness constraints testing. A developer can choose different fairness strategies (Group Unaware, Demographic Parity, Equal Opportunity, Equal Accuracy, Group Thresholds) and check a model accordingly (https://pair-code.github.io/what-if-tool/ai-fairness.html).
There is an open source python library for bias testing in machine learning applications (https://github.com/pymetrics/audit-ai). Another python toolbox for inspecting models for bias is FairML (https://github.com/adebayoj/fairml).
Unbias cards game can be used for raising awareness (https://unbias.wp.horizon.ac.uk/).

There are also other tools and techniques that may be used, including traditional SE techniques. For example, *source code documentation* - to make code transparent to other team members, including the testing team, *Code review* - focusing on FAT requirements, *Testing* - where test cases will be specifically designed (both white box and black box) for discrimination discovery and finally *Auditing* - by an external agency with a mandate for ensuring fairness and transparency. Note that, even if discrimination is not revealed by testing, this is not a guarantee of its absence.

Table 1 lists a set of available discrimination discovery tools. Instruction in the use of these tools and workflows is also best achieved through hands-on experience and exercises.

Tools	Description
Fairness Measures[1]	A framework for testing an algorithm on a variety of datasets and fairness metrics
Fairness Comparison[2]	An extensible test-bed for facilitating direct comparisons of algorithms, based on fairness metrics

[1] https://www.fairness-measures.org/
[2] https://github.com/algofairness/fairness-comparison

Themis-ML[3]	An open source machine learning library that implements several fairness-aware methods
FairML[4]	A python toolbox for auditing machine learning models for bias
Aequitas[5]	An open source bias audit toolkit to audit machine learning models for discrimination and bias
Fairtest[6]	A tool for discovering and testing for suspicious associations between an algorithm's outputs and protected populations
Audit-AI[7]	A Python library that implements fairness-aware machine learning algorithms
AI Fairness 360[8]	IBM’s open source toolkit for discovering discrimination and bias in ML
What-if tool[9]	Google’s tool allows machine learning models inspection, including algorithmic fairness constraints testing. A developer can choose different fairness strategies (Group Unaware, Demographic Parity, Equal Opportunity, Equal Accuracy, Group Thresholds) and check a model accordingly

Table 1. Tools for discrimination discovery and bias mitigation.

Dataset	Link + description
COMPAS dataset	This repository contains a Jupyter notebook and data for the ProPublica story "Machine Bias." Story: https://www.propublica.org/article/machine-bias-risk-assessments-in-criminal-sentencing/

[3] https://themis-ml.readthedocs.io/en/latest/
[4] https://github.com/adebayoj/fairml
[5] https://dsapp.uchicago.edu/projects/aequitas/
[6] https://github.com/columbia/fairtest
[7] https://github.com/pymetrics/audit-ai
[8] https://aif360.mybluemix.net/
[9] https://pair-code.github.io/what-if-tool/ai-fairness.html

	Methodology: https://www.propublica.org/article/how-we-analyzed-the-compas-recidivism-algorithm/ Notebook (you'll probably want to follow along in the methodology): https://github.com/propublica/compas-analysis/blob/master/Compas%20Analysis.ipynb Main Dataset: compas.db - a sqlite3 database containing criminal history, jail and prison time, demographics and COMPAS risk scores for defendants from Broward County. Other files as needed for the analysis. https://github.com/propublica/compas-analysis
German creditcard dataset	The dataset https://archive.ics.uci.edu/ml/datasets/Statlog+%28German+Credit+Data%29 An easy to understand version: The original dataset contains 1000 entries with 20 categorial/symbolic attributes prepared by Prof. Hofmann. In this dataset, each entry represents a person who takes a credit by a bank. Each person is classified as good or bad credit risks according to the set of attributes. The link to the original dataset can be found below. https://www.kaggle.com/uciml/german-credit Analysis examples https://srisai85.github.io/GermanCredit/German.html
Fraud detection dataset	The datasets contains transactions made by credit cards in September 2013 by european cardholders. This dataset presents transactions that occurred in two days, where we have 492 frauds out of 284,807 transactions. The dataset is highly unbalanced, the positive class (frauds) account for 0.172% of all transactions. https://www.kaggle.com/mlg-ulb/creditcardfraud
The Open Images dataset	A dataset of 9.2M images with unified annotations for image classification, object detection and visual relationship detection https://arxiv.org/pdf/1803.09010.pdf
The Open Images dataset - extended crowdsourced	Is an extension to Open Images dataset. To bring geographical and demographic diversity to imagery training data: https://research.google/tools/datasets/open-images-extended-crowdsourced/ Data Card: https://research.google/static/documents/datasets/open-images-extended-crowdsourced.pdf

Table 2. Publicly available biased datasets.

5. Courses/Educational activities/Material

There is a growing number of courses about fairness and transparency of algorithmic systems offered worldwide that can be considered when designing/giving a course about FAT. This plethora of courses includes tutorials, academic courses, courses provided by industry and governments. It is recommended to consult the growing number of material and publications that are available online. Below there is a list of a few possible sources to consult when planning a course:

5.1. Seminar in Algorithm Transparency Spring Semester 2018-2019 (see appendix)
5.2. DS-GA 3001.009: Special Topics in Data Science: Responsible Data Science (https://dataresponsibly.github.io/courses/spring19/)
5.3. A Course on Fairness, Accountability and Transparency in Machine Learning (https://geomblog.github.io/fairness/)
5.4. Online tutorial: A Tutorial on Fairness in Machine Learning (https://towardsdatascience.com/a-tutorial-on-fairness-in-machine-learning-3ff8ba1040cb)
5.5. Online "crash course" of fairness (google developers) (https://developers.google.com/machine-learning/crash-course/fairness/video-lecture)
5.6. Berkley: CS 294: Fairness in Machine Learning (https://fairmlclass.github.io/)
5.7. Coursera course on fairness in ML (https://www.coursera.org/lecture/machine-learning-business-professionals/fairness-in-ml-part-1-IUKPJ)
5.8. Sara Robinson: Building ML models for everyone: understanding fairness in machine learning: (https://cloud.google.com/blog/products/ai-machine-learning/building-ml-models-for-everyone-understanding-fairness-in-machine-learning)
5.9. Foundations of Fairness in ML (http://jamiemorgenstern.com/teaching/f18-fairml/)
5.10. A (very) large number of talks is also available on youtube and it is worth considering them as part of an introductory course:
    5.10.1. https://www.youtube.com/results?search_query=fairness+machine+learning - a few examples below:
    5.10.2. The Emerging Theory of Algorithmic Fairness (MS research) (https://www.youtube.com/watch?v=g-z84_nRQhw)
    5.10.3. Machine Learning Fairness: Lessons Learned (Google I/O'19) (https://www.youtube.com/watch?v=6CwzDoE8J4M)
    5.10.4. Machine Learning, Ethics and Fairness (https://www.youtube.com/watch?v=YQpnd7z0HO8)
5.11. Artificial Intelligence: The Problem with Bias, with Kate Crawford: https://player.fm/series/city-arts-lectures/artificial-intelligence-the-problem-with-bias-with-kate-crawford

6. Additional resources

In addition to the suggested training ideas and the plethora of courses, there is a large number of sources that are available online, some of them are listed below:

6.1. A work-in-progress - online book about Fairness and Machine Learning: Solon Barocas, Moritz Hardt, Arvind Narayanan Fairness and machine learning Limitations and Opportunities (https://fairmlbook.org/)
6.2. Researcher develops algorithm to make artificial intelligence fairer https://www.qub.ac.uk/ecit/News/Researcherdevelopsalgorithmtomakeartificialintelligencefairer.html

7. Summary

This document provided a set of suggestions for tailoring education activities for whoever is involved in the development and operation of algorithmic systems in varying levels of depth and details, together with a list of possible datasets and tools and pointers to similar activities that were carried out in different institutions. It is intended to serve as a guideline to whoever is planning such educational activity.

8. References

1. Dieterich, W., Mendoza, C., & Brennan, T. (2016). COMPAS risk scales: Demonstrating accuracy equity and predictive parity. *Northpoint Inc*.
2. Edelman, B., Luca, M., & Svirsky, D. (2017). Racial discrimination in the sharing economy: Evidence from a field experiment. *American Economic Journal: Applied Economics*, *9*(2), 1-22.
3. Feller, A., Pierson, E., Corbett-Davies, S., & Goel, S. (2016). A computer program used for bail and sentencing decisions was labeled biased against blacks. It's actually not that clear. *The Washington Post*.
4. Holstein, K., Wortman Vaughan, J., Daumé III, H., Dudik, M., & Wallach, H. (2019, May). Improving fairness in machine learning systems: What do industry practitioners need?. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems* (pp. 1-16).
5. Raji, I. D., Smart, A., White, R. N., Mitchell, M., Gebru, T., Hutchinson, B., ... & Barnes, P. (2020). Closing the AI Accountability Gap: Defining an End-to-End Framework for Internal Algorithmic Auditing. *arXiv preprint arXiv:2001.00973*.
6. Washington, A. L. (2018). How to Argue with an Algorithm: Lessons from the COMPAS-ProPublica Debate. *Colo. Tech. LJ*, *17*, 131.
7. Zhang, M. Google Photos Tags Two African-Americans As Gorillas Through Facial Recognition Software. 2015.

Appendix

Seminar in Algorithm Transparency

Spring Semester 2018-2019

Lecturer: Professor Tsvi Kuflik email: tsvikak@is.haifa.ac.il

Office hours: Tuesdays 09:00-10:00 Room 7050 Rabin Building

Class times: Wednesdays 14:15-17:45 Room 6012 Rabin Building

Tutorials: N/A

Eligibility: Seminar for graduate students (M. Sc.) in Information Systems. The course is also open to advanced level undergraduate (B. Sc.) students.

Pre-requisite Courses: Software Engineering, Information System Analysis.

Course Description:

The course is an advanced seminar studying a question of ever growing concern to both the wider public and professionals in Information Systems and Computer Technologies – How to make computing technologies more understandable and friendly (accessible/accountable) to their end users? The first half of the course will present the recent literature on topics related to "Algorithmic Transparency". The rest of the course will consist of in-depth presentations of relevant topics prepared by students. Students will also be required to carry out a practical project related to the course material.

Full attendance at all classes is compulsory.

Required Reading:

Practical Project:

The practical work will be performed in weekly increments by the project teams. Each project will focus on one the ACM's seven principles of Algorithmic Transparency and Accountability (Awareness, Access and Redress, Accountability, Explanation, Data Provenance, Auditability and Validation/Testing). The final reports will be submitted in the last week of the semester, and a presentation of the results will take place in the final class.

Evaluation Scheme:

40% - Theoretical subject presentation

40% - Practical group project

20% - Active participation in seminar discussions.

Detailed course breakdown (subject to modifications based on progress)

Week	Subject	Exercises	Reading
1	Introduction to Algorithmic Transparency, presentation of the issues, the challenges and their significance. Current research directions.		
2	Transparency in machine learning		
3	Explanations		
4	Algorithmic bias		
5	Fairness, ethics and social accountability		
6	Practical examples		
7	3-4 student presentations *		
8	3-4 student presentations		
9	3-4 student presentations		
10	3-4 student presentations		
11	3-4 student presentations		
12	3-4 student presentations		
13	Practical project presentations, course summary		

***** Graduate students will present their theoretical study as individuals, undergraduates will work in pairs. Presentations by graduate students will be of one hour's length, undergraduate pairs will present for one and a half hours.

References:

Guest Lecturers: