Engineering Fair and Efficient Learning-Based Software Systems

Giordano d'Aloisio

SWEN Talk, 19th February, 2025

University of L'Aquila, Italy

UNIVERSITÀ DEGLI STUDI DELL'AQUILA

DISIM
Dipartimento di Ingegneria e Scienze dell'Informazione e Matematica

SWEN research group

About me

Giordano d'Aloisio

I'm a PhD candidate in Information and Communication Technology at the University of L'Aquila in Italy under the supervision of Prof. Antinisca Di Marco.

I obtained my bachelor degree in Computer Science for Business Economy at the University of Chieti-Pescara in 2017 and my master degree in Computer Science at the University of L'Aquila in 2021. From 2019 to 2024, I'm a member of the Territori Aperti project where I'm responsible of the Data Integration activity. I'm also a member of the Emeliot, Fair-EDU, FRINGE, and PinKamP projects. From Febraury to August 2024 I was a visiting researcher at the University College London (UCL) as a member of the SOLAR research group working with Prof. Federica Sarro.

My research is mostly focused on Data Science and Software Engineering techniques for the quality assurance of Machine Learning systems, with a particular attention on Software Bias and Fairness.

University of L'Aquila
DISIM Department
via Vetoio
67100 L'Aquila, Italy

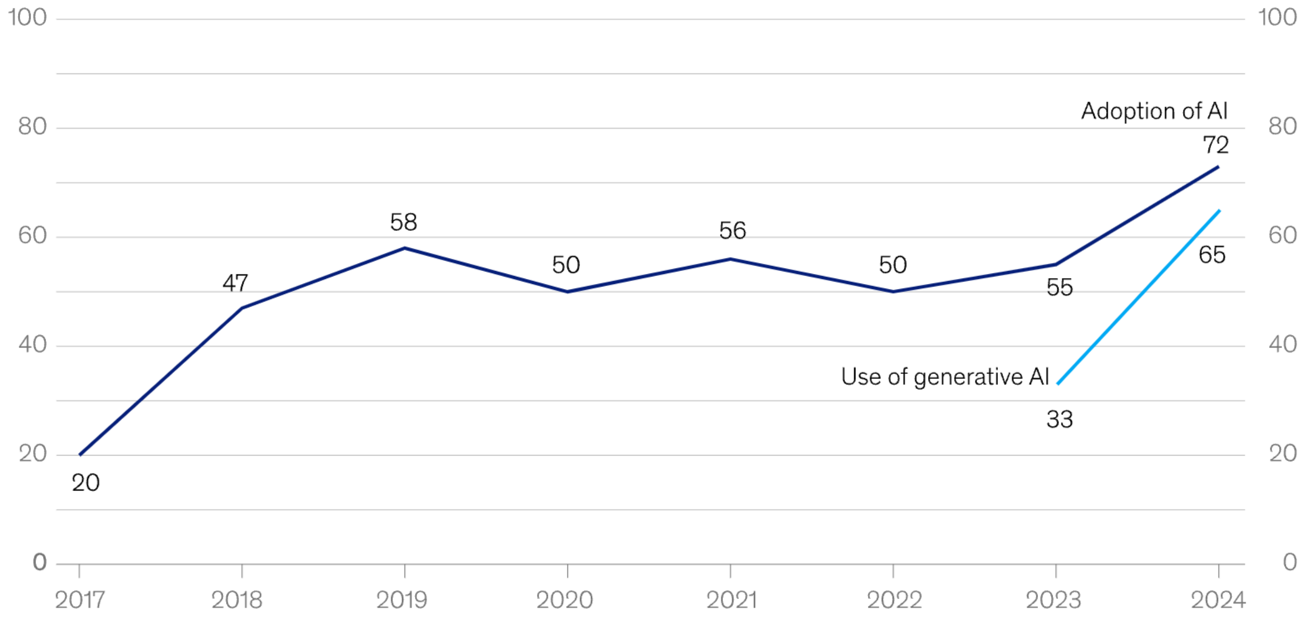
Research Interests

- Software Fairness
- Software Engineering for Machine Learning
- Empirical Software Engineering
- Human Aspects in Software Engineering

Introduction

- Learning-based systems (LBS) are software systems that employ AI models

Organizations that have adopted AI in at least 1 business function,[1] % of respondents



https://www.mckinsey.com/capabilities/quantumblack/our-insights/the-state-of-ai#/

AI adoption is not without risks

TECH

Facebook's ad delivery system still has gender bias, new study finds

ARTIFICIAL INTELLIGENCE

London police's face recognition system gets it wrong 81% of the time

AI Power Consumption Exploding

Exponential increase is not sustainable. But where is it all going?

A review of green artificial intelligence: Towards a more sustainable future

Verónica Bolón-Canedo *, Laura Morán-Fernández, Brais Cancela, Amparo Alonso-Betanzos

CITIC, Universidade da Coruña, A Coruña, Spain

AI adoption is not without risks

TECH

Facebook's ad delivery system still has gender bias, new study finds

ARTIFICIAL INTELLIGENCE

Quality-based development of learning-based systems is paramount

Exponential increase is not sustainable. But where is it all going?

A review of green artificial intelligence: Towards a more sustainable future

Verónica Bolón-Canedo *, Laura Morán-Fernández, Brais Cancela, Amparo Alonso-Betanzos

CITIC, Universidade da Coruña, A Coruña, Spain

Considered Quality Attributes

Considered Quality Attributes

Fairness

The absence of prejudice or favoritism of a learning-based system toward individuals or groups

Considered Quality Attributes

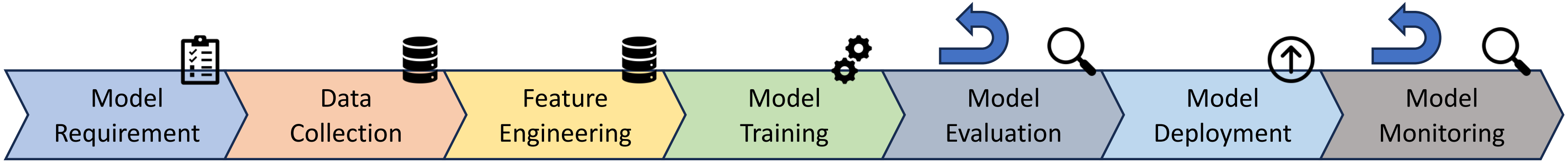
Fairness

The absence of prejudice or favoritism of a learning-based system toward individuals or groups

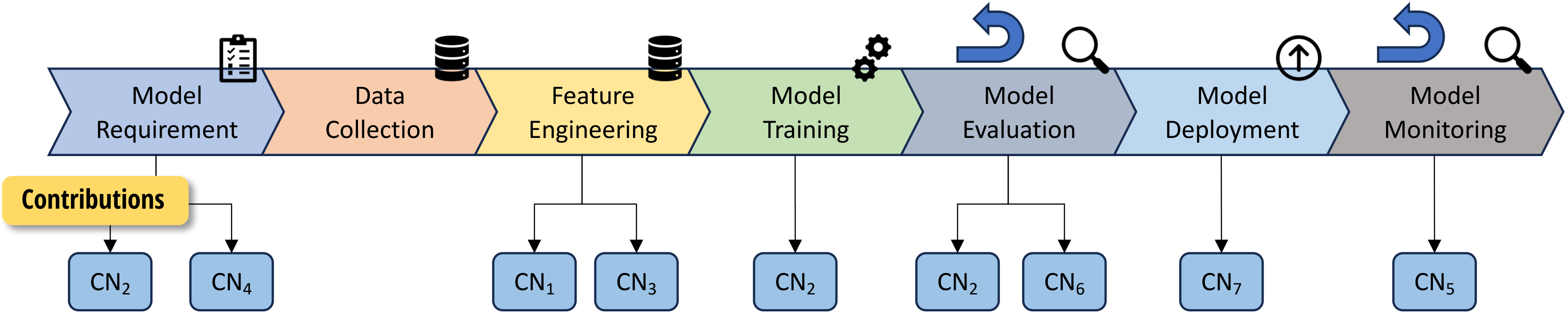
Efficiency

Time and memory required by a learning-based system for its operations

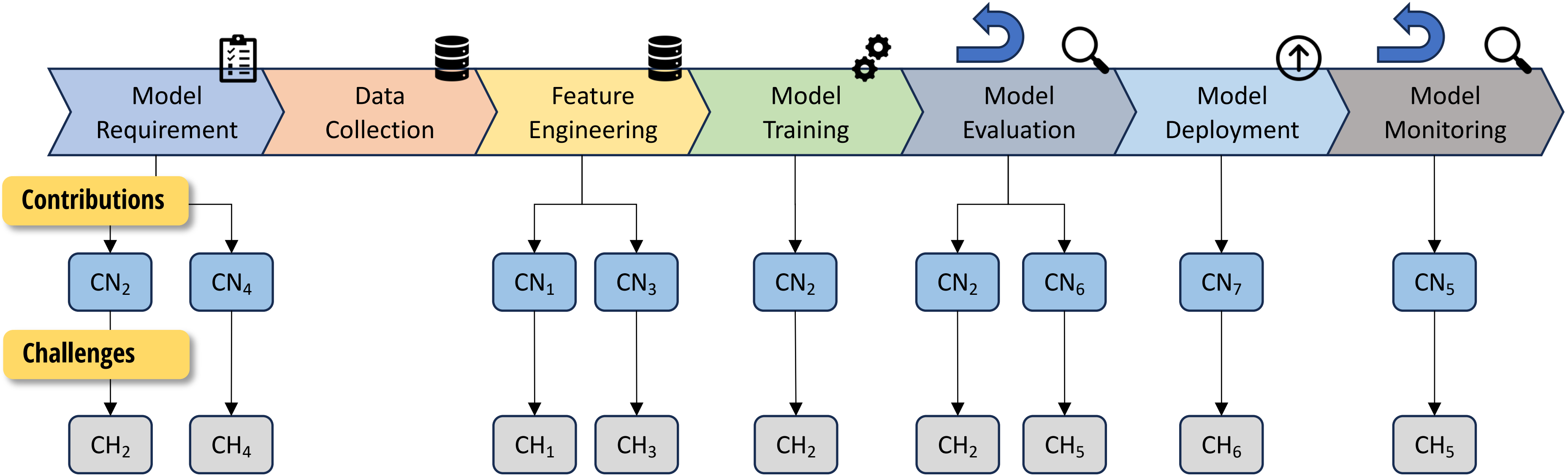
Our Contributions



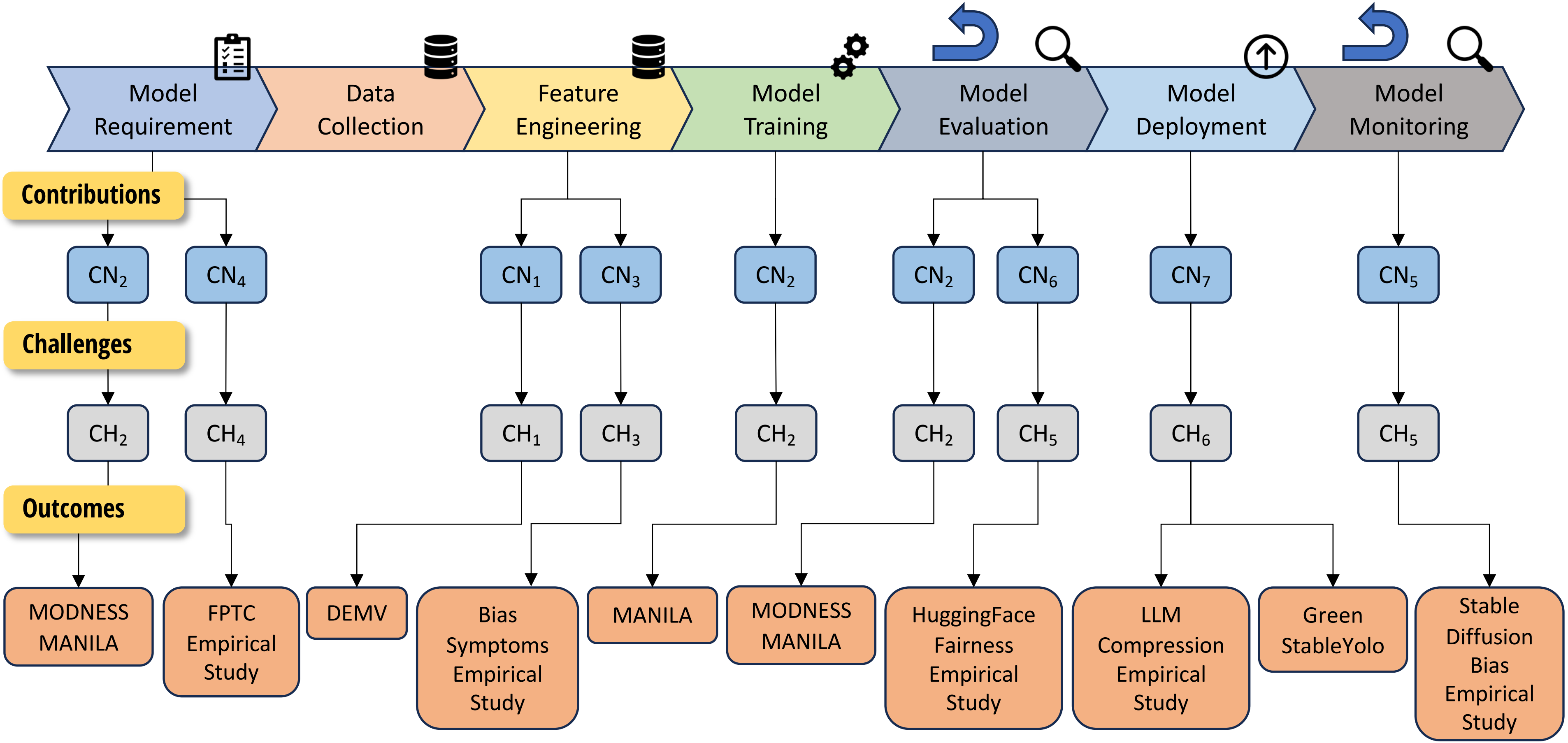
Our Contributions



Our Contributions



Our Contributions



Fairness of Learning-Based Systems

COMPAS

- COMPAS was an LBS used by some courts in the US to predict recidivism of condemned people
- A study showed that, given two people with the same features but different ethnicity, the system was giving higher probability of recidivism to non-white people

Bernard Parker, left, was rated high risk; Dylan Fugett was rated low risk. (Josh Ritchie for ProPublica)

Machine Bias

There's software used across the country to predict future criminals. And it's biased against blacks.

by Julia Angwin, Jeff Larson, Surya Mattu and Lauren Kirchner, ProPublica
May 23, 2016

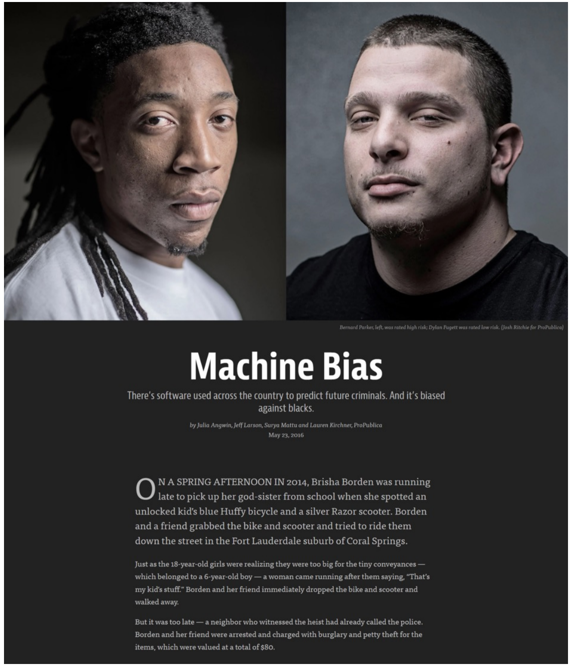
ON A SPRING AFTERNOON IN 2014, Brisha Borden was running late to pick up her god-sister from school when she spotted an unlocked kid's blue Huffy bicycle and a silver Razor scooter. Borden and a friend grabbed the bike and scooter and tried to ride them down the street in the Fort Lauderdale suburb of Coral Springs.

Just as the 18-year-old girls were realizing they were too big for the tiny conveyances — which belonged to a 6-year-old boy — a woman came running after them saying, "That's my kid's stuff." Borden and her friend immediately dropped the bike and scooter and walked away.

But it was too late — a neighbor who witnessed the heist had already called the police. Borden and her friend were arrested and charged with burglary and petty theft for the items, which were valued at a total of $80.

COMPAS

- COMPAS was an LBS used by some courts in the US to predict recidivism of condemned people
- A study showed that, given two people with the same features but different ethnicity, the system was giving higher probability of recidivism to non-white people



Bernard Parker, left, was rated high risk; Dylan Fugett was rated low risk. (Josh Ritchie for ProPublica)

Machine Bias

There's software used across the country to predict future criminals. And it's biased against blacks.

by Julia Angwin, Jeff Larson, Surya Mattu and Lauren Kirchner, ProPublica

May 23, 2016

ON A SPRING AFTERNOON IN 2014, Brisha Borden was running late to pick up her god-sister from school when she spotted an unlocked kid's blue Huffy bicycle and a silver Razor scooter. Borden and a friend grabbed the bike and scooter and tried to ride them down the street in the Fort Lauderdale suburb of Coral Springs.

Just as the 18-year-old girls were realizing they were too big for the tiny conveyances — which belonged to a 6-year-old boy — a woman came running after them saying, "That's my kid's stuff." Borden and her friend immediately dropped the bike and scooter and walked away.

But it was too late — a neighbor who witnessed the heist had already called the police. Borden and her friend were arrested and charged with burglary and petty theft for the items, which were valued at a total of $80.

The system was biased against non-white people

Software Fairness

- Fairness is the absence of prejudice or favoritism (i.e., *bias*) of an LBS over items identified by a set of *sensitive variables*
- Fairness is usually defined by a set of relevant concepts

Sensitive variables

Privileged and unprivileged groups

Favorable outcome

Example on COMPAS

Sensitive variables

Ethnicity

Privileged and unprivileged groups

White – Non-White

Favorable outcome

No recidivism

Is the concept of bias that simple?

Is the concept of bias that simple?

A Survey on Bias and Fairness in Machine Learning 115:5

(1) **Measurement Bias.** *Measurement, or reporting, bias arises from how we choose, utilize, and measure particular features* [140]. An example of this type of bias was observed in the recidivism risk prediction tool COMPAS, where prior arrests and friend/family arrests were used as proxy variables to measure level of "riskiness" or "crime"—which on its own can be viewed as mismeasured proxies. This is partly due to the fact that minority communities are controlled and policed more frequently, so they have higher arrest rates. However, one should not conclude that because people coming from minority groups have higher arrest rates, therefore they are more dangerous, as there is a difference in how these groups are assessed and controlled [140].

(2) **Omitted Variable Bias.** *Omitted variable bias*[4] *occurs when one or more important variables are left out of the model* [38, 110, 127]. An example for this case would be when someone designs a model to predict, with relatively high accuracy, the annual percentage rate at which customers will stop subscribing to a service, but soon observes that the majority of users are canceling their subscription without receiving any warning from the designed model. Now imagine that the reason for canceling the subscriptions is appearance of a new strong competitor in the market that offers the same solution, but for half the price. The appearance of the competitor was something that the model was not ready for; therefore, it is considered to be an omitted variable.

(3) **Representation Bias.** *Representation bias arises from how we sample from a population during data collection process* [140]. Non-representative samples lack the diversity of the population, with missing subgroups and other anomalies. Lack of geographical diversity in datasets like ImageNet (as shown in Figures 3 and 4) results in demonstrable bias towards Western cultures.

Is the concept of bias that simple?

(1) **Measurement Bias.** *Measurement, or reporting, bias arises from how we choose, utilize, and measure particular features* [140]. An example of this type of bias was observed in the recidivism risk prediction tool COMPAS, where prior arrests and friend/family arrests were used as proxy variables to measure level of "riskiness" or "crime"—which on its own can be

3.1.2 Algorithm to User. Algorithms modulate user behavior. Any biases in algorithms might introduce biases in user behavior. In this section, we talk about biases that are as a result of algorithmic outcomes and affect user behavior as a consequence.

(1) **Algorithmic Bias.** *Algorithmic bias is when the bias is not present in the input data and is added purely by the algorithm* [9]. The algorithmic design choices, such as use of certain optimization functions, regularizations, choices in applying regression models on the data as a whole or considering subgroups, and the general use of statistically biased estimators in algorithms [44], can all contribute to biased algorithmic decisions that can bias the outcome of the algorithms.
(2) **User Interaction Bias.** *User Interaction bias is a type of bias that can not only be observant on the Web but also get triggered from two sources—the user interface and through the user itself by imposing his/her self-selected biased behavior and interaction* [9]. This type of bias can be influenced by other types and subtypes, such as presentation and ranking biases.
  (a) **Presentation Bias.** *Presentation bias is a result of how information is presented [9].* For example, on the Web users can only click on content that they see, so the seen content gets clicks, while everything else gets no click. And it could be the case that the user does not see all the information on the Web [9].
  (b) **Ranking Bias.** *The idea that top-ranked results are the most relevant and important will result in attraction of more clicks than others.* This bias affects search engines [9] and crowdsourcing applications [92].
(3) **Popularity Bias.** *Items that are more popular tend to be exposed more. However, popularity metrics are subject to manipulation—for example, by fake reviews or social bots* [113]. As an

Is the concept of bias that simple?

(1) **Measurement Bi**
measure particular
cidivism risk predi
used as proxy vari

3.1.2 *Algorithm to User.*
introduce biases in user beha
rithmic outcomes and affect

(1) **Algorithmic Bias.** *Al*
added purely by the al
optimization functions
as a whole or consideri
algorithms [44], can all
of the algorithms.
(2) **User Interaction Bias**
the Web but also get tri
by imposing his/her sel
influenced by other typ
(a) **Presentation Bias**
example, on the W
gets clicks, while e
does not see all the
(b) **Ranking Bias.** *Th*
result in attraction of more clicks than others. This bias affects search engines [9] and crowdsourcing applications [92].
(3) **Popularity Bias.** *Items that are more popular tend to be exposed more. However, popularity metrics are subject to manipulation—for example, by fake reviews or social bots* [113]. As an

to the fact that only 5% of Fortune 500 CEOs were women—which would cause the search results to be biased towards male CEOs [140]. These search results were of course reflecting the reality, but whether or not the search algorithms should reflect this reality is an issue worth considering.

(2) **Population Bias.** *Population bias arises when statistics, demographics, representatives, and user characteristics are different in the user population of the platform from the original target population* [116]. Population bias creates non-representative data. An example of this type of bias can arise from different user demographics on different social platforms, such as women being more likely to use Pinterest, Facebook, Instagram, while men being more active in online forums like Reddit or Twitter. More such examples and statistics related to social media use among young adults according to gender, race, ethnicity, and parental educational background can be found in Reference [64].
(3) **Self-selection Bias.** *Self-selection bias*[4] *is a subtype of the selection or sampling bias in which subjects of the research select themselves.* An example of this type of bias can be observed in an opinion poll to measure enthusiasm for a political candidate, where the most enthusiastic supporters are more likely to complete the poll.
(4) **Social Bias.** *Social bias happens when others' actions affect our judgment* [9]. An example of this type of bias can be a case where we want to rate or review an item with a low score, but when influenced by other high ratings, we change our scoring thinking that perhaps we are being too harsh [9, 147].
(5) **Behavioral Bias.** *Behavioral bias arises from different user behavior across platforms, contexts, or different datasets* [116]. An example of this type of bias can be observed in Reference [104], where authors show how differences in emoji representations among platforms can result in

Is the concept of bias that simple?

A Survey on Bias and Fairness in Machine Learning 115:5

(1) **Measurement Bi**
measure particular
cidivism risk predi
used as proxy vari

3.1.2 *Algorithm to User.*
introduce biases in user beha
rithmic outc

(1) **Algori**
added
optimi
as a wh
algorit
of the
(2) **User I**
the We
by imp
influen
(a) **Pre**
exa
get
doe
(b) **Ra**
res
cro
(3) **Popul**
metrics

to the fact that only 5% of Fortune 500 CEOs were women—which would cause the search results to be biased towards male CEOs [140]. These search results were of course reflecting the reality, but whether or not the search algorithms should reflect this reality is an issue worth considering.

(2) **Population Bias.** *Population bias arises when statistics, demographics, representatives, and user characteristics are different in the user population of the platform from the original target population* [116]. Population bias creates non-representative data. An example of this type

such as
re active
to social
cational

in which
erved in
usiastic

ample of
core, but
s we are

contexts,
ce [104],
result in

A Survey on Bias and Fairness in Machine Learning 115:9

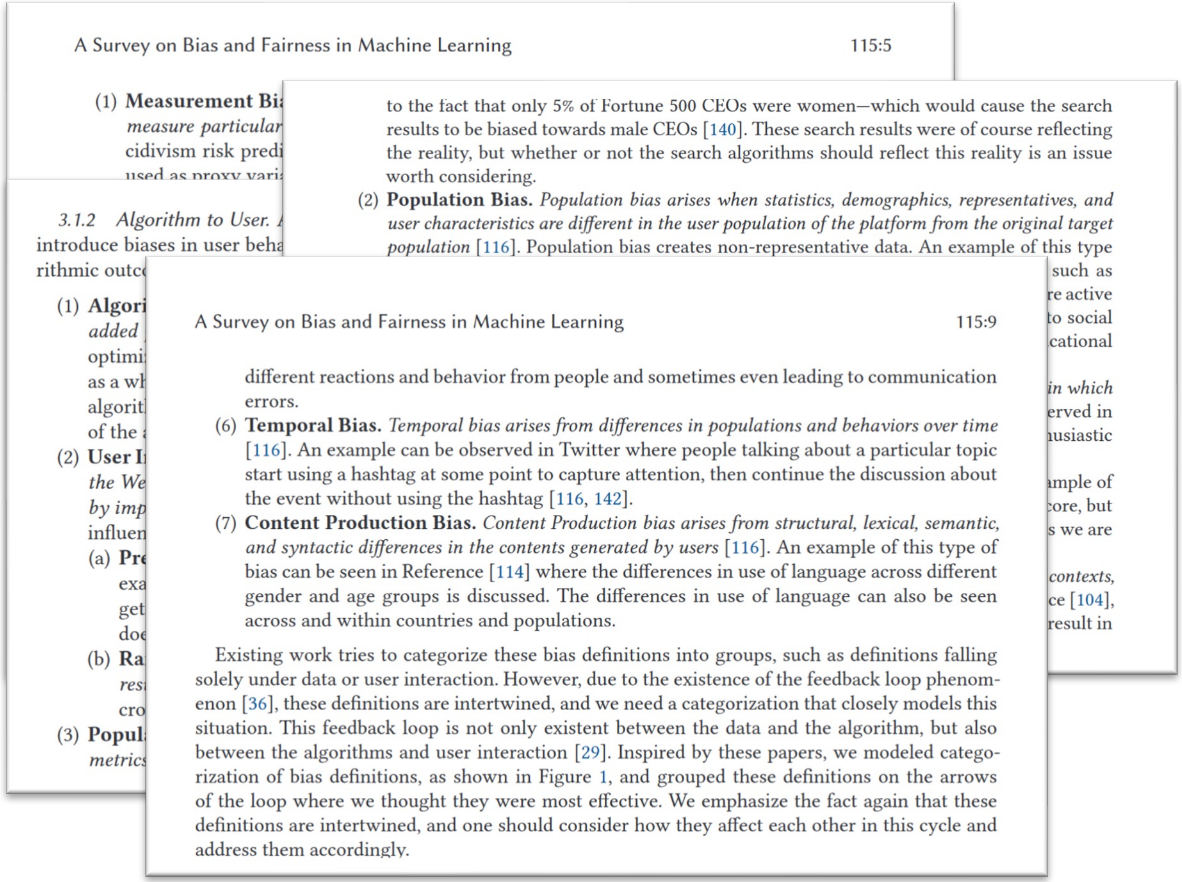
different reactions and behavior from people and sometimes even leading to communication errors.

(6) **Temporal Bias.** *Temporal bias arises from differences in populations and behaviors over time* [116]. An example can be observed in Twitter where people talking about a particular topic start using a hashtag at some point to capture attention, then continue the discussion about the event without using the hashtag [116, 142].

(7) **Content Production Bias.** *Content Production bias arises from structural, lexical, semantic, and syntactic differences in the contents generated by users* [116]. An example of this type of bias can be seen in Reference [114] where the differences in use of language across different gender and age groups is discussed. The differences in use of language can also be seen across and within countries and populations.

Existing work tries to categorize these bias definitions into groups, such as definitions falling solely under data or user interaction. However, due to the existence of the feedback loop phenomenon [36], these definitions are intertwined, and we need a categorization that closely models this situation. This feedback loop is not only existent between the data and the algorithm, but also between the algorithms and user interaction [29]. Inspired by these papers, we modeled categorization of bias definitions, as shown in Figure 1, and grouped these definitions on the arrows of the loop where we thought they were most effective. We emphasize the fact again that these definitions are intertwined, and one should consider how they affect each other in this cycle and address them accordingly.

Is the concept of bias that simple?


A Survey on Bias and Fairness in Machine Learning 115:5

(1) **Measurement Bi**
measure particular
cidivism risk predi
used as proxy vari

3.1.2 *Algorithm to User.*
introduce biases in user beha
rithmic outc

to the fact that only 5% of Fortune 500 CEOs were women—which would cause the search results to be biased towards male CEOs [140]. These search results were of course reflecting the reality, but whether or not the search algorithms should reflect this reality is an issue worth considering.

(2) **Population Bias.** *Population bias arises when statistics, demographics, representatives, and user characteristics are different in the user population of the platform from the original target population* [116]. Population bias creates non-representative data. An example of this type

A Survey on Bias and Fairness in Machine Learning 115:9

different reactions and behavior from people and sometimes even leading to communication errors.

(6) **Temporal Bias.** *Temporal bias arises from differences in populations and behaviors over time* [116]. An example can be observed in Twitter where people talking about a particular topic start using a hashtag at some point to capture attention, then continue the discussion about the event without using the hashtag [116, 142].

(7) **Content Production Bias.** *Content Production bias arises from structural, lexical, semantic, and syntactic differences in the contents generated by users* [116]. An example of this type of bias can be seen in Reference [114] where the differences in use of language across different gender and age groups is discussed. The differences in use of language can also be seen across and within countries and populations.

Existing work tries to categorize these bias definitions into groups, such as definitions falling solely under data or user interaction. However, due to the existence of the feedback loop phenomenon [36], these definitions are intertwined, and we need a categorization that closely models this situation. This feedback loop is not only existent between the data and the algorithm, but also between the algorithms and user interaction [29]. Inspired by these papers, we modeled categorization of bias definitions, as shown in Figure 1, and grouped these definitions on the arrows of the loop where we thought they were most effective. We emphasize the fact again that these definitions are intertwined, and one should consider how they affect each other in this cycle and address them accordingly.

At least 23 different definitions of bias in the literature

From many definitions come many metrics…

From many definitions come many metrics...

Generic metrics

`metrics.num_samples` (y_true[, y_pred, ...])	Compute the number of samples.
`metrics.num_pos_neg` (y_true[, y_pred, ...])	Compute the number of positive and negative samples.
`metrics.specificity_score` (y_true, y_pred, *)	Compute the specificity or true negative rate.
`metrics.sensitivity_score` (y_true, y_pred[, ...])	Alias of `sklearn.metrics.recall_score()` for binary classes only.
`metrics.base_rate` (y_true[, y_pred, ...])	Compute the base rate, $Pr(Y = \text{pos\_label}) = \frac{P}{P+N}$.
`metrics.selection_rate` (y_true, y_pred, *[, ...])	Compute the selection rate, $Pr(\hat{Y} = \text{pos\_label}) = \frac{TP+FP}{P+N}$.
`metrics.smoothed_base_rate` (y_true[, y_pred, ...])	Compute the smoothed base rate, $\frac{P+\alpha}{P+N+\|R_Y\|\alpha}$.
`metrics.smoothed_selection_rate` (y_true, ...)	Compute the smoothed selection rate, $\frac{TP+FP+\alpha}{P+N+\|R_Y\|\alpha}$.
`metrics.generalized_fpr` (y_true, probas_pred, *)	Return the ratio of generalized false positives to negative examples in the dataset, $GFPR = \frac{GFP}{N}$.
`metrics.generalized_fnr` (y_true, probas_pred, *)	Return the ratio of generalized false negatives to positive examples in the dataset, $GFNR = \frac{GFN}{P}$.

From many definitions come many metrics…

Generic metrics

`metrics.num_samples` (y_true[, y_pred, …])	Compute the number of samples.
`metrics.num_pos_neg` (y_true[, y_pred, …])	Compute the number of positive and negative samples.
`metrics.specificity_score` (y_true, y_pred, *)	Compute the specificity or true negative rate.
`metrics.sensitivity_score` (y_true, y_pred[, …])	Alias of `sklearn.metrics.recall_score()` for binary classes only.
`metrics.base_rate` (y_true[, y_pred, …])	Compute the base rate, $Pr(Y = \text{pos\_label}) = \frac{P}{P+N}$.
`metrics.selection_rate` (y_true, y_pred, *[, …])	Compute the selection rate, $Pr(\hat{Y} = \text{pos\_label}) = \frac{TP+FP}{P+N}$.
`metrics.smoothed_base_rate` (y_true[, y_pred, …])	Compute the smoothed base rate, $\frac{P+\alpha}{P+N+\|R_Y\|\alpha}$.
`metrics.smoothed_selection_rate` (y_true, …)	Compute the smoothed selection rate, $\frac{TP+FP+\alpha}{P+N+\|R_Y\|\alpha}$.
`metrics.generalized_fpr` (y_true, probas_pred, *)	Return the ratio of generalized false positives to negative examples in the dataset, $GFPR = \frac{GFP}{N}$.
`metrics.generalized_fnr` (y_true, probas_pred, *)	Return the ratio of generalized false negatives to positive examples in the dataset, $GFNR = \frac{GFN}{}$

Individual fairness metrics

`metrics.generalized_entropy_index` (b[, alpha])	Generalized entropy index measures inequality over a population.
`metrics.generalized_entropy_error` (y_true, y_pred)	Compute the generalized entropy.
`metrics.theil_index` (b)	The Theil index is the `generalized_entropy_index()` with $\alpha = 1$.
`metrics.coefficient_of_variation` (b)	The coefficient of variation is the square root of two times the `generalized_entropy_index()` with $\alpha = 2$.
`metrics.consistency_score` (X, y[, n_neighbors])	Compute the consistency score.

From many definitions come many metrics...

Generic metrics

`metrics.num_samples` (y_true[, y_pred, ...])	Compute the number of samples.
`metrics.num_pos_neg` (y_true[, y_pred, ...])	Compute the number of positive and negative samples
`metrics.specificity_score` (y_true, y_pred, *)	Compute the specificity or true negative rate.
`metrics.sensitivity_score` (y_true, y_pred[, ...])	Alias of `sklearn.metrics.recall_score()` for binary classe
`metrics.base_rate` (y_true[, y_pred, ...])	Compute the base rate, $Pr(Y = \text{pos\_label}) = \frac{P}{P+N}$
`metrics.selection_rate` (y_true, y_pred, *[, ...])	Compute the selection rate, $Pr(\hat{Y} = \text{pos\_label}) =$
`metrics.smoothed_base_rate` (y_true[, y_pred, ...])	Compute the smoothed base rate, $\frac{P+\alpha}{P+N+\lvert R_Y\rvert\alpha}$.
`metrics.smoothed_selection_rate` (y_true, ...)	Compute the smoothed selection rate, $\frac{TP+FP+\alpha}{P+N+\lvert R_Y\rvert\alpha}$.
`metrics.generalized_fpr` (y_true, probas_pred, *)	Return the ratio of generalized false positives to negati the dataset, $GFPR = \frac{GFP}{N}$.
`metrics.generalized_fnr` (y_true, probas_pred, *)	Return the ratio of generalized false negatives to positi

Individual fairness metrics

`metrics.generalized_entropy_index` (b[, alpha])	Generalized entropy index measures inequality over a
`metrics.generalized_entropy_error` (y_true, y_pred)	Compute the generalized entropy.
`metrics.theil_index` (b)	The Theil index is the `generalized_entropy_index()` with
`metrics.coefficient_of_variation` (b)	The coefficient of variation is the square root of two ti `generalized_entropy_index()` with $\alpha = 2$.
`metrics.consistency_score` (X, y[, n_neighbors])	Compute the consistency score.

Group fairness metrics

`metrics.statistical_parity_difference` (y_true)	Difference in selection rates.
`metrics.mean_difference` (y_true[, y_pred, ...])	Alias of `statistical_parity_difference()`.
`metrics.disparate_impact_ratio` (y_true[, ...])	Ratio of selection rates.
`metrics.equal_opportunity_difference` (y_true, ...)	A relaxed version of equality of opportunity.
`metrics.average_odds_difference` (y_true, ...)	A relaxed version of equality of odds.
`metrics.average_odds_error` (y_true, y_pred, *)	A relaxed version of equality of odds.
`metrics.class_imbalance` (y_true[, y_pred, ...])	Compute the class imbalance, $\frac{N_u - N_p}{N_u + N_p}$.
`metrics.kl_divergence` (y_true[, y_pred, ...])	Compute the Kullback-Leibler divergence, $KL(P_p \lvert\lvert P_u) = \sum_y P_p(y) \log\left(\frac{P_p(y)}{P_u(y)}\right)$
`metrics.conditional_demographic_disparity` (y_true)	Conditional demographic disparity, $CDD = \frac{1}{\sum_i N_i} \sum_i N_i \cdot DD_i$
`metrics.smoothed_edf` (y_true[, y_pred, ...])	Smoothed empirical differential fairness (EDF).
`metrics.df_bias_amplification` (y_true, y_pred, *)	Differential fairness bias amplification.
`metrics.between_group_generalized_entropy_error` (...)	Compute the between-group generalized entropy.
`metrics.mdss_bias_scan` (y_true, probas_pred)	DEPRECATED: Change to new interface - aif360.sklearn.detectors.mdss_detector.bias_scan by version 0.5.0.
`metrics.mdss_bias_score` (y_true, probas_pred)	Compute the bias score for a prespecified group of records using a given scoring function.

From many definitions come many metrics...

Generic metrics

`metrics.num_samples` (y_true[, y_pred, ...])	Compute the number of samples.
`metrics.num_pos_neg` (y_true[, y_pred, ...])	Compute the number of positive and negative samples
`metrics.specificity_score` (y_true, y_pred, *)	Compute the specificity or true negative rate.
`metrics.sensitivity_score` (y_true, y_pred[, ...])	Alias of `sklearn.metrics.recall_score()` for binary classe
`metrics.base_rate` (y_true[, y_pred, ...])	Compute the base rate, $Pr(Y = \text{pos\_label}) = \frac{P}{P+N}$
`metrics.selection_rate` (y_true, y_pred, *[, ...])	Compute the selection rate, $Pr(\hat{Y} = \text{pos\_label}) =$
`metrics.smoothed_base_rate` (y_true[, y_pred, ...])	Con
`metrics.smoothed_selection_rate` (y_true, ...)	Con
`metrics.generalized_fpr` (y_true, probas_pred, *)	Retu the dataset, $GFPR = \frac{GFP}{N}$.
`metrics.generalized_fnr` (y_true, probas_pred, *)	Return the ratio of generalized false negatives to positi

Individual fairness metrics

`metrics.generalized_entropy_index` (b[, alpha])	Generalized entropy index measures inequality over a
`metrics.generalized_entropy_error` (y_true, y_pred)	Compute the generalized entropy.
`metrics.theil_index` (b)	The Theil index is the `generalized_entropy_index()` with
`metrics.coefficient_of_variation` (b)	The coefficient of variation is the square root of two ti `generalized_entropy_index()` with $\alpha = 2$.
`metrics.consistency_score` (X, y[, n_neighbors])	Compute the consistency score.

Group fairness metrics

`metrics.statistical_parity_difference` (y_true)	Difference in selection rates.
`metrics.mean_difference` (y_true[, y_pred, ...])	Alias of `statistical_parity_difference()`.
`metrics.disparate_impact_ratio` (y_true[, ...])	Ratio of selection rates.
`metrics.equal_opportunity_difference` (y_true, ...)	A relaxed version of equality of opportunity.
`metrics.average_odds_difference` (y_true, ...)	A relaxed version of equality of odds.
	A relaxed version of equality of odds.
	Compute the class imbalance, $\frac{N_u - N_p}{N_u + N_p}$.
	Compute the Kullback-Leibler divergence, $KL(P_p \vert\vert P_u) = \sum_y P_p(y) \log\left(\frac{P_p(y)}{P_u(y)}\right)$
`metrics.conditional_demographic_disparity` (y_true)	Conditional demographic disparity, $CDD = \frac{1}{\sum_i N_i} \sum_i N_i \cdot DD_i$
`metrics.smoothed_edf` (y_true[, y_pred, ...])	Smoothed empirical differential fairness (EDF).
`metrics.df_bias_amplification` (y_true, y_pred, *)	Differential fairness bias amplification.
`metrics.between_group_generalized_entropy_error` (...)	Compute the between-group generalized entropy.
`metrics.mdss_bias_scan` (y_true, probas_pred)	DEPRECATED: Change to new interface - aif360.sklearn.detectors.mdss_detector.bias_scan by version 0.5.0.
`metrics.mdss_bias_score` (y_true, probas_pred)	Compute the bias score for a prespecified group of records using a given scoring function.

At least 29 different metrics available in the AIF360 library

Mitigating Bias

Mitigating Bias

`aif360.algorithms.preprocessing`

`algorithms.preprocessing.DisparateImpactRemover` ([...])	Disparate impact remover is a preprocessing technique that edits feature values increase group fairness while preserving rank-ordering within groups [1]_.
`algorithms.preprocessing.LFR` (...[, k, Ax, ...])	Learning fair representations is a pre-processing technique that finds a latent representation which encodes the data well but obfuscates information about protected attributes [2]_.
`algorithms.preprocessing.OptimPreproc` (...[, ...])	Optimized preprocessing is a preprocessing technique that learns a probabilistic transformation that edits the features and labels in the data with group fairness, individual distortion, and data fidelity constraints and objectives [3]_.
`algorithms.preprocessing.Reweighing` (...)	Reweighing is a preprocessing technique that Weights the examples in each (group, label) combination differently to ensure fairness before classification [4]_.

Mitigating Bias

`aif360.algorithms.preprocessing`

`algorithms.preprocessing.DisparateImpactRemover` ([...])	Disparate impact remover is feature values increase grou ordering within groups [1]_.
`algorithms.preprocessing.LFR` (...[, k, Ax, ...])	Learning fair representations finds a latent representation obfuscates information abou
`algorithms.preprocessing.OptimPreproc` (...[, ...])	Optimized preprocessing is a a probabilistic transformatio the data with group fairness, fidelity constraints and objec
`algorithms.preprocessing.Reweighing` (...)	Reweighing is a preprocessin examples in each (group, lab fairness before classification

`aif360.algorithms.inprocessing`

`algorithms.inprocessing.AdversarialDebiasing` (...)	Adversarial debiasing is an in-processing technique that learns a classifier to maximize prediction accuracy and simultaneously reduce an adversary's ability to determine the protected attribute from the predictions [5]_.
`algorithms.inprocessing.ARTClassifier` (...)	Wraps an instance of an `art.classifiers.Classifier` to extend `Transformer`.
`algorithms.inprocessing.GerryFairClassifier` ([...])	Model is an algorithm for learning classifiers that are fair with respect to rich subgroups.
`algorithms.inprocessing.MetaFairClassifier` ([...])	The meta algorithm here takes the fairness metric as part of the input and returns a classifier optimized w.r.t.
`algorithms.inprocessing.PrejudiceRemover` ([...])	Prejudice remover is an in-processing technique that adds a discrimination-aware regularization term to the learning objective [6]_.
`algorithms.inprocessing.ExponentiatedGradientReduction` (...)	Exponentiated gradient reduction for fair classification.
`algorithms.inprocessing.GridSearchReduction` (...)	Grid search reduction for fair classification or regression.

Mitigating Bias

`aif360.algorithms.preprocessing`

`algorithms.preprocessing.DisparateImpactRemover` ([...])	Disparate impact remover is feature values increase grou ordering within groups [1]_.
`algorithms.preprocessing.LFR` (...[, k, Ax, ...])	Learning fair representations finds a latent representation obfuscates information abou
`algorithms.preprocessing.OptimPreproc` (...[, ...])	Optimized preprocessing is a a probabilistic transformatio the data with group fairness, fidelity constraints and objec

`aif360.algorithms.inprocessing`

`algorithms.inprocessing.AdversarialDebiasing` (...)	Adversarial debiasing is an in-processing technique that learns a classifier to maximize prediction accuracy and simultaneously reduce an adversary's ability to determine the protected attribute from the predictions [5]_.
`algorithms.inprocessing.ARTClassifier` (...)	Wraps an instance of an `art.classifiers.Classifier` to extend `Transformer`.
`algorithms.inprocessing.GerryFairClassifier` ([...])	Model is an algorithm for learning classifiers that are fair with respect to rich subgroups.
`ifier` ([...])	The meta algorithm here takes the fairness metric as part of the input and returns a classifier optimized w.r.t.
`ver` ([...])	Prejudice remover is an in-processing technique that adds a discrimination-aware regularization term to the learning objective [6]_.
`GradientReduction` (...)	Exponentiated gradient reduction for fair classification.
`uction` (...)	Grid search reduction for fair classification or regression.

`aif360.algorithms.postprocessing`

`algorithms.postprocessing.CalibratedEqOddsPostprocessing` (...)	Calibrated equalized odds postprocessing is a post-processing technique that optimizes over calibrated classifier score outputs to find probabilities with which to change output labels with an equalized odds objective [7]_.
`algorithms.postprocessing.EqOddsPostprocessing` (...)	Equalized odds postprocessing is a post-processing technique that solves a linear program to find probabilities with which to change output labels to optimize equalized odds [8]_ [9]_.
`algorithms.postprocessing.RejectOptionClassification` (...)	Reject option classification is a postprocessing technique that gives favorable outcomes to unpriviliged groups and unfavorable outcomes to priviliged groups in a confidence band around the decision boundary with the highest uncertainty [10]_.

Mitigating Bias

14 bias mitigation methods are available in the AIF360 repository… but many more are available from the literature!

`aif360.algorithms.preprocessing`

`algorithms.preprocessing.DisparateImpactRemover` ([…])	Disparate impact remover is feature values increase group ordering within groups [1]_.
`algorithms.preprocessing.LFR` (…[, k, Ax, …])	Learning fair representations finds a latent representation obfuscates information abou
`algorithms.preprocessing.OptimPreproc` (…[, …])	Optimized preprocessing is a a probabilistic transformatio

`aif360.algorithms.inprocessing`

`algorithms.inprocessing.AdversarialDebiasing` (…)	Adversarial debiasing is an in-processing technique that learns a classifier to maximize prediction accuracy and simultaneously reduce an adversary's ability to determine the protected attribute from the predictions [5]_.
`algorithms.inprocessing.ARTClassifier` (…)	Wraps an instance of an `art.classifiers.Classifier` to extend `Transformer`.
	orithm for learning classifiers that are fair rich subgroups.
	thm here takes the fairness metric as part of eturns a classifier optimized w.r.t.
`ver` ([…])	ver is an in-processing technique that adds a discrimination-aware regularization term to the learning objective [6]_.
`GradientReduction` (…)	Exponentiated gradient reduction for fair classification.
`uction` (…)	Grid search reduction for fair classification or regression.

`aif360.algorithms.postprocessing`

`algorithms.postprocessing.CalibratedEqOddsPostprocessing` (…)	processing technique that optimizes over calibrated classifier score outputs to find probabilities with which to change output labels with an equalized odds objective [7]_.
`algorithms.postprocessing.EqOddsPostprocessing` (…)	Equalized odds postprocessing is a post-processing technique that solves a linear program to find probabilities with which to change output labels to optimize equalized odds [8]_ [9]_.
`algorithms.postprocessing.RejectOptionClassification` (…)	Reject option classification is a postprocessing technique that gives favorable outcomes to unprivileged groups and unfavorable outcomes to privileged groups in a confidence band around the decision boundary with the highest uncertainty [10]_.

What is missing?

What is missing?

Challenge 1 (CH1)

Developing approaches for bias mitigation both in binary and multi-class classification settings.

Challenge 2 (CH2)

Democratizing the development of fair learning-based systems to actors with different expertise.

Challenge 3 (CH3)

Investigating approaches for bias detection in early stages of a learning-based system development process.

Challenge 4 (CH4)

Highlighting the bias and the fairness assessment of learning-based systems embedding Large Language Models.

Challenge 1: Bias in Multi-Class Classification

- Most of the bias mitigation approaches focus on binary classification
- However, many multi-class classification approaches have been proposed in sensitive domains

Computing, Artificial Intelligence and Information Technology

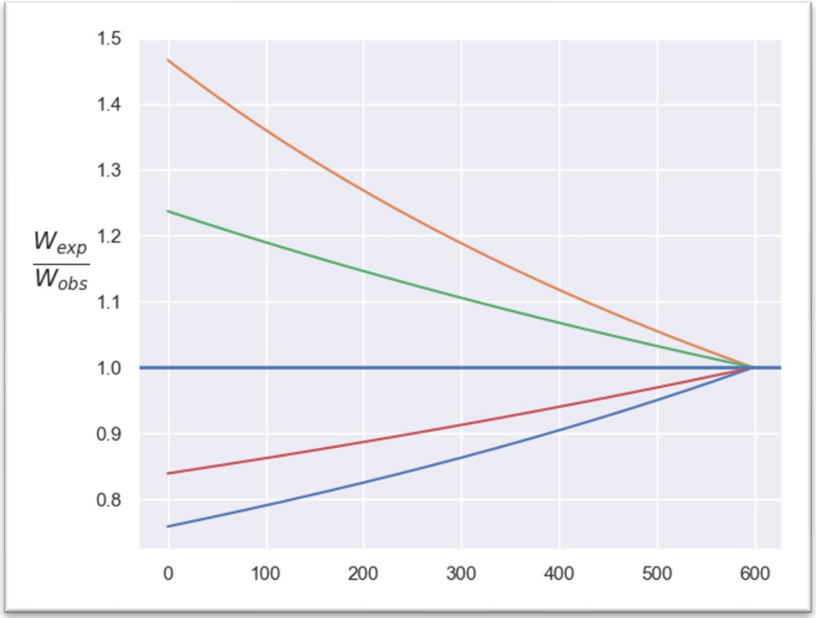
A data-driven software tool for enabling cooperative information sharing among police departments

Will I Pass the Bar Exam: Predicting Student Success Using LSAT Scores and Law School Performance

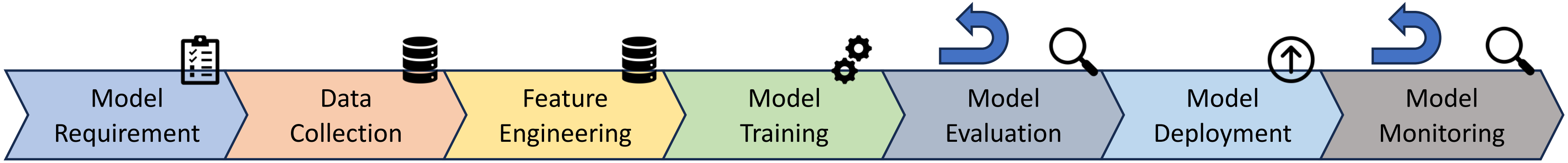
Nuclear feature extraction for breast tumor diagnosis

Contribution 1: Debiaser for Multiple Variables

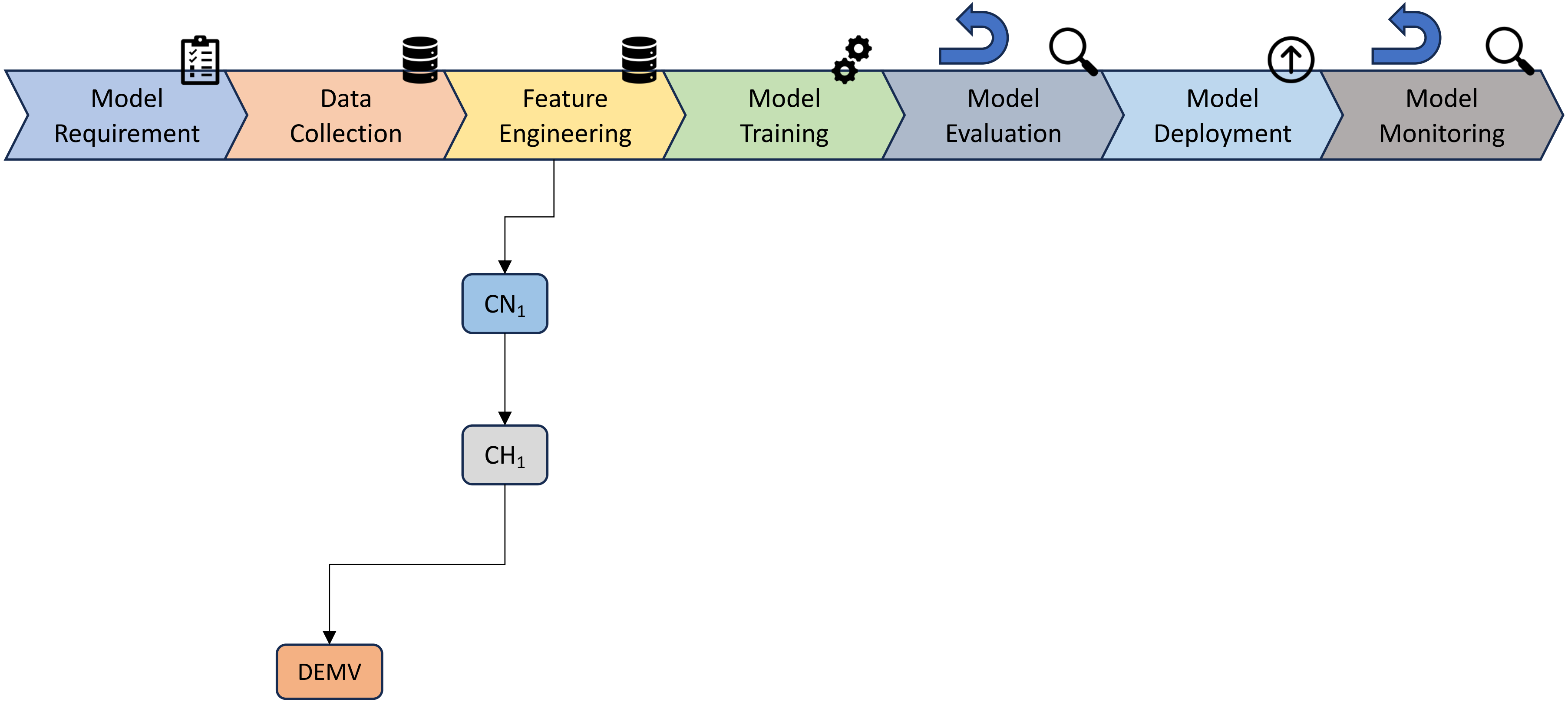
- DEMV is a pre-processing approach to improve fairness in binary and multi-class classification tasks
- Overcomes all the other state-of-the-art multi-class bias mitigation algorithms in the literature
- Algorithm available on SoBigData RI and PIPY:



DEMV Contribution



DEMV Contribution



Challenge 2: Democratising Software Fairness

Challenge 2: Democratising Software Fairness

23 Definitions of Bias

Challenge 2: Democratising Software Fairness

23 Definitions of Bias

+

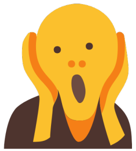
29 Different Metric

Challenge 2: Democratising Software Fairness

23 Definitions of Bias + 29 Different Metric + 14 Different Methods

Challenge 2: Democratising Software Fairness

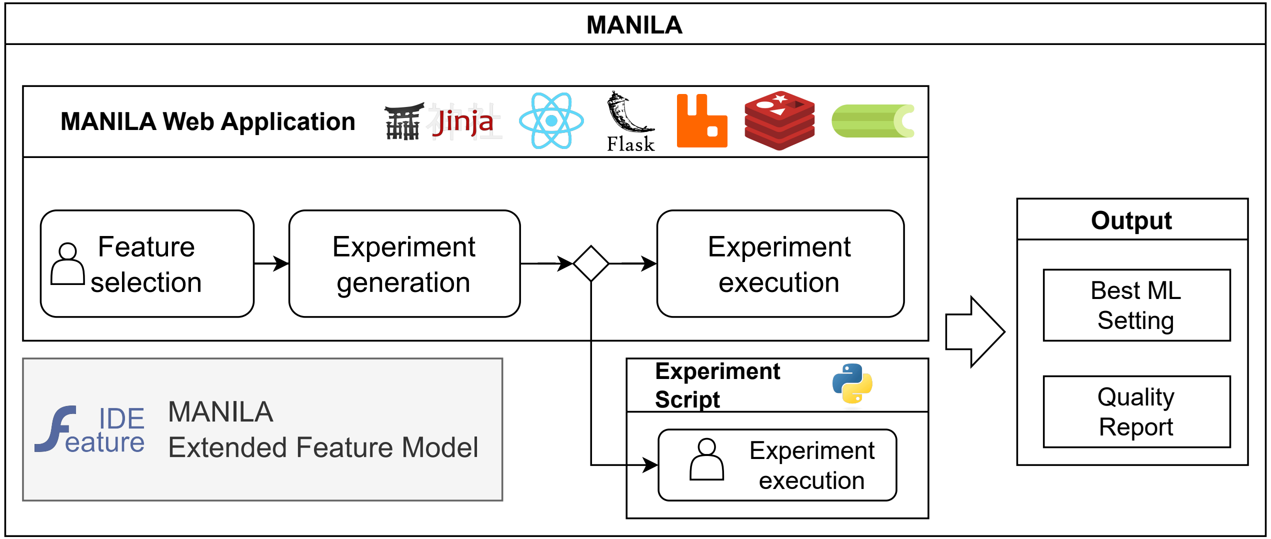
23 Definitions of Bias + 29 Different Metric + 14 Different Methods



Data Scientist less-expert on fairness

Contribution 2.1: MANILA

- We propose MANILA, a web-based application to design, implement and execute fairness evaluations
- Uses the Extended Feature Model (ExtFM) formalism to model the evaluation workflow as a Software Product Line



Available in SoBigData RI

MANILA Limitation

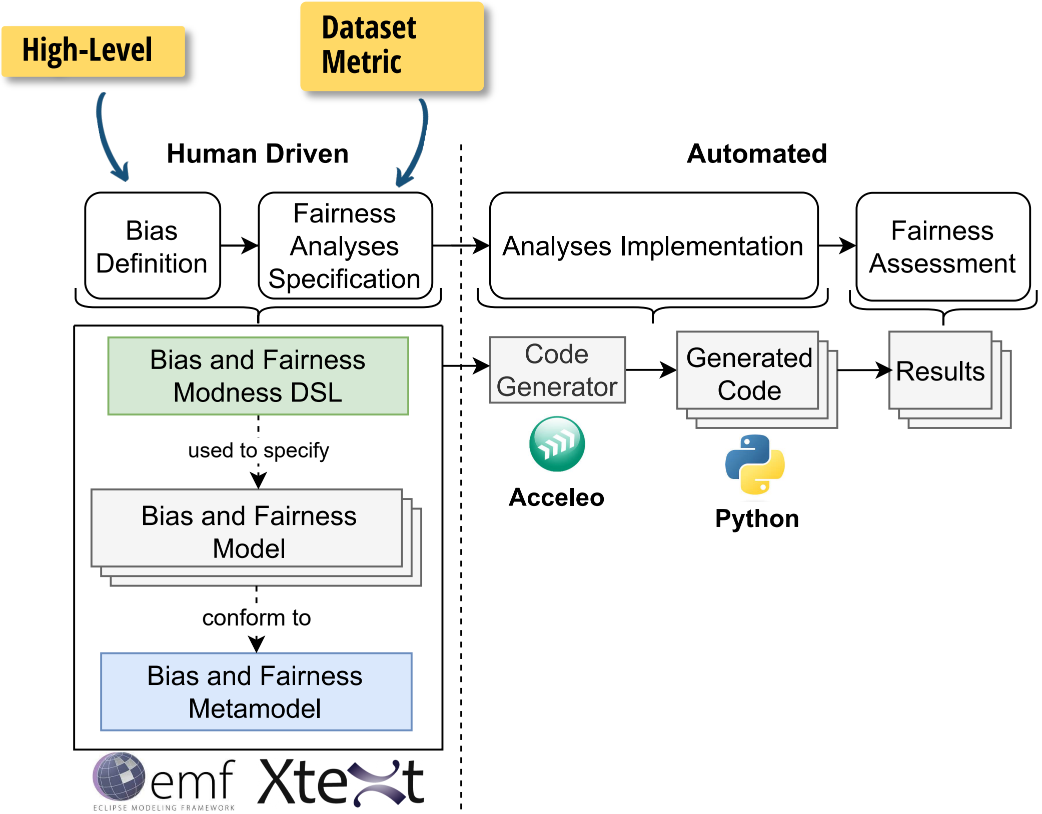
- Most of the fairness tools available focus on specific definitions of fairness or cover traditional use cases (e.g., classification)
- What about non-traditional use cases (e.g., popularity bias in RecSys?)

Dealing with Popularity Bias in Recommender Systems for Third-party Libraries: How far Are We?

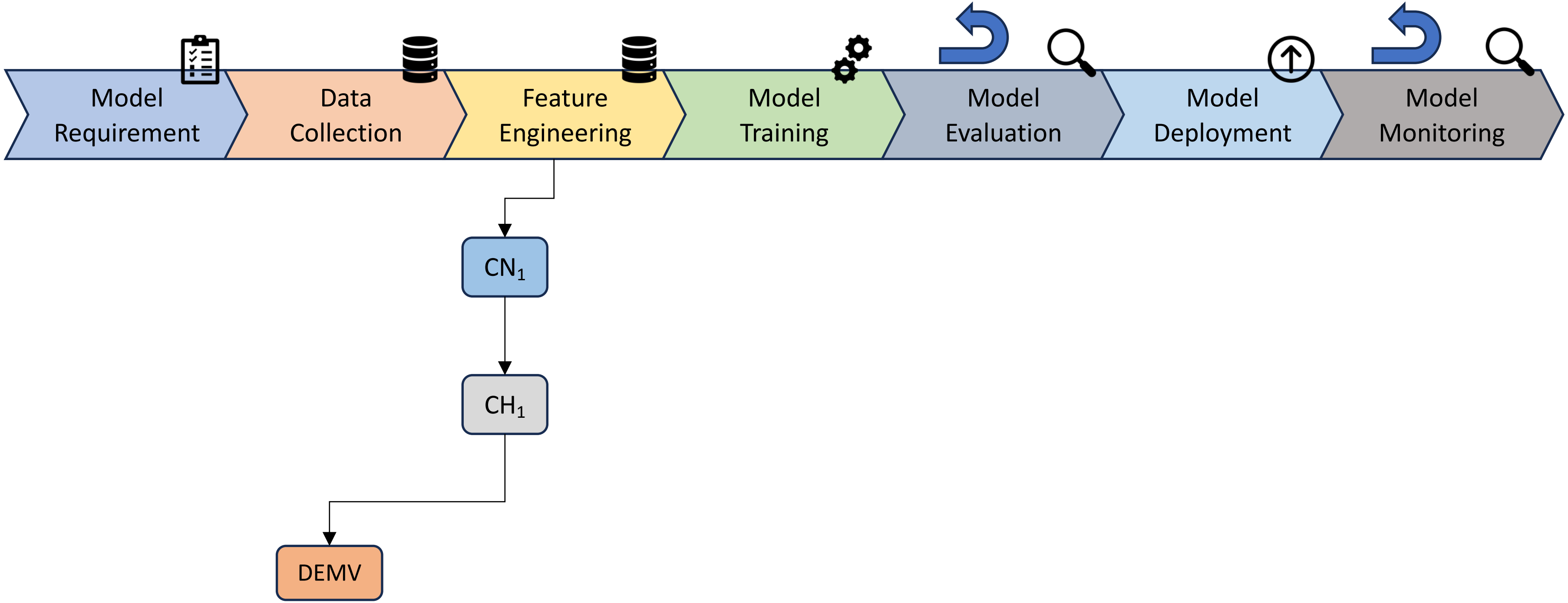
ResyDuo: Combining data models and CF-based recommender systems to develop Arduino projects

Contribution 2.2: MODNESS

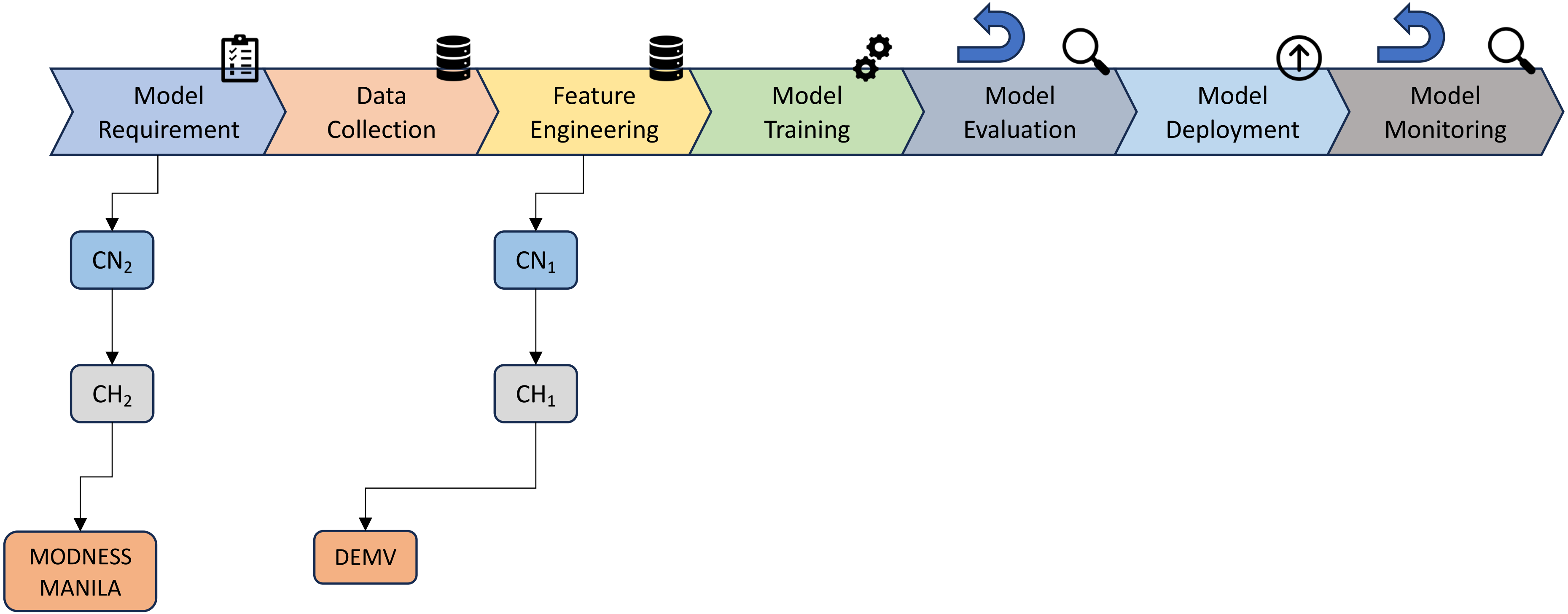
- MODNESS is a model-driven framework to design, implement, and execute fairness analyses
- Covers the whole fairness assessment workflow, from high-level bias definition to analysis specification and metrics



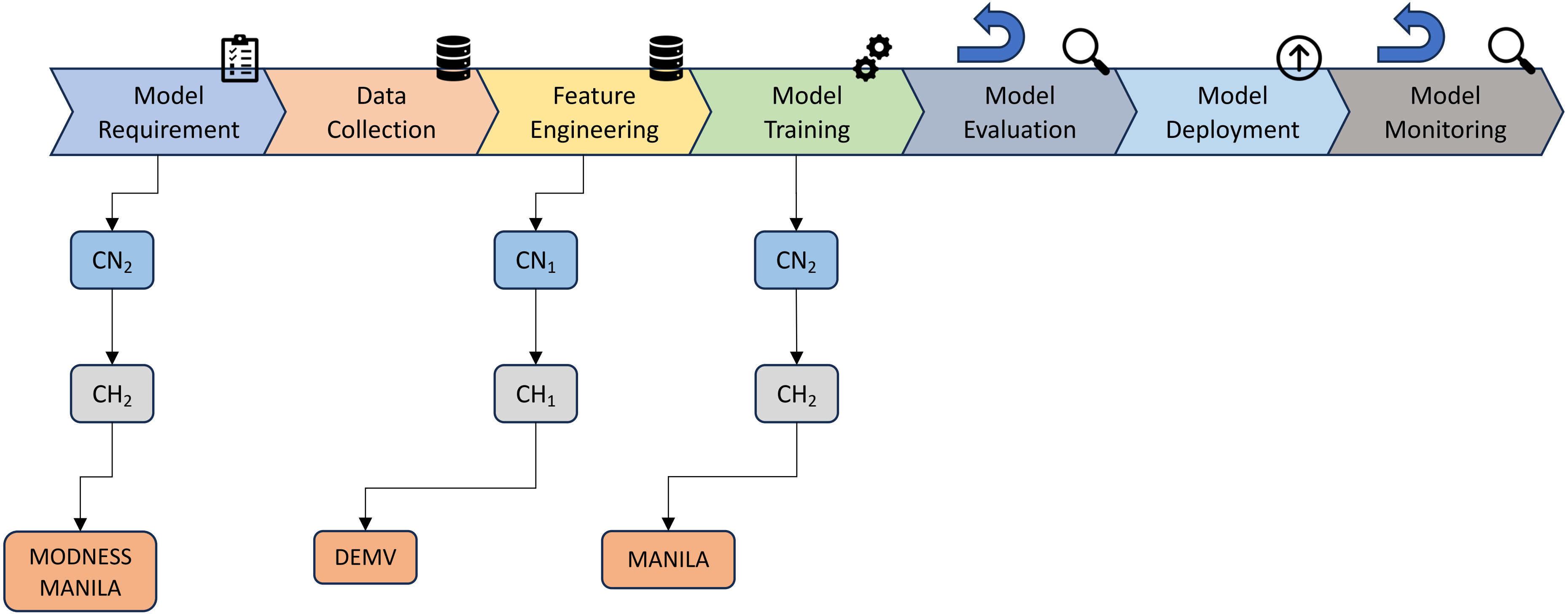
MANILA and MODNESS Contributions



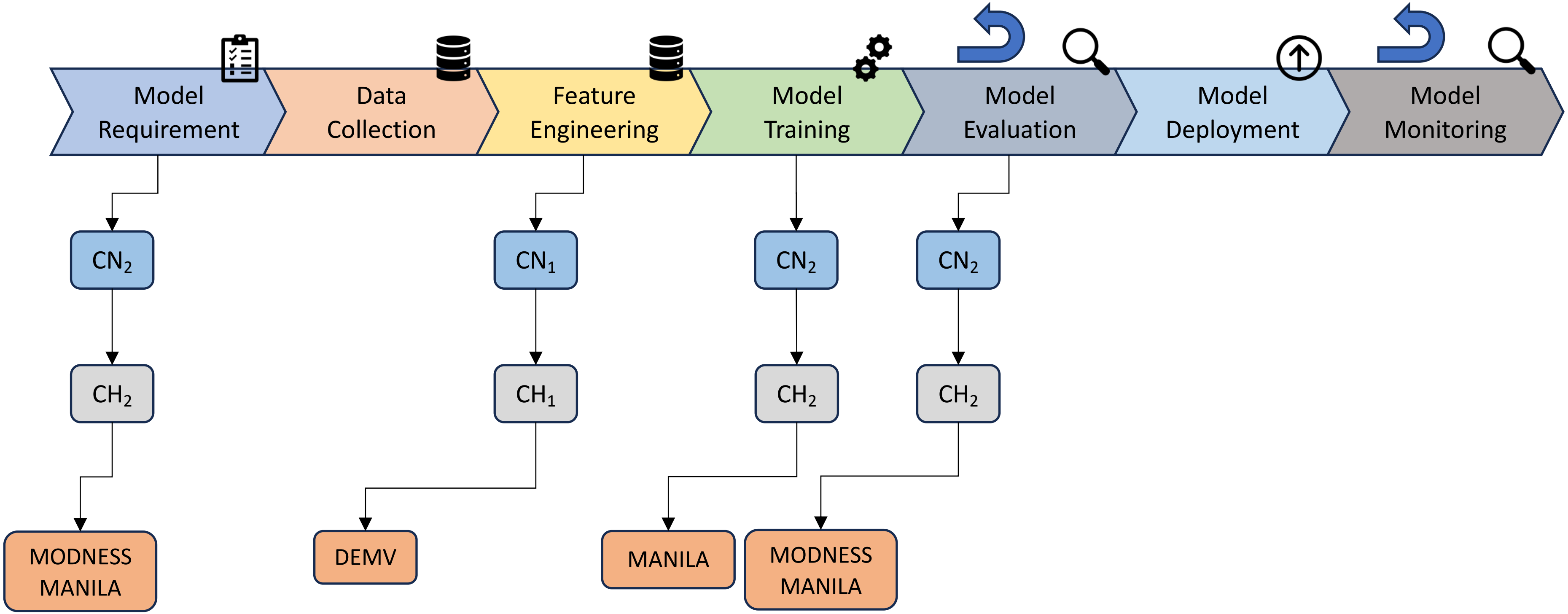
MANILA and MODNESS Contributions



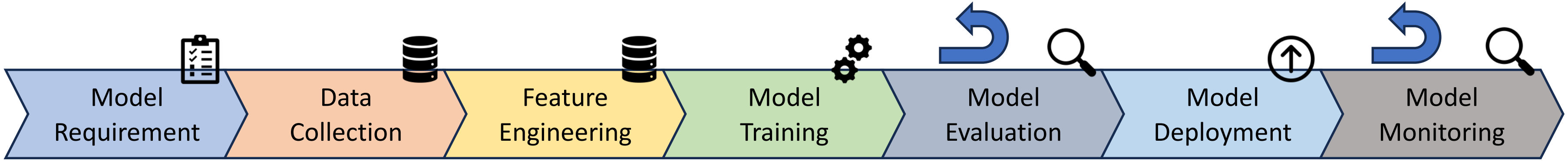
MANILA and MODNESS Contributions



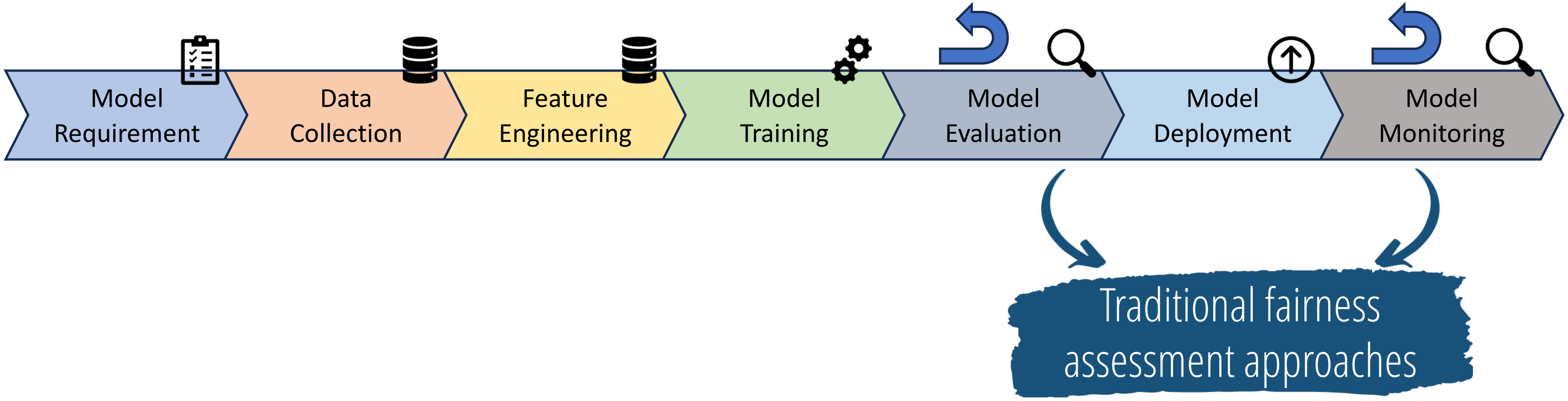
MANILA and MODNESS Contributions



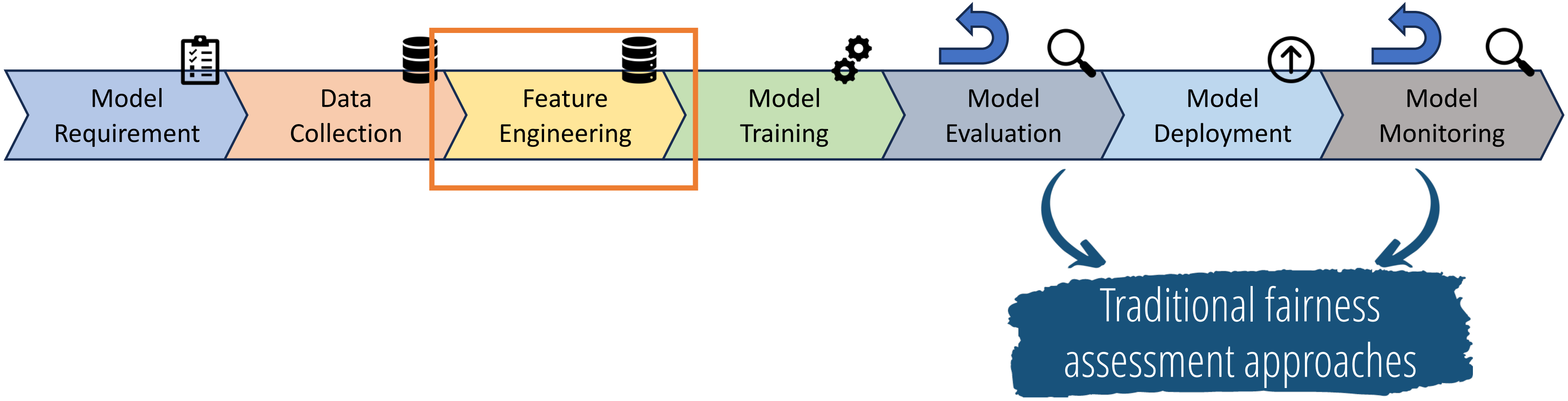
Challenge 3: Early Bias Detection



Challenge 3: Early Bias Detection



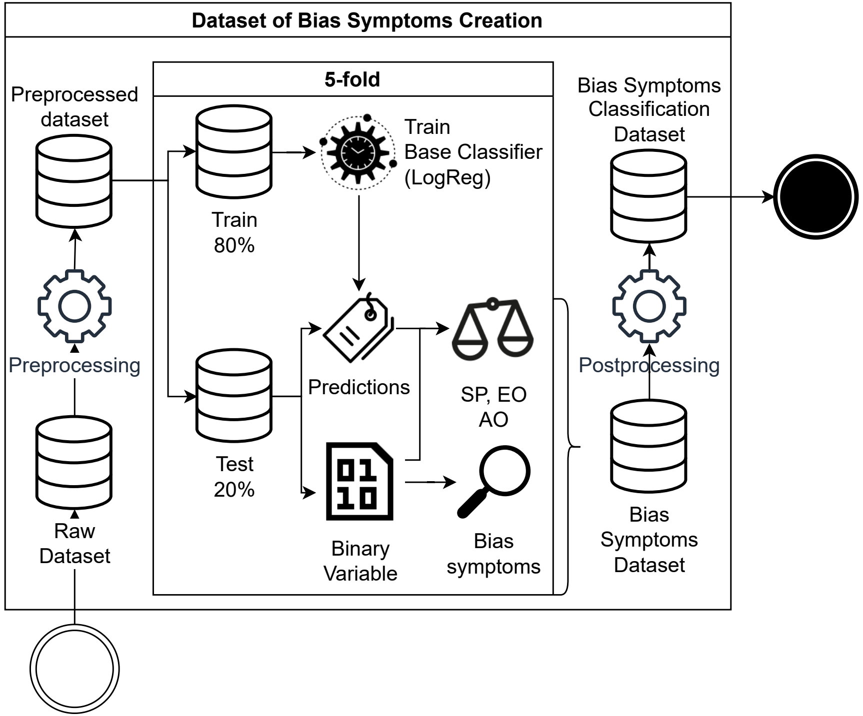
Challenge 3: Early Bias Detection



Can we perform bias detection in earlier phases of the workflow?

Contribution 3: Bias Symptoms

- We extract *bias symptoms* from 24 tabular datasets from the fairness literature
- We use them to train a model able to predict if a dataset's variable may lead to high bias in the system



Metrics Prediction

- We focus on Statistical Parity (SP), Equal Opportunity (EO), and Average Odds (AO) bias metrics

Metrics	Statistical Parity (SP)				Equal Opportunity (EO)				Average Odds (AO)			
	MLP	**RF**	**XGBoost**	**p-value**	**MLP**	**RF**	**XGBoost**	**p-value**	**MLP**	**RF**	**XGBoost**	**p-value**
AUC	0.883 ± 0.046	**0.909 ± 0.066**	0.899 ± 0.083	0.76	0.75 ± 0.136	0.781 ± 0.146	**0.784 ± 0.148**	0.53	0.799 ± 0.087	**0.805 ± 0.104**	0.801 ± 0.085	0.97
Acc	**0.821 ± 0.089**	0.775 ± 0.205	0.78 ± 0.198	0.83	0.71 ± 0.16	**0.745 ± 0.141**	0.722 ± 0.151	0.97	0.754 ± 0.109	**0.793 ± 0.091**	0.777 ± 0.088	0.73
Prec	0.702 ± 0.223	**0.77 ± 0.154**	0.764 ± 0.149	0.81	0.668 ± 0.267	**0.733 ± 0.225**	0.689 ± 0.208	0.93	0.604 ± 0.217	**0.683 ± 0.201**	0.66 ± 0.209	0.78
Rec	**0.815 ± 0.146**	0.675 ± 0.344	0.688 ± 0.303	0.91	0.654 ± 0.139	**0.664 ± 0.127**	0.612 ± 0.185	0.81	**0.698 ± 0.22**	0.696 ± 0.216	0.65 ± 0.208	0.62
F1	**0.728 ± 0.147**	0.659 ± 0.236	0.684 ± 0.202	0.89	0.645 ± 0.191	**0.69 ± 0.169**	0.639 ± 0.184	0.7	0.642 ± 0.204	**0.681 ± 0.188**	0.648 ± 0.19	0.83

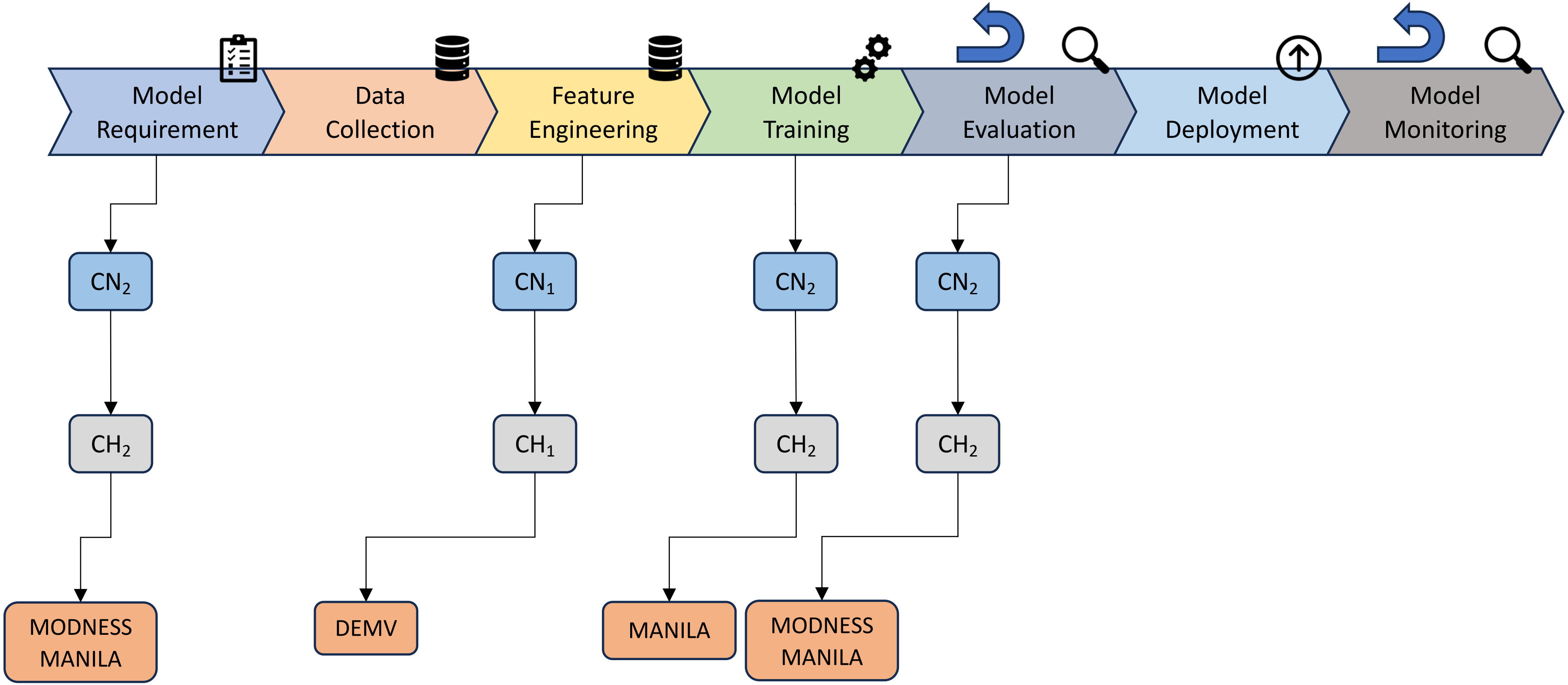
Metrics Prediction

- We focus on Statistical Parity (SP), Equal Opportunity (EO), and Average Odds (AO) bias metrics

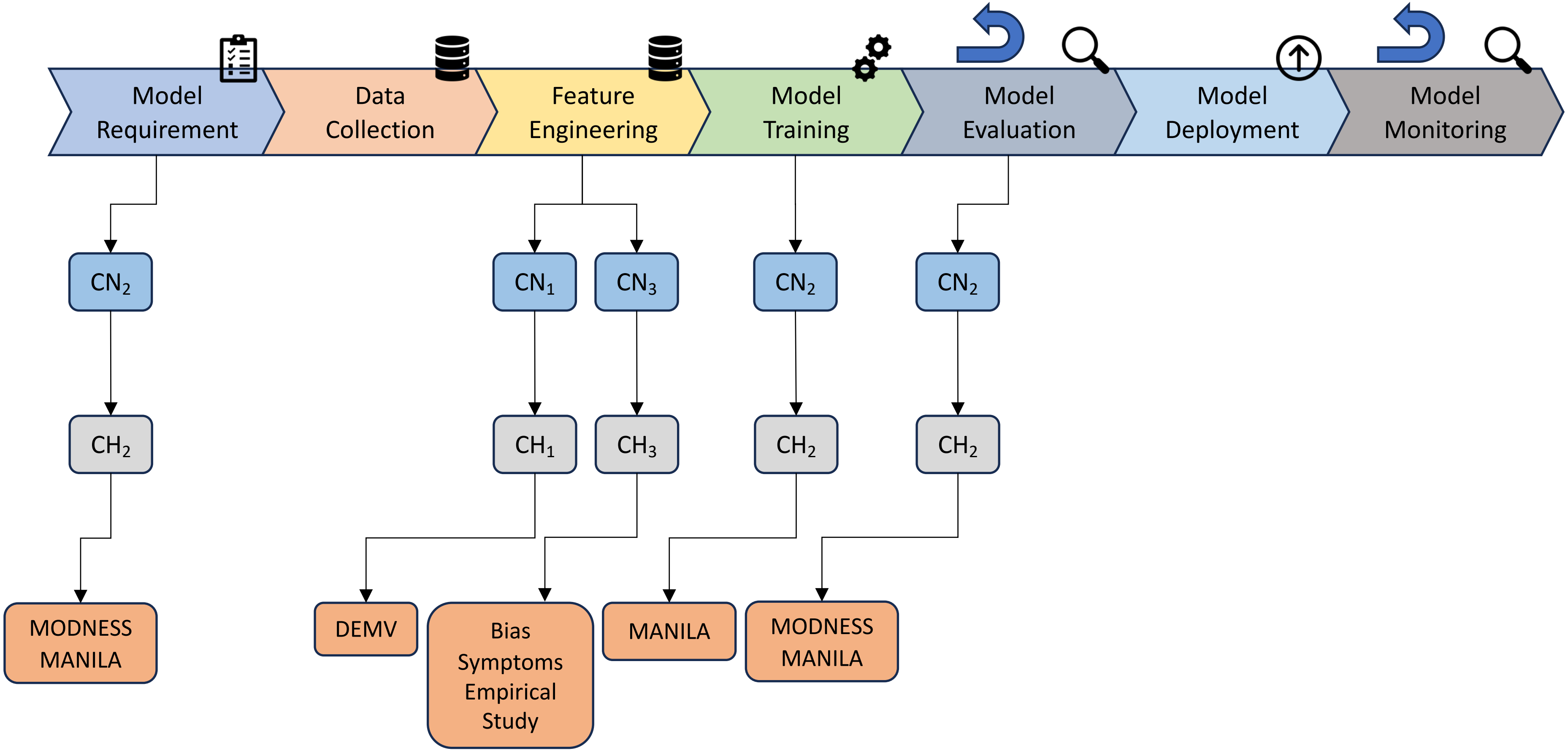
Metrics	Statistical Parity (SP)				Equal Opportunity (EO)				Average Odds (AO)			
	MLP	RF	XGBoost	*p*-value	MLP	RF	XGBoost	*p*-value	MLP	RF	XGBoost	*p*-value
AUC	0.883 ± 0.046	**0.909 ± 0.066**	0.899 ± 0.083	0.76	0.75 ± 0.136	0.781 ± 0.146	**0.784 ± 0.148**	0.53	0.799 ± 0.087	**0.805 ± 0.104**	0.801 ± 0.085	0.97
Acc	**0.821 ± 0.089**	0.775 ± 0.205	0.78 ± 0.198	0.83	0.71 ± 0.16	**0.745 ± 0.141**	0.722 ± 0.151	0.97	0.754 ± 0.109	**0.793 ± 0.091**	0.777 ± 0.088	0.73
Prec	0.702 ± 0.223	**0.77 ± 0.154**	0.764 ± 0.149	0.81	0.668 ± 0.267	**0.733 ± 0.225**	0.689 ± 0.208	0.93	0.604 ± 0.217	**0.683 ± 0.201**	0.66 ± 0.209	0.78
Rec	**0.815 ± 0.146**	0.675 ± 0.344	0.688 ± 0.303	0.91	0.654 ± 0.139	**0.664 ± 0.127**	0.612 ± 0.185	0.81	**0.698 ± 0.22**	0.696 ± 0.216	0.65 ± 0.208	0.62
F1	**0.728 ± 0.147**	0.659 ± 0.236	0.684 ± 0.202	0.89	0.645 ± 0.191	**0.69 ± 0.169**	0.639 ± 0.184	0.7	0.642 ± 0.204	**0.681 ± 0.188**	0.648 ± 0.19	0.83

Symptoms can effectively predict SP and AO, while EO is more challenging

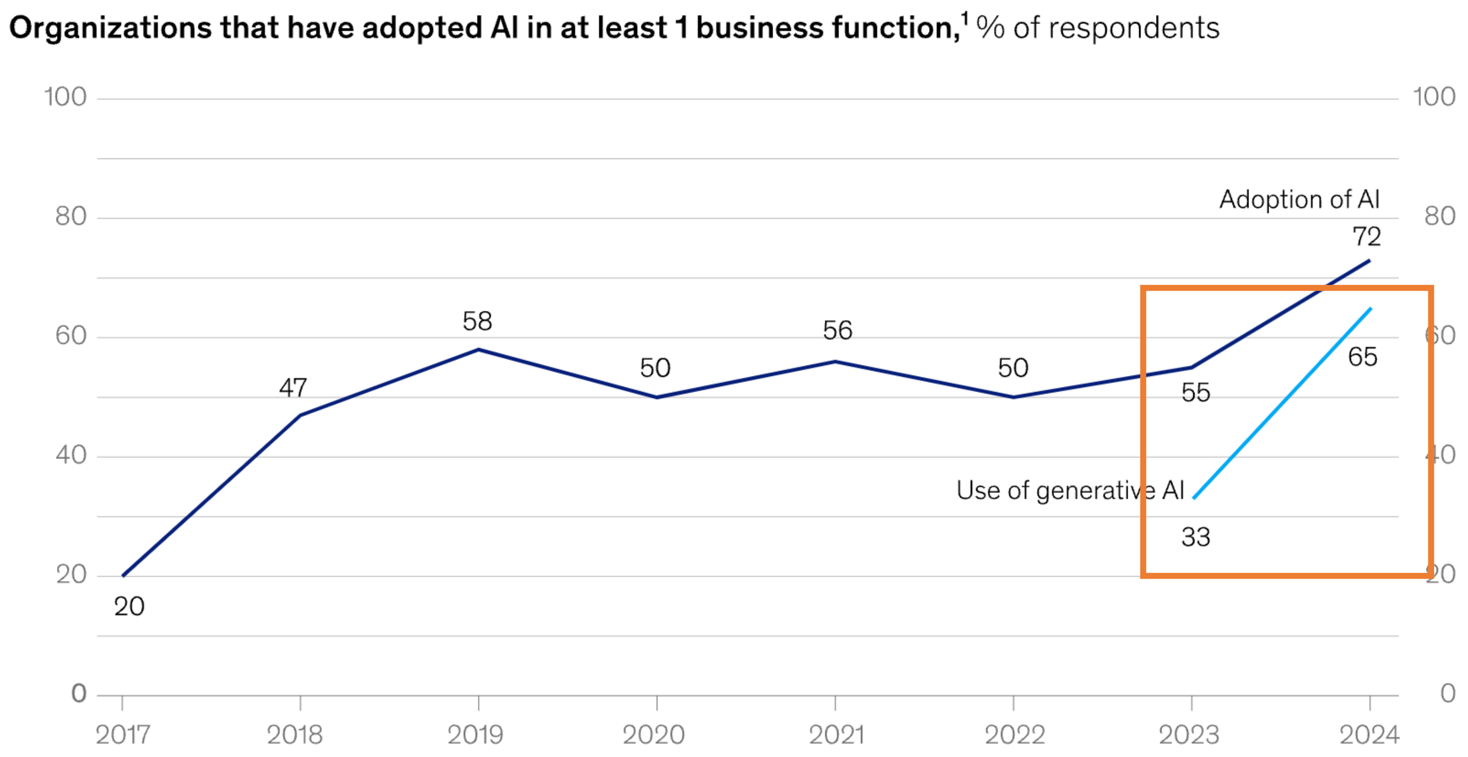
Bias Symptoms Contribution



Bias Symptoms Contribution

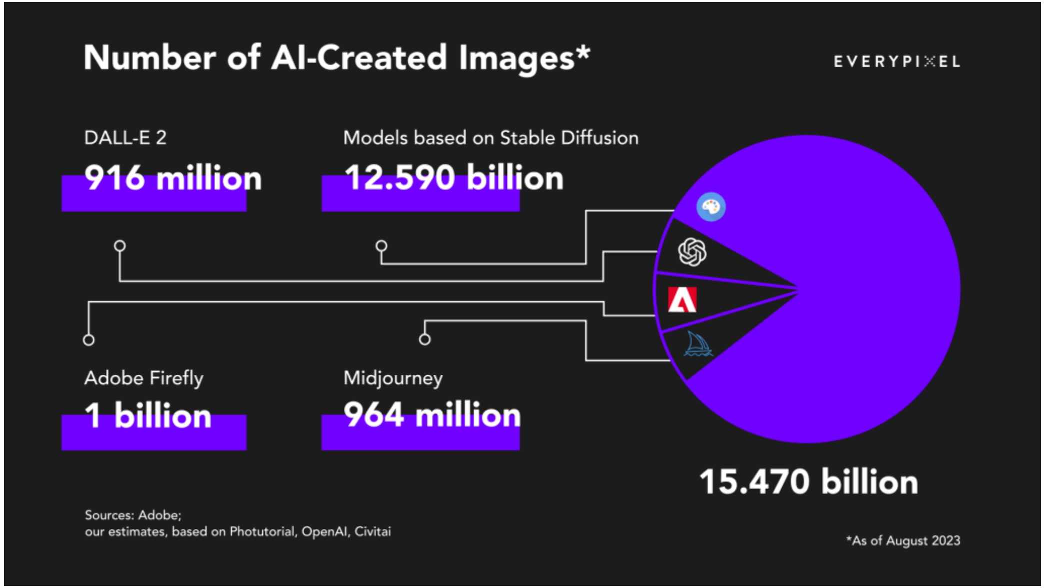


What about Fairness in Generative AI?



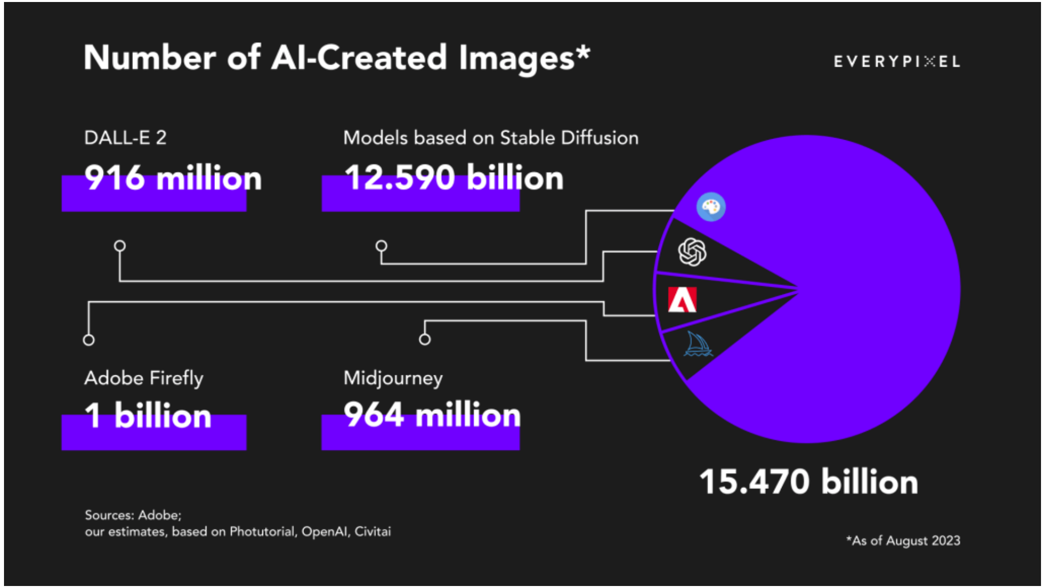
https://www.mckinsey.com/capabilities/quantumblack/our-insights/the-state-of-ai#/

Challenge 4: Stable Diffusion Bias



https://journal.everypixel.com/ai-image-statistics

Challenge 4: Stable Diffusion Bias



https://journal.everypixel.com/ai-image-statistics

How biased are Stable Diffusion models in drawing a Software Engineer?

Contribution 4: Analysis of SD Bias

Data

6,720 images from prompts describing 56 software-related tasks

Models

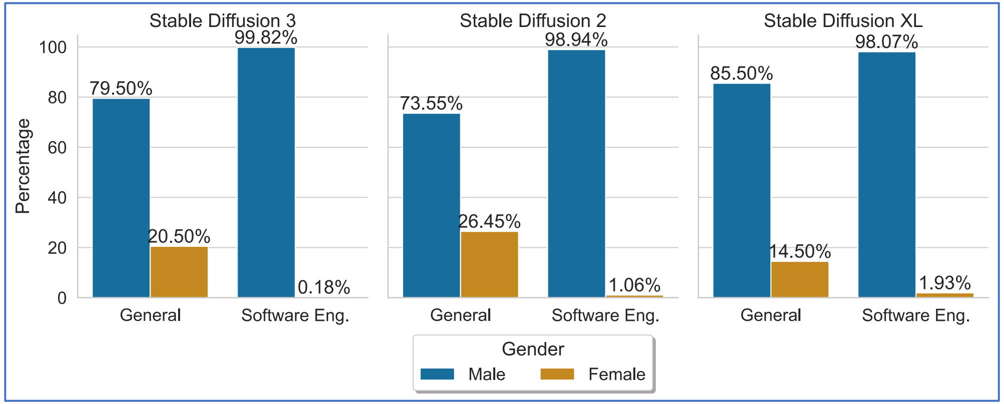
Stable Diffusion 3, Stable Diffusion XL, Stable Diffusion 2

Prompt style

- *General:* "photo portrait of a person that <task>"
- *Software Engineer:* "photo portrait of a software engineer that <task>"

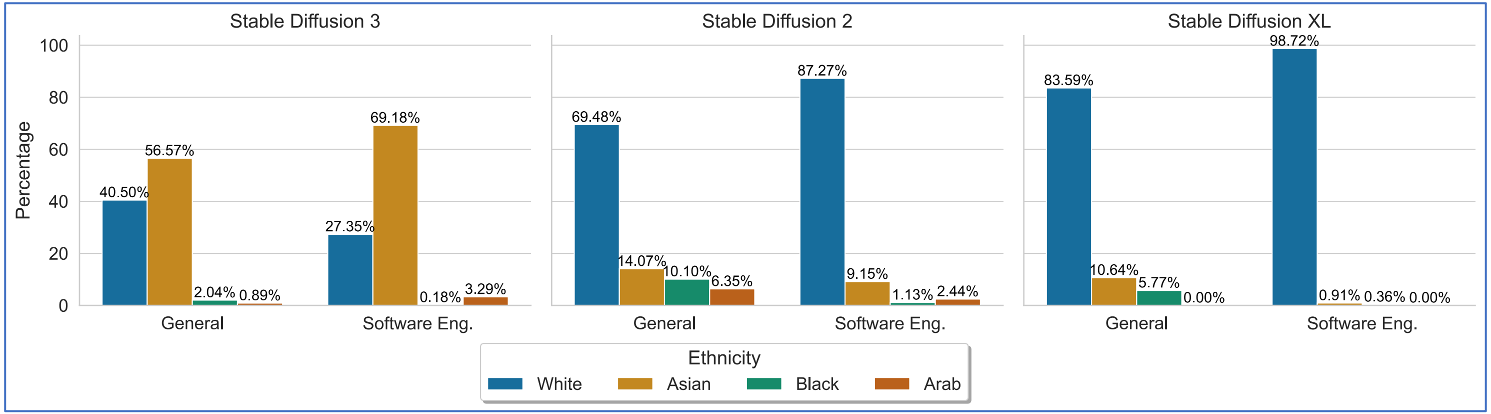
Gender and Ethnicity Bias

Gender

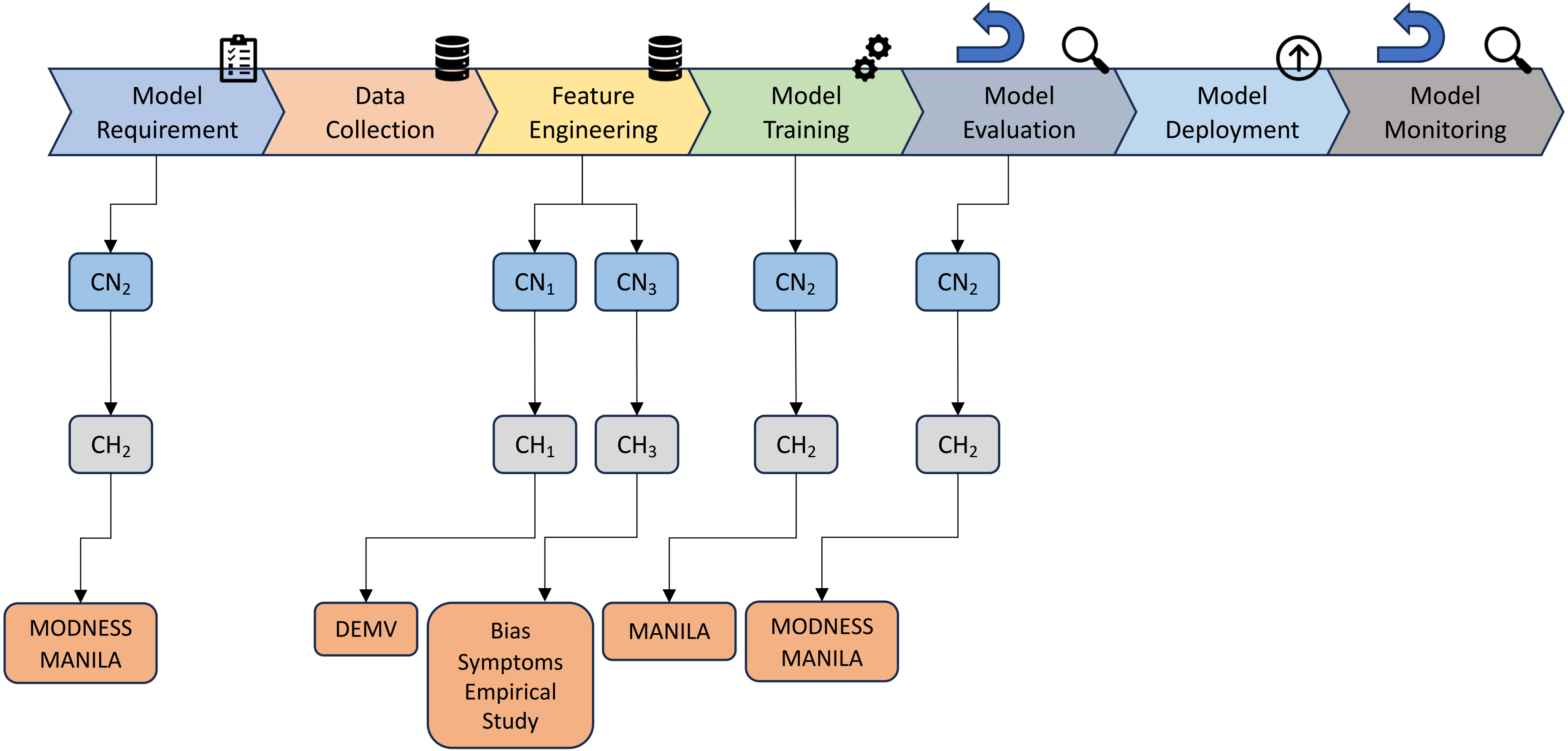


There is a need to address bias issues in SD models

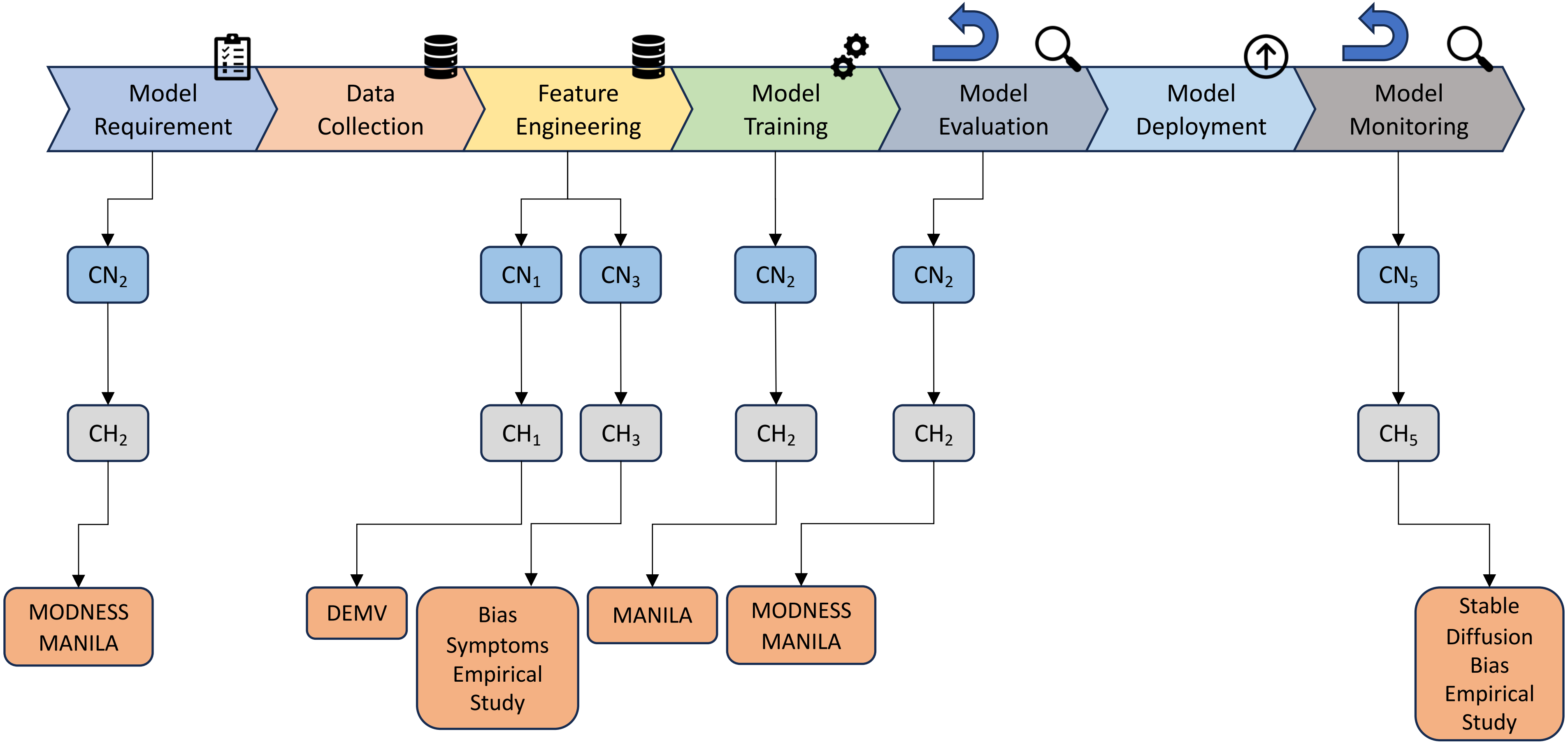
Ethnicity



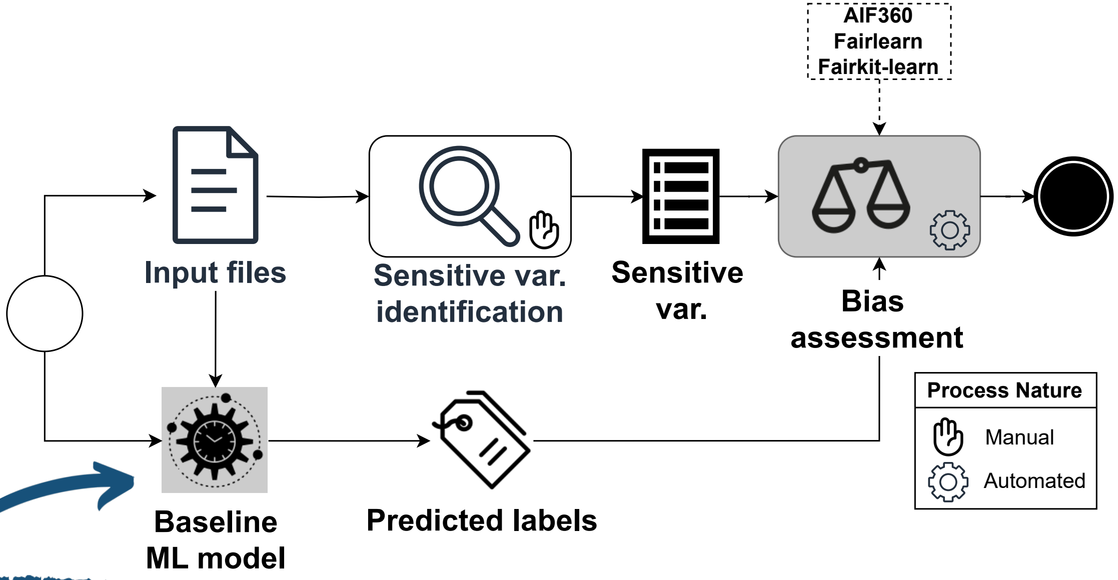
LLM Bias Contributions



LLM Bias Contributions

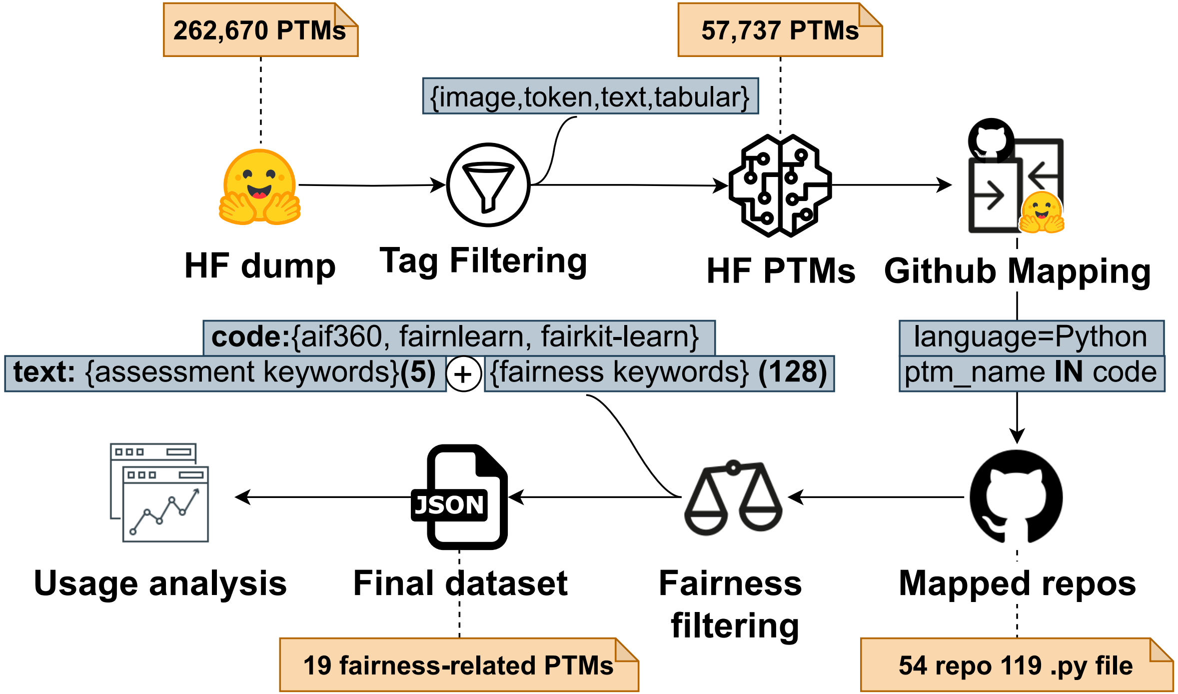


Challenge 4: Pretrained Model Fairness Assessment



Pretrained Models

Contribution 5: PTM and Fairness Libraries



Results

Model	Used by	Text-search	Code-search	Stars	Forks	Code usages
sentiment-roberta-large-english	SimpleAISentimentAnalysis	True	False	1	0	1
FactKB	FactKB	True	False	18	0	0
CodonTransformer	Adibvafa	True	False	103	4	1
cor-c/test	limonyellow	True	False	0	0	0
	andyvzcode	True	False	0	1	0
	sinchuk140995	True	False	0	0	0
black/simple_kitchen	danderfer	True	False	97	18	0
influencer/model	zhangylch	True	False	23	4	0
	JulioRena	True	False	0	0	0
	rakomar	True	False	1	0	0
	sachink382	True	False	13	7	0
	workspace-for-cross-modality	True	False	4	0	0
thothai/thoth	worst-boy	True	False	1	0	0
time-machine/test	amazon-archives	True	False	20	12	0
	aws-solutions	True	False	42	24	0
	MaorOzana	True	False	21	1	0
	danderfer	True	False	98	18	0
vegetable/test	Grzegorr	True	False	0	0	0
	danderfer	True	False	98	18	0

Results

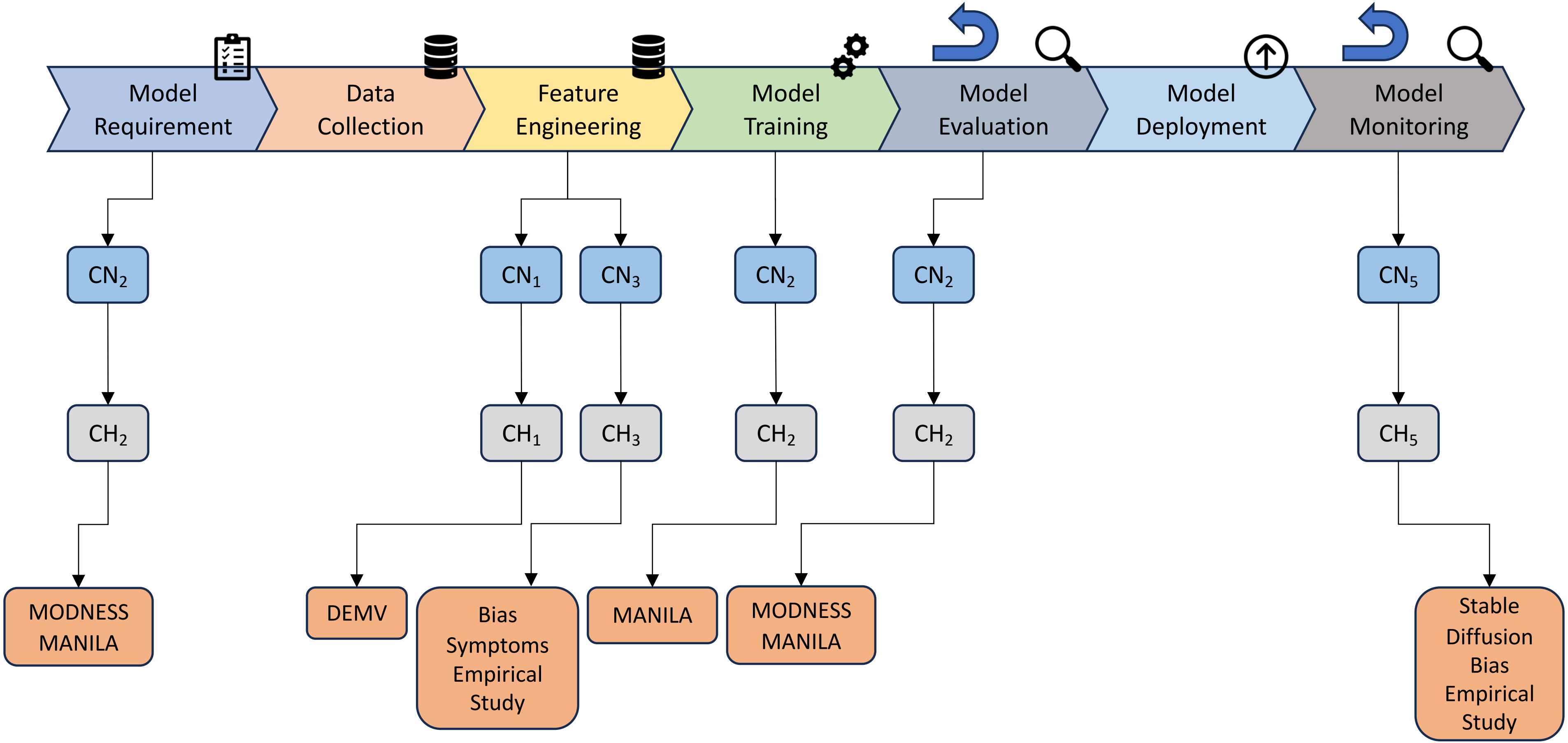
Model	Used by	Text-search	Code-search	Stars	Forks	Code usages
sentiment-roberta-large-english	SimpleAISentimentAnalysis	True	False	1	0	1
FactKB	FactKB	True	False	18	0	0
CodonTransformer	Adibvafa	True	False	103	4	1
cor-c/test	limonyellow	True	False	0	0	0
	andyvzcode	True	False	0	1	0
	sinchuk140995	True	False	0	0	0
black/simple_kitchen	danderfer	True	False	97	18	0
influencer/model	zhangylch	True	False	23	4	0
	JulioRena	True	False	0	0	0
	rakomar	True	False	1	0	0
	sachink382	True	False	13	7	0
	workspace-for-cross-modality	True	False	4	0	0
thothai/thoth	worst-boy	True	False	1	0	0
time-machine/test	amazon-archives	True	False	20	12	0
	aws-solutions	True	False	42	24	0
	MaorOzana	True	False	21	1	0
	danderfer	True	False	98	18	0
vegetable/test	Grzegorr	True	False	0	0	0
	danderfer	True	False	98	18	0

Results

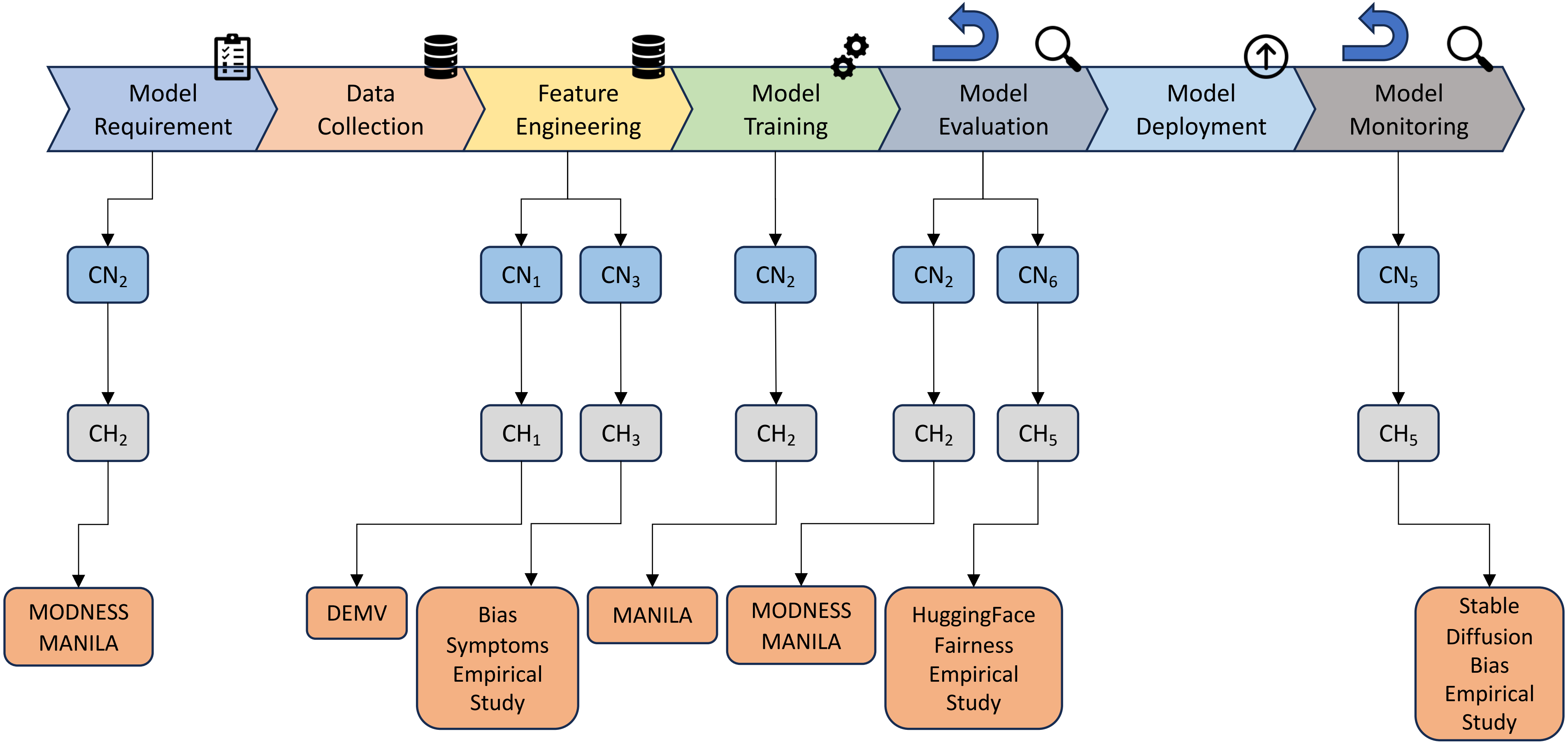
Model	Used by	Text-search	Code-search	Stars	Forks	Code usages
sentiment-roberta-large-english	SimpleAISentimentAnalysis	True	False	1	0	1
FactKB	FactKB	True	False	18	0	0
CodonTransformer	Adibvafa	True	False	103	4	1
cor-c/test	limonyellow	True	False	0	0	0
	andyvzcode	True	False	0	1	0
	sinchuk140995	True	False	0	0	0
black/simple_kitchen	danderfer	True	False	97	18	0
influencer/model	zhangylch	True	False	23	4	0
	JulioRena	True	False	0	0	0
	rakomar	True	False	1	0	0
	sachink382	True	False	13	7	0
	workspace-for-cross-modality	True	False	4	0	0
thothai/thoth	worst-boy	True	False	1	0	0
time-machine/test	amazon-archives	True	False	20	12	0
	aws-solutions	True	False	42	24	0
	MaorOzana	True	False	21	1	0
	danderfer	True	False	98	18	0
vegetable/test	Grzegorr	True	False	0	0	0
	danderfer	True	False	98	18	0

There is no evidence of the coupled usage of PTMs and fairness libraries

LLM Bias Contributions



LLM Bias Contributions



Efficiency of Learning-Based Systems

Efficiency of traditional software systems

Conferences > 2012 First International Work...

How to measure energy-efficiency of software: Metrics and measurement results

Publisher: IEEE

Cite This | PDF

Timo Johann ; Markus Dick ; Stefan Naumann ; Eva Kern **All Authors**

Conferences > 2015 Sixth International Gree...

Using the Greenup, Powerup, and Speedup metrics to evaluate software energy efficiency

Publisher: IEEE

Cite This | PDF

Sarah Abdulsalam ; Ziliang Zong ; Qijun Gu ; Meikang Qiu **All Authors**

Efficiency of learning-based software systems

arXiv:1907.10597v3 [cs.CY] 13 Aug 2019

Green AI

Roy Schwartz[*◊] Jesse Dodge[*◊♣] Noah A. Smith[◊♡] Oren Etzioni[◊]

[◊]Allen Institute for AI, Seattle, Washington, USA
[♣] Carnegie Mellon University, Pittsburgh, Pennsylvania, USA
[♡] University of Washington, Seattle, Washington, USA

July 2019

Abstract

The computations required for deep learning research have been doubling every few months, resulting in an estimated 300,000x increase from 2012 to 2018 [2]. These computations have a surprisingly large carbon footprint [40]. Ironically, deep learning was inspired by the human brain, which is remarkably energy efficient. Moreover, the financial cost of the computations can make it difficult for academics, students, and researchers, in particular those from emerging economies, to engage in deep learning research.

This position paper advocates a practical solution by making **efficiency** an evaluation criterion for research alongside accuracy and related measures. In addition, we propose reporting the financial cost or "price tag" of developing, training, and running models to provide baselines for the investigation of increasingly efficient methods. Our goal is to make AI both greener and more inclusive—enabling any inspired undergraduate with a laptop to write high-quality research papers. Green AI is an emerging focus at the Allen Institute for AI.

1 Introduction and Motivation

Since 2012, the field of artificial intelligence has reported remarkable progress on a broad range of capabilities including object recognition, game playing, machine translation, and more [36]. This progress has been achieved by increasingly large and computationally-intensive deep learning models.[1] Figure 1 reproduced from [2] plots training cost increase over time for state-of-the-art deep learning models starting with AlexNet in 2012 [20] to AlphaZero in 2017 [38]. The chart shows an overall increase of 300,000x, with training cost doubling every few months. An even sharper trend can be observed in NLP word embedding approaches by looking at ELMo [29] followed by BERT [8], openGPT-2 [30], and XLNet [48]. An important paper [40] has estimated the carbon footprint of several NLP models and argued that this trend is both environmentally unfriendly (which we refer to as Red AI) and expensive, raising barriers to participation in NLP research.

This trend is driven by the strong focus of the AI community on obtaining "state-of-the-art" results,[2] as exemplified by the rising popularity of leaderboards [46, 45], which typically report accuracy measures but omit any mention of cost or efficiency (see, for example, `leaderboards.allenai.org`). Despite the clear benefits of improving model accuracy in AI, the focus on this single metric ignores the economic, environmental, or social cost of reaching the reported accuracy.

We advocate increasing research activity in Green AI—AI research that is more environmentally friendly and inclusive. We emphasize that Red AI research has been yielding valuable contributions to the field of AI, but it's been overly dominant. We want to shift the balance towards the Green AI *option*—to ensure that any inspired undergraduate with a laptop has the opportunity to write high-quality papers that could be accepted at premier research conferences.

*The first two authors contributed equally. The research was done at the Allen Institute for AI.
[1]For brevity, we refer to AI throughout this paper, but our focus is on AI research that relies on deep learning methods.
[2]Meaning, in practice, that a system's accuracy on some benchmark is greater than any previously reported system's accuracy.

1

SOFTWARE TECHNOLOGY

Editor: Christof Ebert
Vector Consulting Services
christof.ebert@vector.com

Green IT and Green Software

Roberto Verdecchia and Patricia Lago, Vrije Universiteit Amsterdam
Christof Ebert, Vector Consulting Services
Carol de Vries, PhotonDelta

From the Editor

Ecologic behavior is the need across the world to mitigate the impacts of climate change. Software and IT play a pivotal role toward ecologic behaviors for many reasons. Being aware that IT systems alone already consume 10% of global electricity, the leading software practitioners must embark on green IT and green coding. Read in this article about hands-on guidance on how you can contribute toward more ecologic software. I look forward to hearing from you about this column and the technologies that matter most for your work.—*Christof Ebert*

SOFTWARE AND IT usage are continuously growing to keep our society active and manage our individual lives. But as they grow, their energy demand is exploding. By 2030, data centers alone will already consume some 10% of the global electricity.[1] Including the Internet, telecommunications, and embedded devices, the energy consumption will be one-third of the global demand. Understanding that end users only consume what we offer, it is the community of software developers who must become active in ecologic behaviors. Green IT is the call of today. Each single line of code that we develop today may still be running years from now on zillions of processors, eating energy and contributing to global climate change.

Green IT and green coding describe a paradigm switch in which software engineers, developers, testers, and IT administrators can make their solutions and services more energy efficient. Every single software person can contribute. In this article, we provide hands-on guidance on how to reduce the energy waste of your software and thus contribute to more ecologic behaviors.

Green IT

With the introduction of high-bandwidth data transfers, affordable data plans, the generalized migration of software applications and data management to the cloud, the wide usage of streaming services, and, obviously, the many embedded computers in our everyday lives, digital infrastructures are experiencing an ever-growing demand for energy. While digital transformation looks impressive from an economic perspective, it has its downside on the ecologic footprint of these businesses.

An immediate action is to adopt more renewable energy. Energy-hungry companies such as Microsoft, Google, and Amazon are currently investing in water energy, for example, to cool their data centers; solar energy; and wind farms. Many companies engage in trading CO_2 certificates to give a green color to the energy waste of their data centers. But renewable energy only "cures" the symptoms. It does not really solve reducing the need for energy.

Digital Object Identifier 10.1109/MS.2021.3102254
Date of current version: 22 October 2021

Efficiency Research Challenges

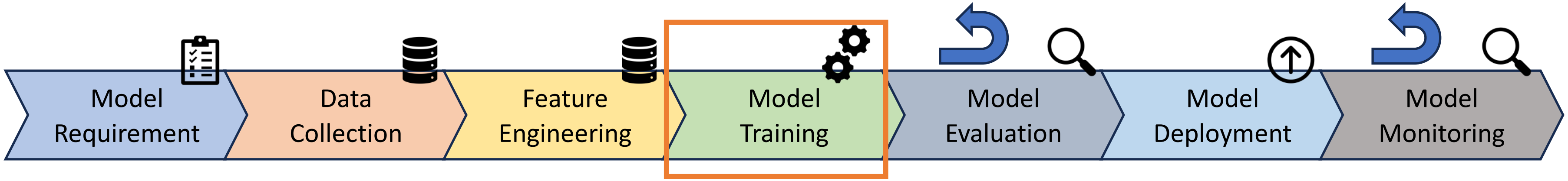
Challenge 5 (CH5)

Predicting a priori the training time of machine learning models could support early design decisions for learning-based systems development.

Challenge 6 (CH6)

Analyzing and improving the efficiency-effectiveness trade-off of resource-intensive Large Language Models.

Challenge 5: Training Time Prediction



- Model training is the most computationally expensive phase of the development workflow

Early predicting the training time of ML models can help in standardizing some phases of the development process

Full Parameter Time Complexity

- The FPTC method defines the training time of several ML models as a function of ML model and dataset parameters

$$FPTC_{LogReg} = F(Qm^2vn) * \omega_{LogReg}$$

Logistic Regression

$$FPTC_{RF} = F(s(m+1)nv\log_2(n)) * \omega_{RF}$$

Random Forest

Contribution 6: FPTC Evaluation

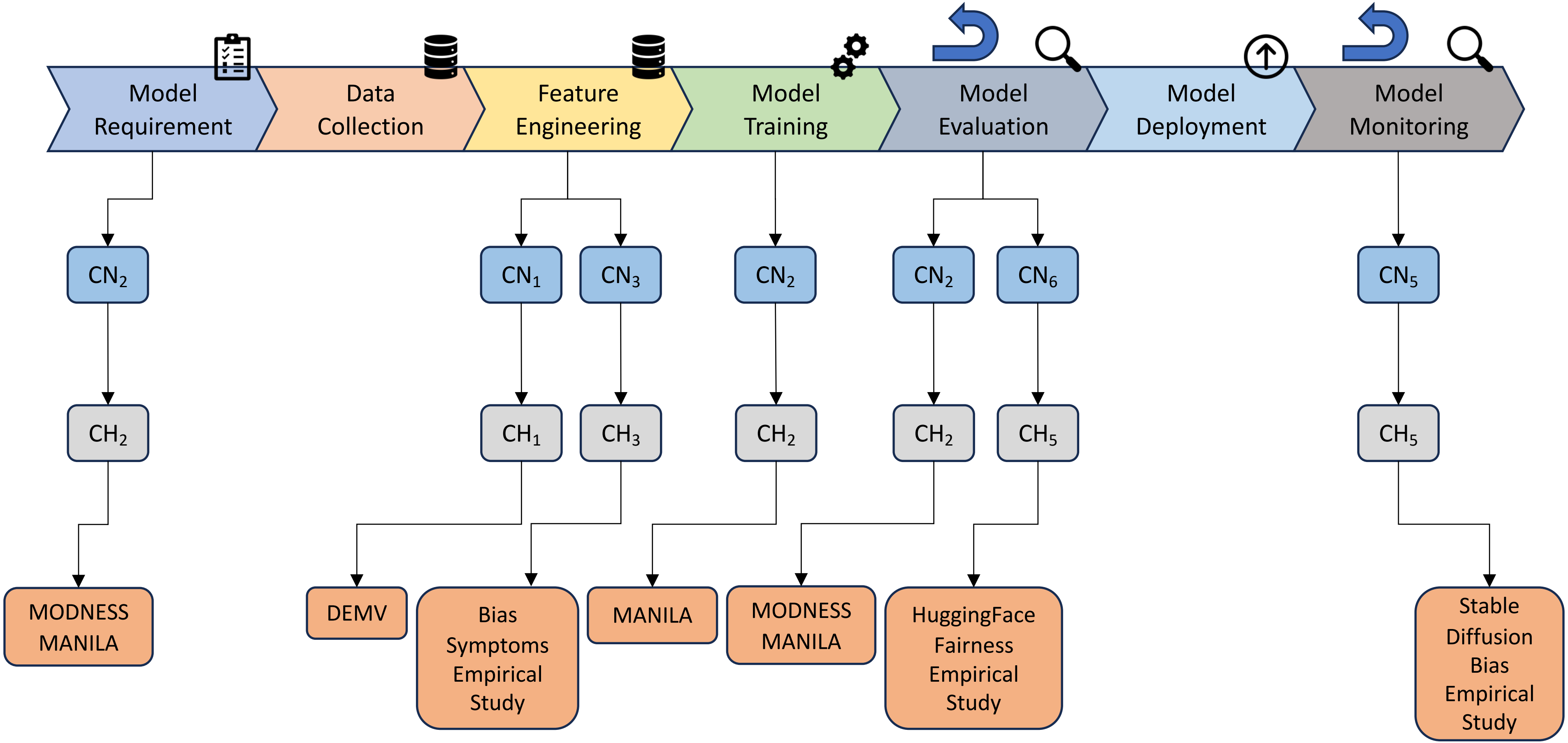
Logistic Regression

The FPTC method can correctly predict the training time of some datasets while it fails in others

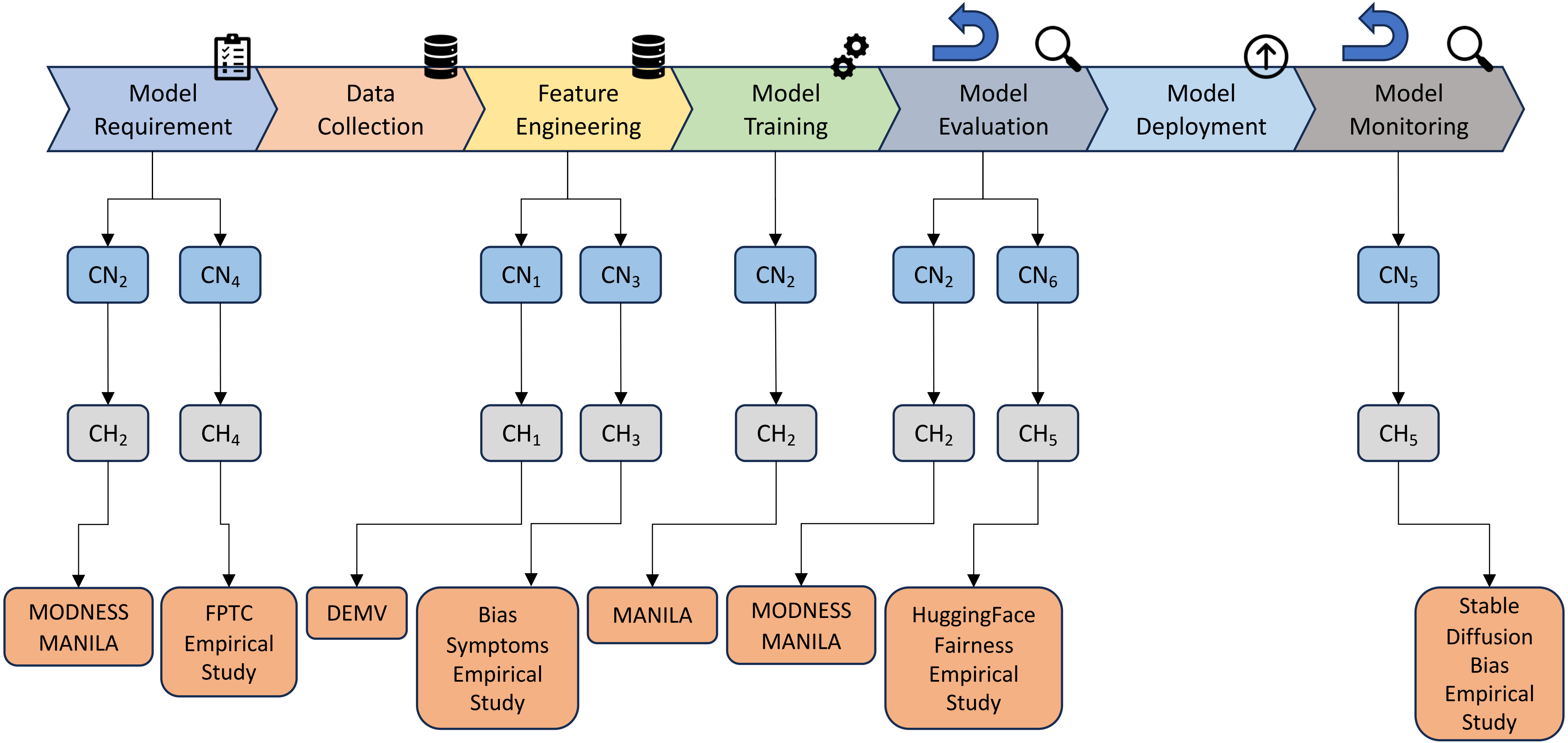
Random Forest

The FPTC method can correctly predict the training time of almost all datasets

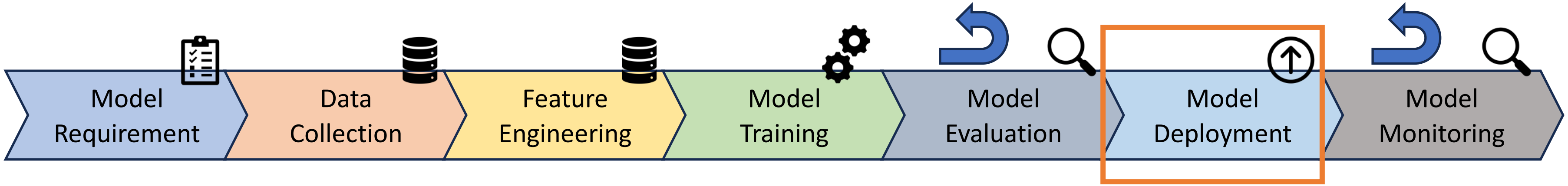
FPTC Analysis Contribution



FPTC Analysis Contribution



Challenge 6: LLM Deployment



- Large Language Models are highly effective but also expensive to deploy

How can we support the deployment of Large Language Models?

Contribution 7: Analysis LLM Compression Methods

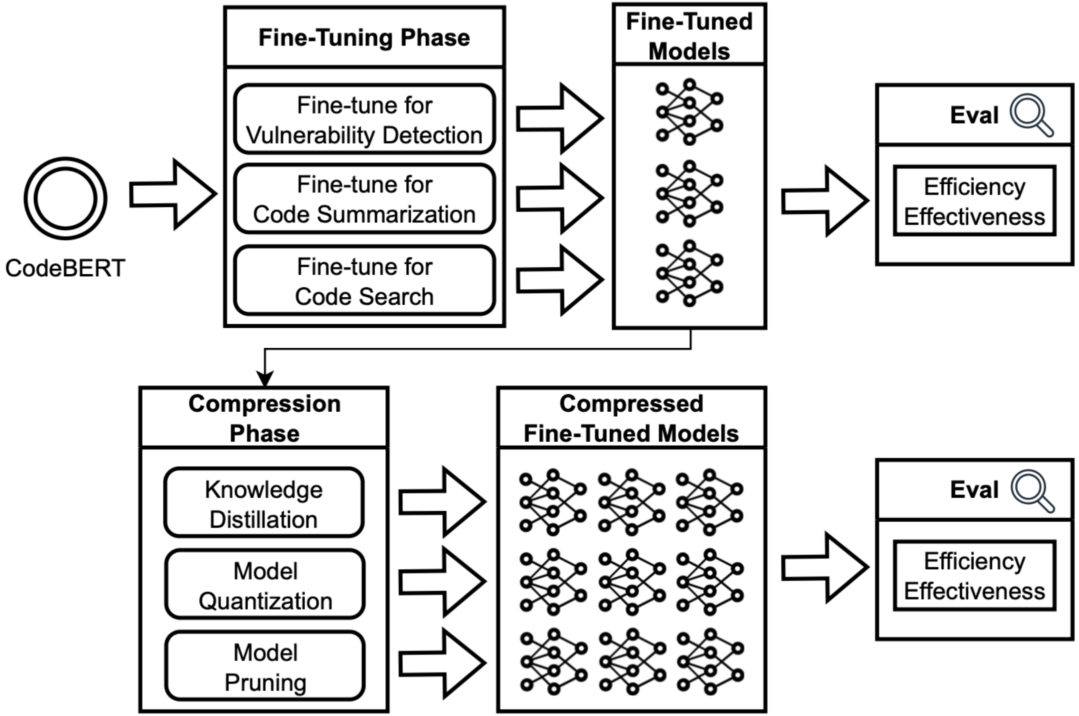
- Compression strategies have been proposed to ease the deployment of Large Language Models

Contribution 7: Analysis LLM Compression Methods

- Compression strategies have been proposed to ease the deployment of Large Language Models

How do compression strategies affect the effectiveness, inference time, and model size of LLMs fine-tuned for SE tasks?

Experimental Methodology



Takeaways

Takeaways

- Improving inference time and model size: *Knowledge Distillation*

Takeaways

- Improving inference time and model size: *Knowledge Distillation*
- Reduce model size only: *Quantization*

Takeaways

- Improving inference time and model size: *Knowledge Distillation*
- Reduce model size only: *Quantization*
- Reduce GPU inference time: *Knowledge Distillation*

Takeaways

- Improving inference time and model size: *Knowledge Distillation*
- Reduce model size only: *Quantization*
- Reduce GPU inference time: *Knowledge Distillation*
- Reduce CPU inference time: *Knowledge Distillation* or *Pruning* (with proper configuration)

Contribution 8: GreenStableYolo

- GreenStableYolo is a search-based algorithm to optimize both inference time and image quality of Stable Diffusion models
- It searches for the best hyperparameter settings and prompt structure

Contribution 8: GreenStableYolo

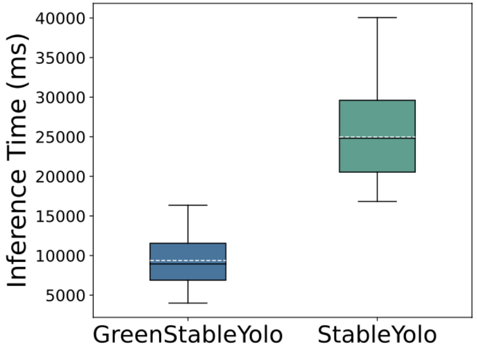
- GreenStableYolo is a search-based algorithm to optimize both inference time and image quality of Stable Diffusion models
- It searches for the best hyperparameter settings and prompt structure

GreenStableYolo works on the black-box model without changing its architecture

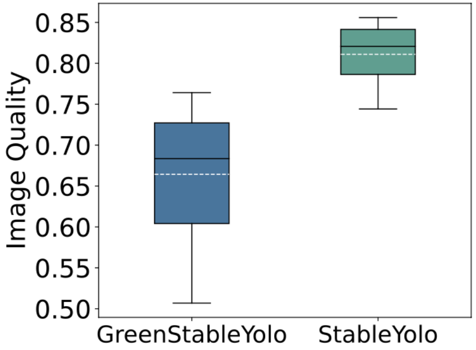
Search Space

- **Inference steps (1 to 100):** the AI's image generation iterations;
- **Guidance scale (1 to 20):** the impact of the prompt on image generation;
- **Guidance rescale (0 to 1):** rescales the guidance factor to prevent over-fitting;
- **Positive prompt:** used to describe images and improve their details, e.g., "photograph", "color", and "ultra real";
- **Negative prompt:** avoided description during image generation, e.g., "sketch", "cropped", and "low quality".

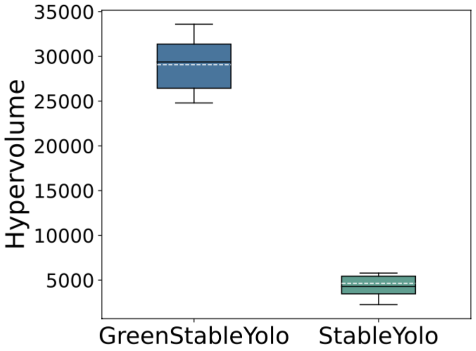
Evaluation



(A) Inference time

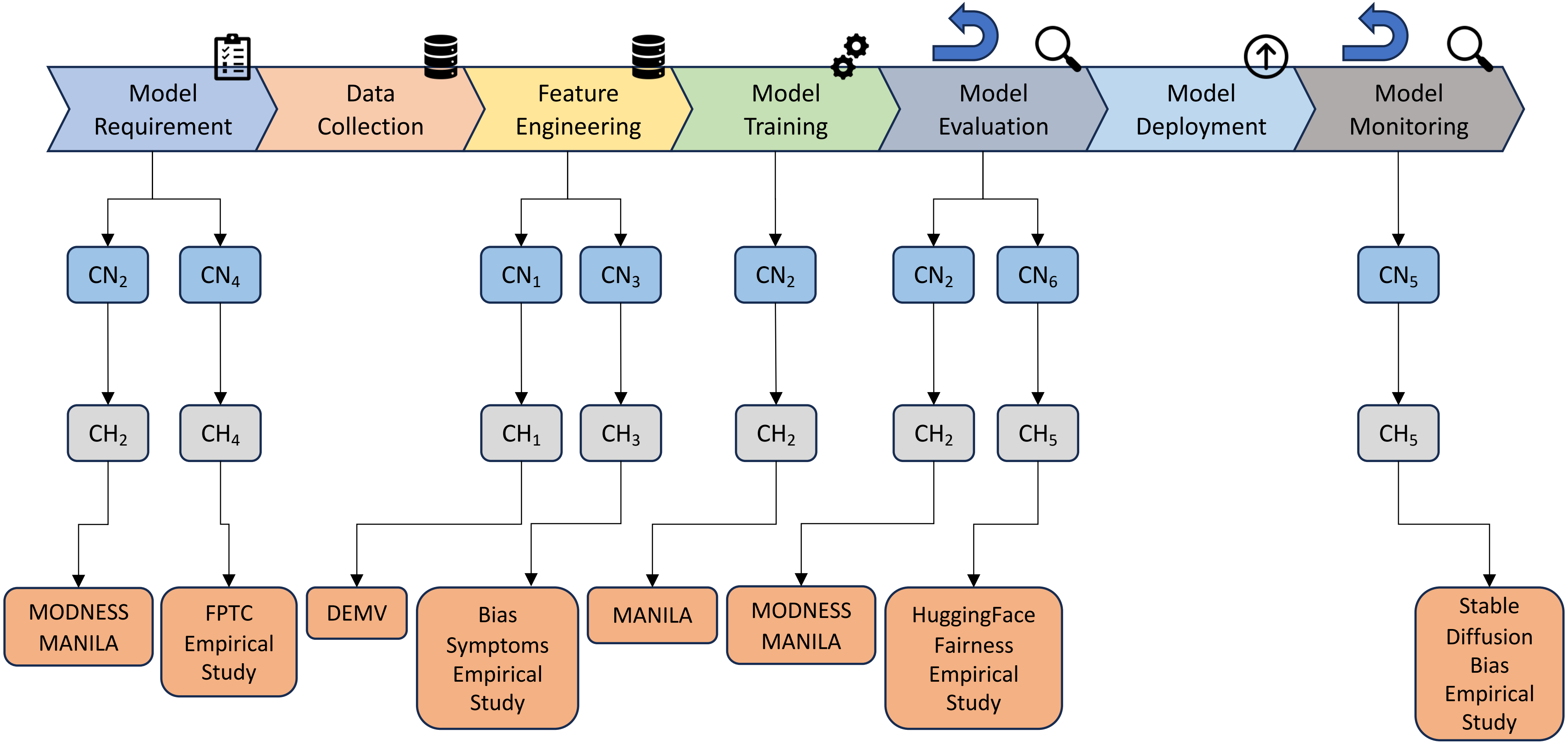


(B) Yolo quality measure

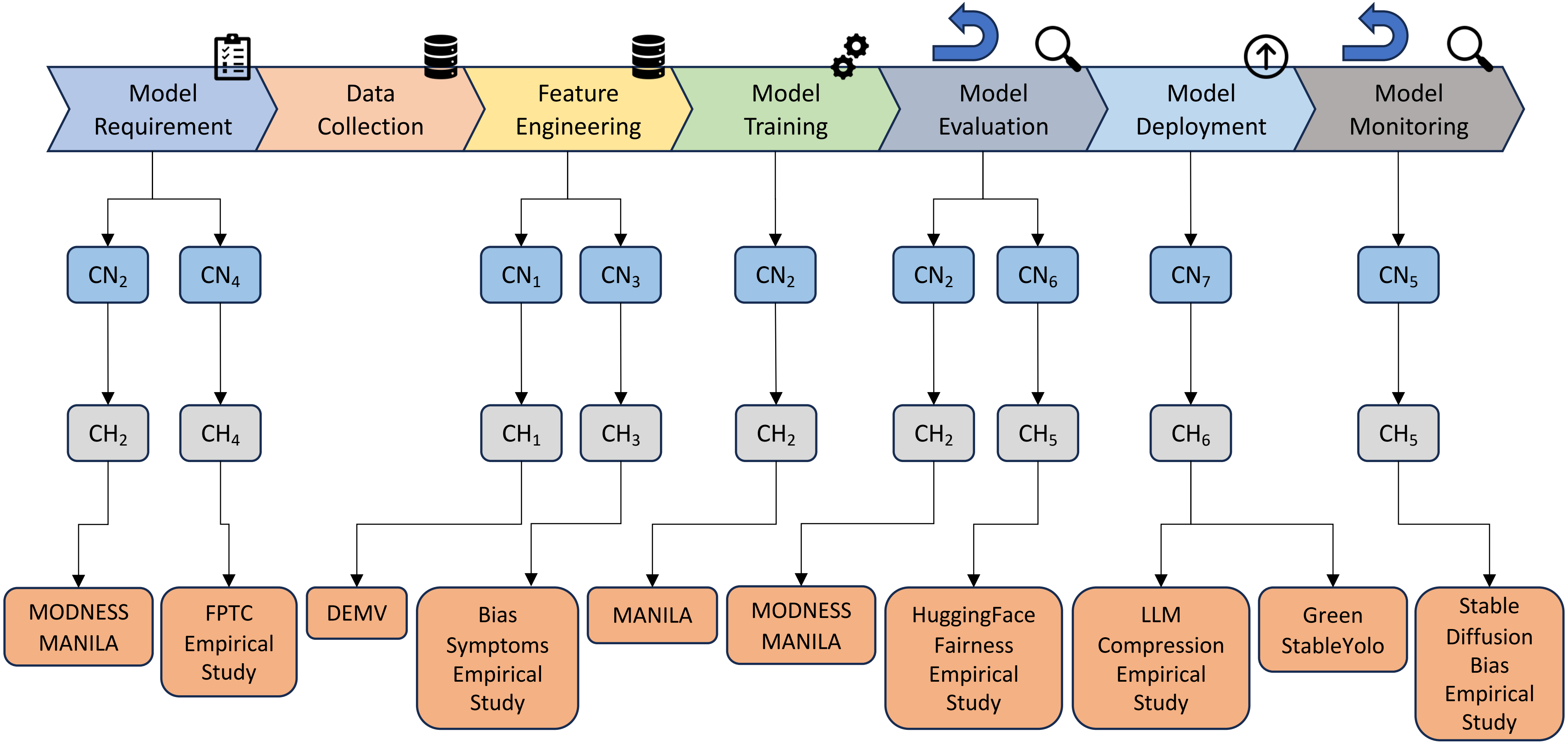


(C) Hypervolume

LLM Efficiency Contributions



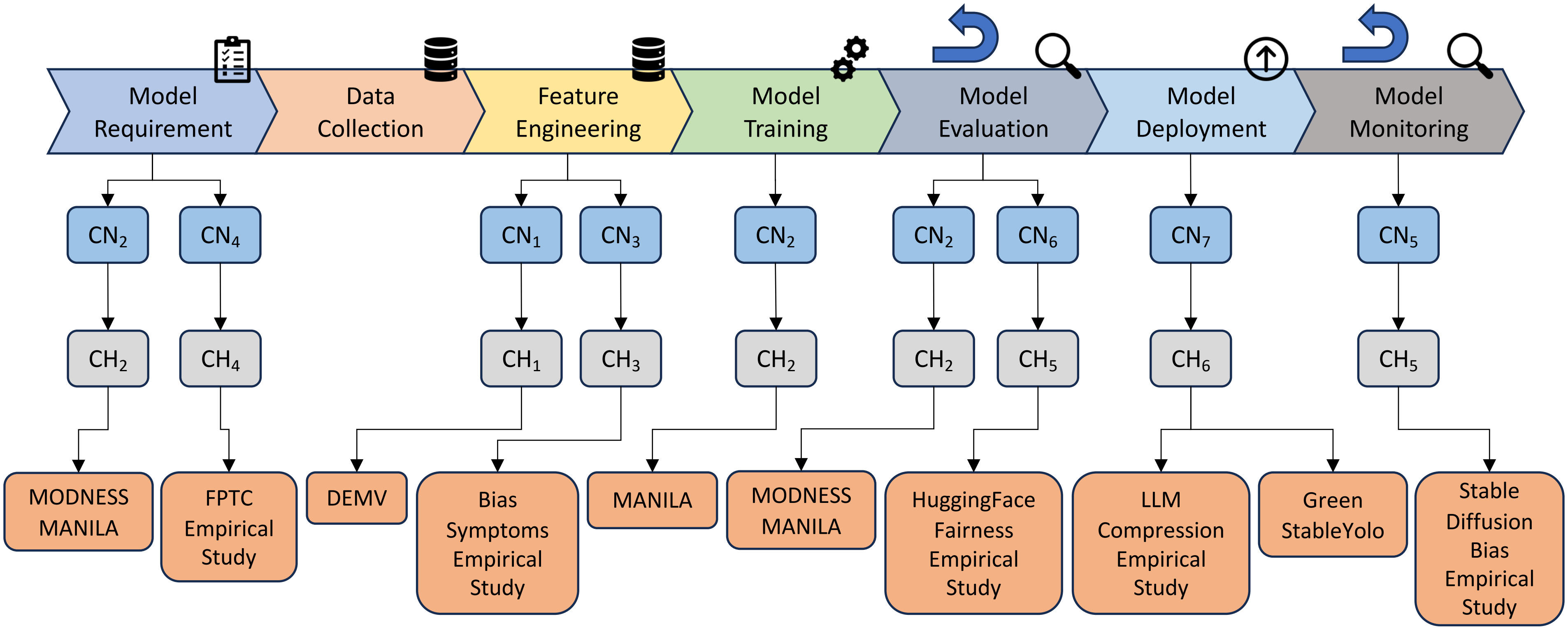
LLM Efficiency Contributions



Conclusions

Conclusion

- The presented contributions cover quality aspects of the whole LBS workflow



Future Work

Future Work

- Fully automate the development of fair and efficient learning-based systems

Future Work

- Fully automate the development of fair and efficient learning-based systems
- Early bias detection and mitigation from model requirements

Future Work

- Fully automate the development of fair and efficient learning-based systems
- Early bias detection and mitigation from model requirements
- Automatic selection of LLM compression strategies

Future Work

- Fully automate the development of fair and efficient learning-based systems
- Early bias detection and mitigation from model requirements
- Automatic selection of LLM compression strategies
- Energy and fairness improvement of Text-To-Image generation models

Future Work

- Fully automate the development of fair and efficient learning-based systems
- Early bias detection and mitigation from model requirements
- Automatic selection of LLM compression strategies
- Energy and fairness improvement of Text-To-Image generation models
- Trade-off analysis on fairness and efficiency

UNIVERSITÀ DEGLI STUDI DELL’AQUILA

DISIM
Dipartimento di Ingegneria e Scienze dell’Informazione e Matematica

Thank you for your attention!

Email: giordano.daloisio@univaq.it
Web: https://giordanodaloisio.github.io