Computational Communication Science 2
Week 7 - Lecture
»Coding in an academic context«

Marthe Möller

a.m.moller@uva.nl

May 13, 2024

Digital Society Minor, University of Amsterdam

Today

Recap

A critical reflection

Consequences of bias

Validation revised

Pulling your weight

Looking Back and Ahead

Recap

Recap

Techniques you now master:

- Text as data
- Recommender systems
- Supervised machine learning

Today, we:

- Reflect on computational methods and validation
- Discuss responsible coding

Recap

Last week, we talked about validation.

Validation: How well does the model work?

Recap

Validation metrics:

- Precision (How much of what we found is correct?)
- Recall (How many of the cases that we wanted to find did we find?)
- Accuracy (In which percentage of all cases was the model correct?)
- F1-score (Harmonic mean of precision and recall)

Recap

Validation using metrics such as recall and precision pertains to how well the model reflects the data it was trained on.

A critical reflection

Assessing the performance of models

Validation using metrics such as recall and precision pertains to how well the model reflects the data it was trained on.

Could this be problematic? What do you think?

Problems associated to comparison with training data: 1

Training data can be biased due to bad input

- Training data can be of poor quality

Bad-quality training data

Poor quality coding

- The definition of concepts is unclear
- Characteristics of a coding-task may increase bias (e.g., length)
- Concepts are subjective: The ground truth itself is biased (e.g., Hube et al., 2019; Van der Velden et al., 2023; Webb Williams et al., 2023)
- Basile et al. (2023): Perspectivism-approach

Problems associated to comparison with training data: 1

Training data can be biased due to bad input

- Training data can be of poor quality
- Training data may not be representative

Unrepresentative training data

Sources of training data

- Training data are typically gathered from the internet and specific platforms, introducing bias (Bender et al., 2021)
- Moderations on platforms can prevent marginalised groups from being heard (Bender et al., 2021)

Problems associated to comparison with training data: 1

Training data can be biased due to bad input

- Coding can be of poor quality
- Training data may not be representative

The above can lead to models that only represent certain viewpoints held by certain groups.

Problems associated to comparison with training data: 2

No consideration for the costs and benefits of training models

- CO_2-emissions

Costs and benefits

CO2-emissions

- Training large models requires a lot of computing power and leads to CO2-emissions (Bender et al., 2021)
- The consequences of this are mostly felt by already marginalized groups (Bender et al., 2021)

Problems associated to comparison with training data: 2

No consideration for the costs and benefits of training models

- CO_2-emissions
- Unequal access to resources

Costs and benefits

Unequal access to resources

- Not all researchers have equal access to the resources for developing computational models (Baden et al., 2021; Bender et al., 2021)
- Not all non-researchers have equal access to the resources for using computational models (Bender et al., 2021)

Problems associated to comparison with training data: 2

No consideration for the costs and benefits of training models

- CO2-emissions
- Unequal access to resources

The above leads to increased societal differences and inequality.

Problems associated to comparison with training data: 3

Metrics say nothing about how a model works

- Latent constructs

Metrics say nothing about how a model works

Latent constructs (Baden et al., 2021)

- Social scientists measure complex constructs that can be measured in various ways
- For theory development, we need to know how concepts are measured
- Measurement validity versus technological performance

Problems associated to comparison with training data: 3

Metrics say nothing about how a model works

- Latent constructs
- Theoretical concepts

Metrics say nothing about how a model works

Theoretical concepts (Baden et al., 2021)

- Focus on detecting single and simple constructs instead of multiple and complex ones
- Difficult to get a holistic view
- Scholars often have to combine models to study their complext constructs

Problems associated to comparison with training data: 3

Metrics say nothing about how a model works

- Latent constructs
- Theoretical concepts

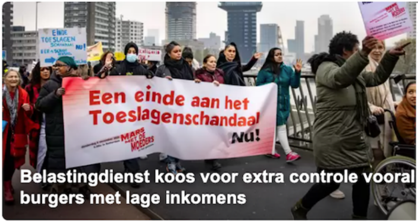
The above makes it increasingly difficult to use computational methods for theory development.

Consequences of bias

Bias in computational models

We see different types of challenges associated to computational models.
For society, are these actually problems? Why (not)?

Consequences of bias



23 november 2021 00:47
Laatste update: 1 dag geleden

761 NUjij-reacties

Het systeem van de Belastingdienst koos ervoor om de kinderopvangtoeslag vooral bij mensen met een laag inkomen extra te controleren. Dat heeft de fiscus toegegeven in antwoord op vragen van *Trouw* en *RTL Nieuws*.

Trouw

Toeslagen

Belastingdienst ging vooral achter lage inkomens aan

Om toeslagen te controleren op fouten en fraude gebruikte de Belastingdienst een zelflerend algoritme. Dat selecteerde vooral lage inkomens voor controle.

Jan Kleinnijenhuis 22 november 2021

De Belastingdienst heeft jarenlang specifiek burgers met een laag inkomen geselecteerd voor extra controle op

Fraudejacht bij Toeslagen

Belastingdienst controleerde extra bij lage inkomens in jacht op fraude

22 november 2021 22:59

Validation revised

A broader view on bias

Bias is not only about how well a model can mimic training data. It is also about evaluating the process underlying it.

Potential remedies

Invest in better training data 1

- Reflect on who provided the training data: e.g., who has access to the platforms that you gathered the data from? (Bender et al., 2022)
- Explain and argue for all the pre-processing steps that were taken, e.g., exclusion of specific language (Baden et al., 2022)
- Explicitely state your approach dependent on the concepts studied: Perspectivism-approach or a Ground-truth approach (Basile et al., 2023)
- Invest more resources into different approaches to validation (Baden et al., 2022)

Potential remedies

Invest in better training data 2

- Systematically study how bias can be mitigated
- Hube et al. (2019): Task characteristics can also be a remedy!

Potential remedies

Consider costs and benefits

- Report the time it took to train models (Bender et al., 2022)
- Add efficiency as an evaluation metric (Bender et al., 2022)
- Reflect on who the developed models can be valuable for (Bender et al., 2022)

Potential remedies

Theory development

- Clearly define and operationalize the constructs you are studying (Baden et al., 2022)
- Include scholars in the development of methods (Baden et al., 2022)

Pulling your weight

Potential remedies

Key to implementing the remedies listed in the previous section, is Open Science.

Open Science

Open Science is the movement that aims at more open and collaborative research practices in which publications, data, software and other types of academic output are shared at the earliest possible stage and made available for reuse. Open Science leads to greater scientific and societal impact. (NWO, 2024)

Open Science

Key to Open Science in this context, is sharing your code.

Open Science

Sharing your code is useful because:

- Increases the transparency of your own projects
- Increases the availability of and access to materials
- Reduces the need to keep reinventing the wheel

Open Science

Sharing your code is only useful if it is reproducible.

Reproducible code

Reproducble code is:

- Clear
- Readible
- Efficient

Open Computational Science

1. Document your code

Open Computational Science

Documentation debt: "When we rely on ever larger datasets we risk incurring documentation debt, i.e. putting ourselves in a situation where the datasets are both undocumented and too large to document post hoc." (Bender et al., p. 615)

Source Acknowledgement

```
"""
In your code, you can use """ or # to create a comment
    explaining your code. For example, to say that you
    are using a scraper developed by A. Person (2015),
    which can be found on www.somewebsite.com
"""
```

Writing reproducible code

2. Create functions (instead of repeating code)

Not:

```
# Rerun this code three times, once for each name:
# Mike, Elsa, and Minna

print("[change name here] is cool!")
```

But:

```
names = ["Mike", "Elsa", "Minna"]

def cool_caller(name):
print(name + " is cool!")

for name in names:
cool_caller(name)
```

Writing reproducible code

3. Avoid hard-coding values

Not: "myfile.csv" or 50 within your script.

But:
OUTPUTFILE ="myfile.csv"
MAXNUMBER=50

Writing reproducible code

4. Use built-in functions and libraries

And load all modules/packages at the start of your code

Writing reproducible code

6. Use informative names for files and variables (check out PEP8!)

Writing reproducible code

7. Keep lines of code fairly short

Writing reproducible code

8. Write code that makes sense to others, and it will make sense to the future you as well.

Looking Back and Ahead

Looking back

You started with the basics (e.g., what is a list, how to save data, write a loop).

In two months, you learned:

- How to read in data
- How to preprocess data
- How transform text into data that a computer can understand
- How to compare texts to provide a recommendation
- How to analyze text to classify it automatically

Looking at the very near future

Two grades for this course remain:

- The last MC-questions
- The take-home exam

Looking at the near future

Final course of the minor: the research project!

- You will combine your programming skills with your skills as a researcher
- Run a research project about ComScience using the materials you created for the group assignment in this course
- More information follows in the first meeting of the research project

Looking at the future

CCS-1 and CCS-2: An introduction to coding. You can continue to learn and work with Python.

No course materials and instructors to help you out, but there are a lot of resources online!

Our tips:

- Error message? Google is your best friend!
- Check out the documentation of any module that you use to learn how it works
- Check out pubs using the method you are interested in. Often, they publish their python scripts (e.g., Meppelink et al., 2021)

Looking at the future

When you use your computational skills for your thesis or any other project, you are part of the research community. You add value to your own work by making it accessible and understandable for others!

Looking at the future

We taught you basic principles and techniques that underly frequently used methods. These techniques develop and change constantly - make sure that you stay updated on them!

Looking at the future

We taught you how to drive, now you can go out and explore the world of computational (communication) science!

Looking at the future

Thank you for the past weeks and enjoy the research project!