dlbs Mini-Challenge Title

Names of the team members

28. Oktober 2025

1 General Remarks on the Assignment

Choose a problem statement and use case from the fields of image and/or signal processing that you would like to tackle in this mini-challenge. The problem/use case should be solvable using deep learning for object detection, image segmentation, instance segmentation, or signal prediction. You may choose any programming language. The solution approach to solving the MC is based on these articles:

- How to Avoid Machine Learning Pitfalls: A Guide for Academic Researchers [Lon23]
- A Recipe for Training Neural Networks [Kar19]

1.1 Collaboration

The mini-challenge assignment is typically submitted in teams of two. Exceptions, such as individual work, must be discussed with the DS subject matter expert: In the submission, for certain designated tasks, the person who performed that specific part must be indicated. Students are allowed to collaborate with other teams for this mini-challenge, but each team must hand in a unique submission.

1.2 Scheduling and Deadlines

- Milestone 1: Problem statement (SW 9)
- Milestone 2: Two individual check-ins (SW2 to SW14)
- Milestone 3: Demo/Presentation of the MC (SW 13/14)
- Milestone 4: Final submission of the written documentation (SW 15)

The deadlines can be found on Spaces in the calendar.

1.3 Submissions and Deliverables

Document your solution to the MC in an appropriate format (in German or English), including a final reflection. The data does not need to be submitted. However, the solution process must be traceable and selected data must be visualized. If available, provide the link to the data. Note: If a model is did not learn the target, no points will be awarded.

Deliverables:

- Problem statement: Summarize the research question, data situation, and methods/approach (see Sect. 2, typically 1-2 pages) and discuss it with a DS subject matter expert during one of the check-ins.
- Report (see Sect. 2), typically 8-12 pages, max 15 page, for more use appendix)
- Slides from your presentation, if applicable
- Git repository with source code – including an HTML export of your notebooks
- Self-assessment of grades incl. reflection why

1.4 Demo/Presentation

10 minutes per team with subsequent Q&As. Content: Recap of the problem statement / use case, main results incl. overview of achieved performances, conclusion. The demo will be stopped if your presentation exceeds the allotted time. Attendance at other presentations is encouraged.

2 Assignment

2.1 Problem Statement / Use Case (Report)

TASK: Describe the problem statement and use case. Explicitly formulate hypotheses, research questions, or objectives. This section should explain the WHAT of your MC. Approximately 100-200 words.

2.2 Data (Report)

TASK: Choose a dataset with which the problem statement / use case can be solved. You may use images (2D, 3D), videos, sensor signals, point cloud data, audio data, or similar. The dataset can be publicly available, your own, or from a business context. The data does not need to be shared with the DS subject matter experts as long as the solution process is traceable. Since we are solving a supervised problem in this MC, ground truth data or labels should be available or can be generated with minimal effort. Note: For this MC, it is acceptable to work with "too small" dataset. In such a case, this aspect should be discussed accordingly. Describe the data in approximately 100-200 words.

2.3 Methods (Report)

TASK: Describe and discuss the choice of your methods (baseline, DL architectures, evaluation) in relation to your problem statement / use case (HOW). Draw/Sketch a figure to illustrate your workflow. Provide an overview of the methods and parameters you used. WHY did you choose a particular method, metric or parameter? Approximately 300 words.

3 Results

Zusammenfassung

(Report) *TASK: Summary of the problem statement, data, methods, and the main results including key metrics. Approximately 200 words.*

3.1 Data Inspection (Notebook)

TASK:

- *Perform a dlbs-specific exploratory data analysis following these three steps [Kar19] and have each person discuss one step in 2-3 sentences:*
  - *Inspect the data visually*
  - *Develop a qualitative understanding of the data*
  - *Analyze the data quantitatively (using code)*
- *Follow for the data inspection guidelines from [Lon23]:*
  - *Do take the time to understand your data*
  - *Don't look at all your data*
  - *(Do make sure you have enough data)*
  - *Do talk to domain experts*
  - *Do survey the literature*
  - *Do think how your model will be deployed*
  - *Don't allow test data to leak into the training process*
- *Report in a notebook, a Weights & Biases (WandB) report, or similar (not in report).*

3.2 Setup a Baseline (Notebook)

TASK:

- *Choose a simple model as a baseline, select appropriate metrics for evaluating the networks, and discuss your choice including parameters. The baseline should be trained/generated/calculated, analyzed, and visualized (loss, predictions, etc.).*
- *Each person should discuss one additional selected aspect from [Kar19] in 2-3 sentences.*
- *Follow and discuss the baseline guidelines from [Lon23]:*
  - *Do use an appropriate test set*
  - *Do use a validation set*
  - *Do save some data to evaluate your final model instance*
  - *Don't use accuracy with imbalanced data sets*
  - *Do evaluate a model multiple times*
  - *Don't do data augmentation before splitting your data*
  - *Don't allow test data to leak into the training process*
- *Report in a notebook, a Weights & Biases (WandB) report, or similar (not in report).*

3.3 Overfit (Notebook)

TASK:

- *Focus on the loss of the training set*
- *Gradually introduce additional complexities and document them*
- *If a model does not learn, first resolve this issue (by gradually increasing complexities) before proceeding further*
- *Choose a model architecture to solve your research question and perform overfitting according to [Kar19]. Each person should choose a different aspect from [Kar19] to implement or execute and discuss it in 2-3 sentences.*
- *Follow the overfitting guidelines from [Lon23]:*
  - *Don't use inappropriate models*
  - *Do try out a range of different models*
  - *Do evaluate a model multiple times*
  - *Don't do data augmentation before splitting your data*
- *Optional bonus task: Compare and evaluate an additional DL architecture or another model*
- *Report in a notebook, a Weights & Biases (WandB) report, or similar (not in report).*

3.4 Regularize (Notebook)

TASK:

- *Gain higher validation accuracy by sacrificing some training accuracy*
- *Each person conducts 1-2 regularization experiment(s) according to [Kar19]. Discuss your results for each regularization aspect in 2-3 sentences.*
- *Compare your models in an appropriate manner and follow the guidelines from [Lon23]:*
  - *Do evaluate a model multiple times*
  - *Don't assume a bigger number means a better model*
  - *Do use statistical tests when comparing models*
  - *Do correct for multiple comparisons*

  - *Don't always believe results from community benchmarks*
  - *Do be transparent*
  - *Do report performance in multiple ways*
  - *Do be careful when reporting statistical significance*
  - *Do look at your models*
- *Report in a notebook, a Weights & Biases (WandB) report, or similar (not in report).*

3.5 Tune (Notebook)

TASK:

- *Continue with the selected model and perform one of the following tuning variants:*
  - *Random search*
  - *Grid search*
  - *Hyperparameter optimization*
- *Follow the tuning guidelines from [Lon23]:*
  - *Do evaluate a model multiple times*
  - *Do optimize your model's hyperparameters*
  - *Do be careful where you optimize hyperparameters and select features*
- *Report in a notebook, a Weights & Biases (WandB) report, or similar (not in report).*

3.6 Optional: Squeeze the Juice Out of It (Notebook)

TASK:

- *Choose a method from the following:*
  - *Ensemble models*
  - *Train longer*
- *Follow the guidelines from [Lon23] for this task:*
  - *Do consider combinations of models*
  - *Do evaluate a model multiple times*
- *Report in a notebook, a Weights & Biases (WandB) report, or similar (not in report).*

3.7 Most Relevant Results (Report)

TASK: Select your most important results and describe them in approximately 400-500 words. Use figures if applicable (e.g., particularly good vs. poor model predictions).

4 Discussion (Report)

TASK:

- *Discuss your results explicitly in relation to your research question(s) in approximately 500-600 words. How well does your model address your problem statement / use case? What do the errors/scores mean for your use case? By discussing, it is meant that you should examine the advantages and disadvantages, opportunities and risks of your solution approach or the results specifically for your use case. How do the baseline results compare to the optimized models? Which cases work well and which do not? Can the research question be answered with the chosen models and data?*
- *For the discussion, follow the guidelines from [Lon23]:*
  - *Don't generalize beyond the data*
  - *Don't always believe results from community benchmarks*
  - *Do be transparent*

5 Reflection (Report)

TASK: Provide a final reflection in approximately 150-200 words. What are the conclusions for the specific problem statement / use case? What went well in this MC? What would you do differently next time? What would you change in the assignment? Focus on technical reflection.

6 Code

TASK: Provide access to a tidy Git repository with a README. The submission must not contain any commented-out code sections (except pip install statements). Note on code comments: Code comments are helpful when they explain why a decision was made, why a calculation was performed in a specific way, or according to which source something was implemented. In practice, however, during code development it is often forgotten to update the code comments. Therefore, comments are often obsolete. Commented-out code is typically not tested, and later it may not be clear whether it still works. My recommendation is: as many comments as necessary, but as few as possible. In general, the code should be written in a self-explanatory manner.

Literatur

[Kar19] Andrej Karpathy. A recipe for training neural networks, 2019.

[Lon23] Michael A. Lones. How to avoid machine learning pitfalls: a guide for academic researchers, 2023.