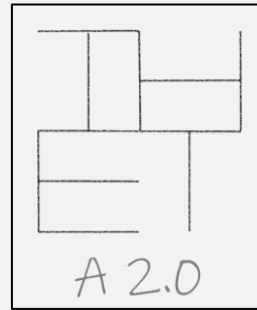


Comparison of Hyperparameter Optimization Tools in Deep Learning Using AutoEncoders in the Context of Classification



Jakub Marecki, Joanna Szyda



Wrocław University
of Science and Technology



WROCLAW UNIVERSITY
OF ENVIRONMENTAL
AND LIFE SCIENCES



Goal

- Evaluate the performance of popular **HyperParameter Optimization (HPO)** tools for **Deep Learning (DL)** using supervised **AutoEncoders (AE)**
→ noting new 🤔
- For biological data
→ noting new still 🤔🤔
- ... small biological data with class imbalance

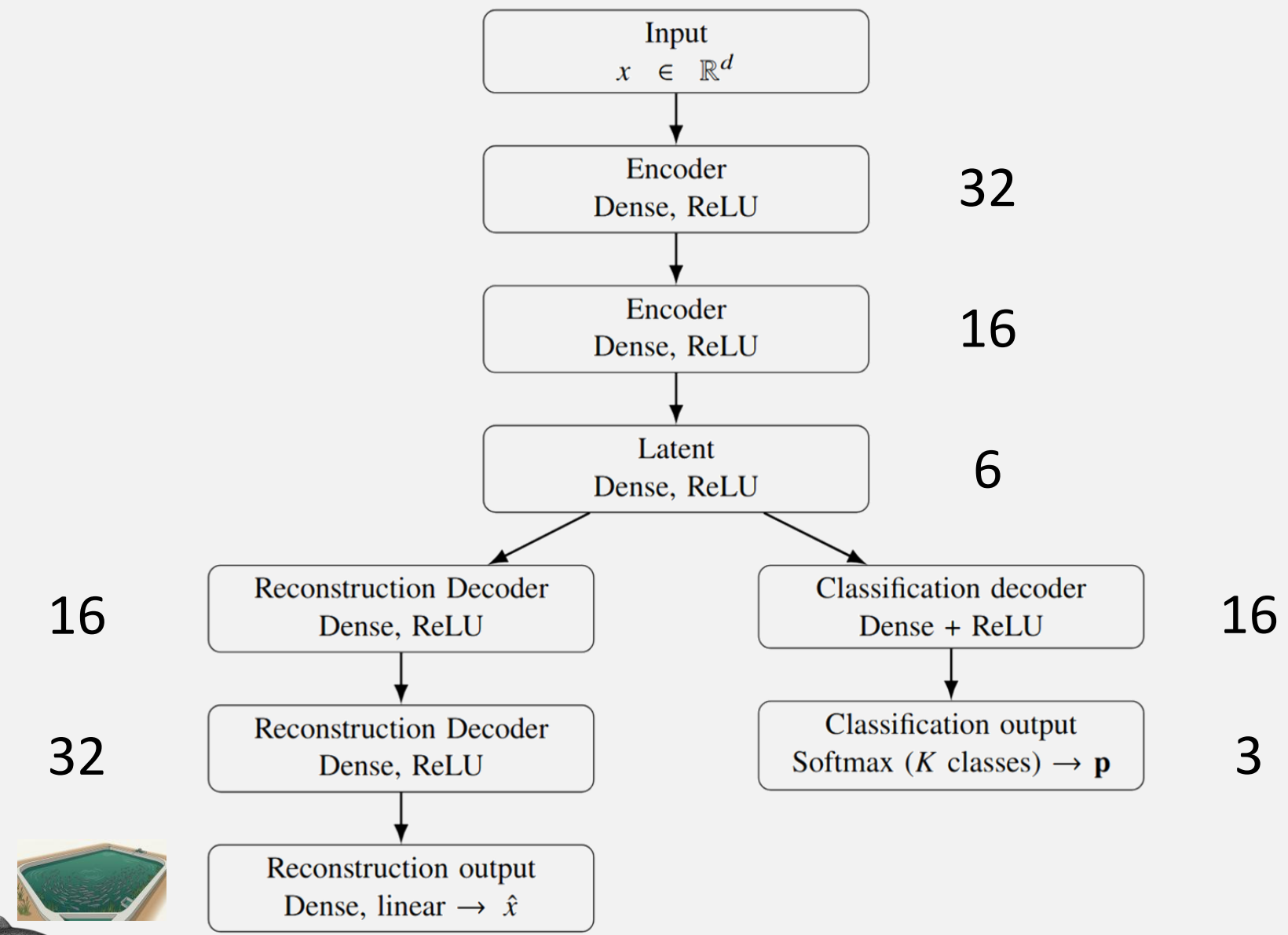


Outline

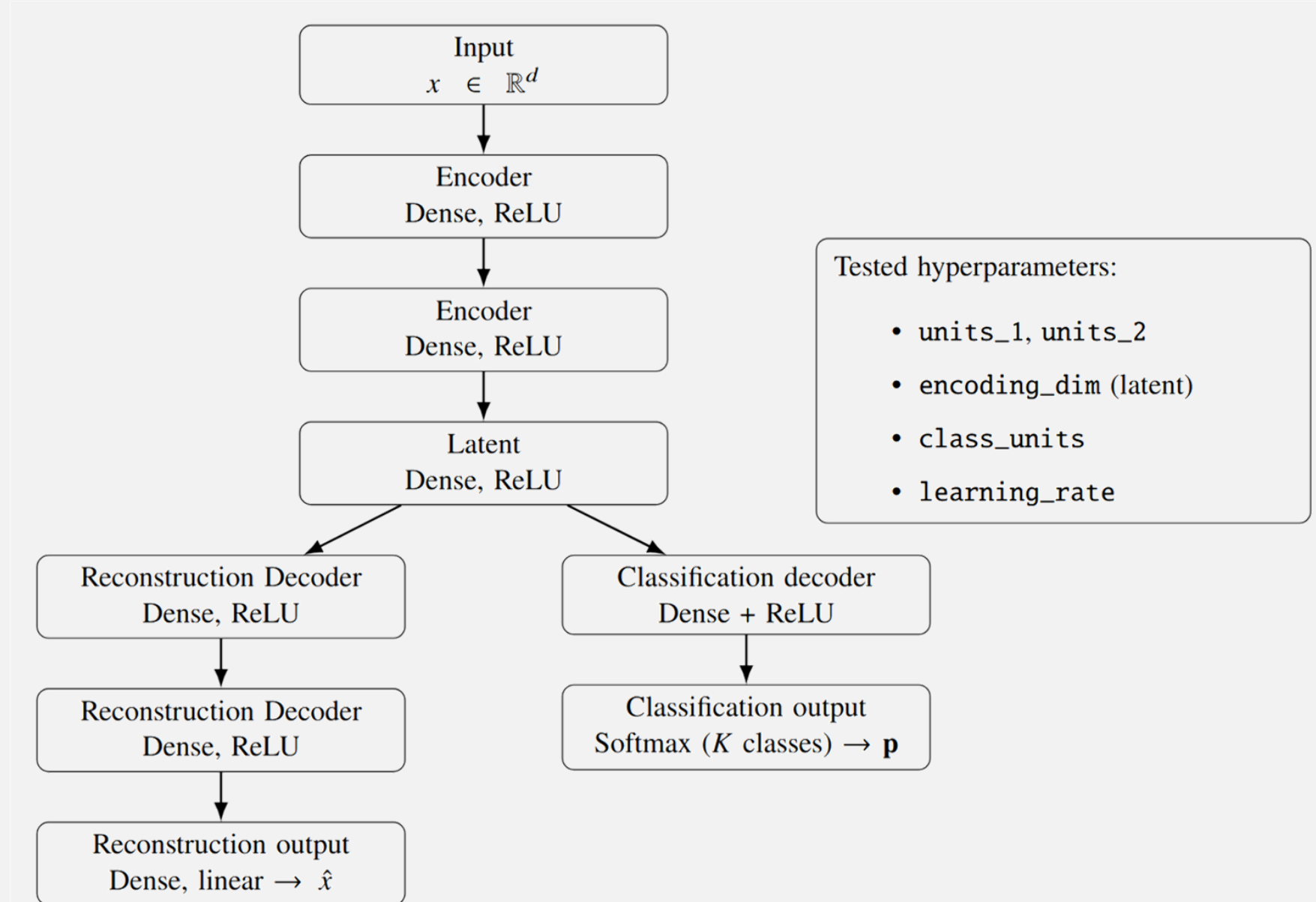
1. Baseline architecture
2. HPOs
3. (simulated) Data sets
4. Results



AutoEncoder – Baseline heuristic architecture



AutoEncoder – Hyperparameter search space



Hyperparameter Optimisation Tools

- Software environment → TensorFlow with Keras
- HPO software
 - Keras Tuner
 - Random Search
 - Hyperband
 - Optuna
 - pruning – early stopping unpromising configurations
 - no pruning
 - RayTune
 - ASHA – early stopping unpromising configurations
 - no ASHA



Simulated scenarios

Environment of fish breeding ponds

- **Features:** water temperature, water transparency, pH level, dissolved oxygen, nitrates, phosphates, pond depth, vegetation density, average fish weight
- **Outcome:** pond types: [carp , tench , trout ] → classification

Balanced class count



Imbalanced class count



Imbalanced class count



100



500



10 000



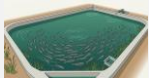
100



500



10 000



100



500



10 000



Training

- data → Training : Validation : Test

- Adam

- Weighted loss

0.5 · reconstruction[mse] + **1.0** · classification[sparse_categorical_crossentropy]

- Metrics

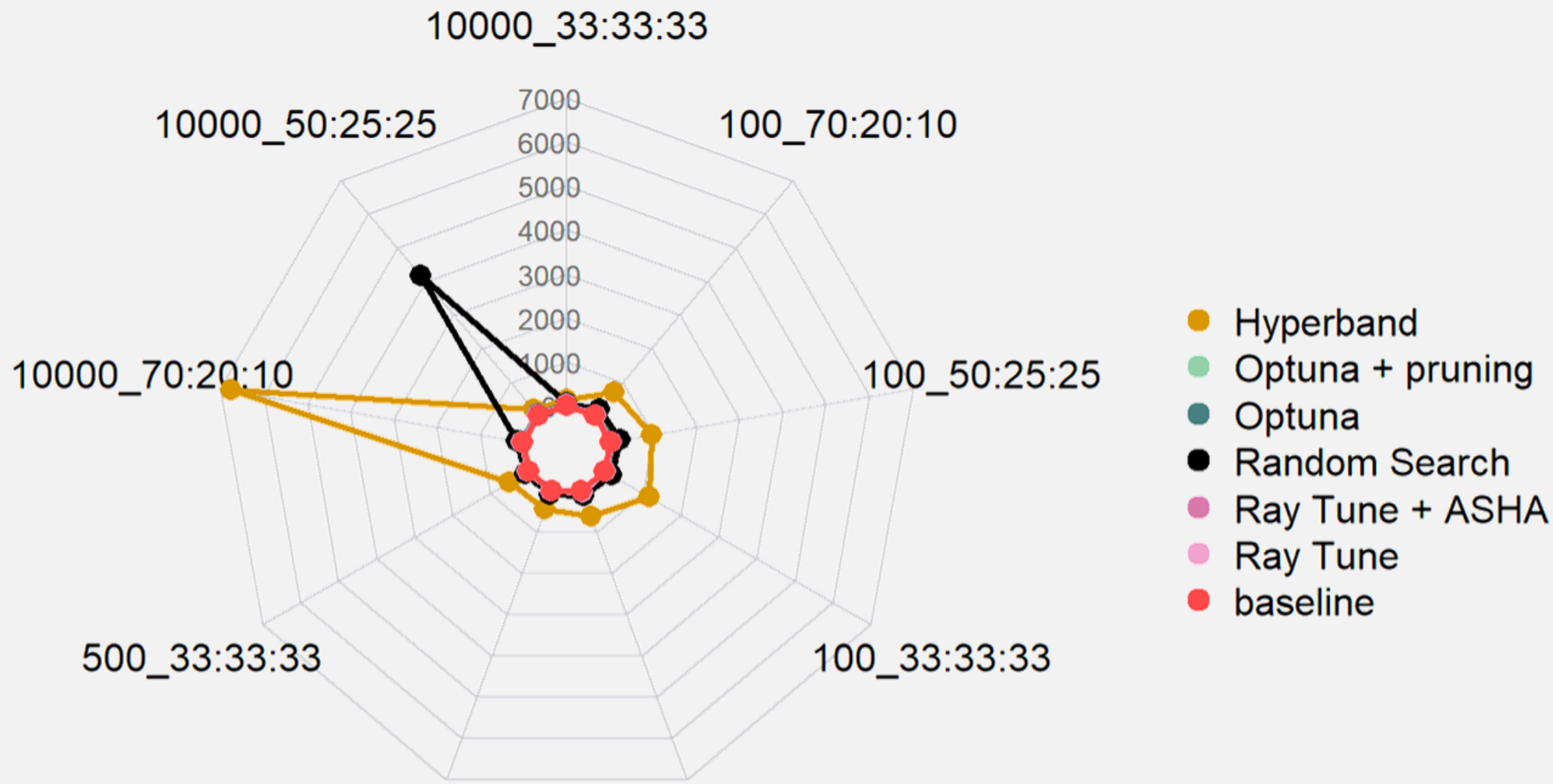
 - F1

 - Total training memory

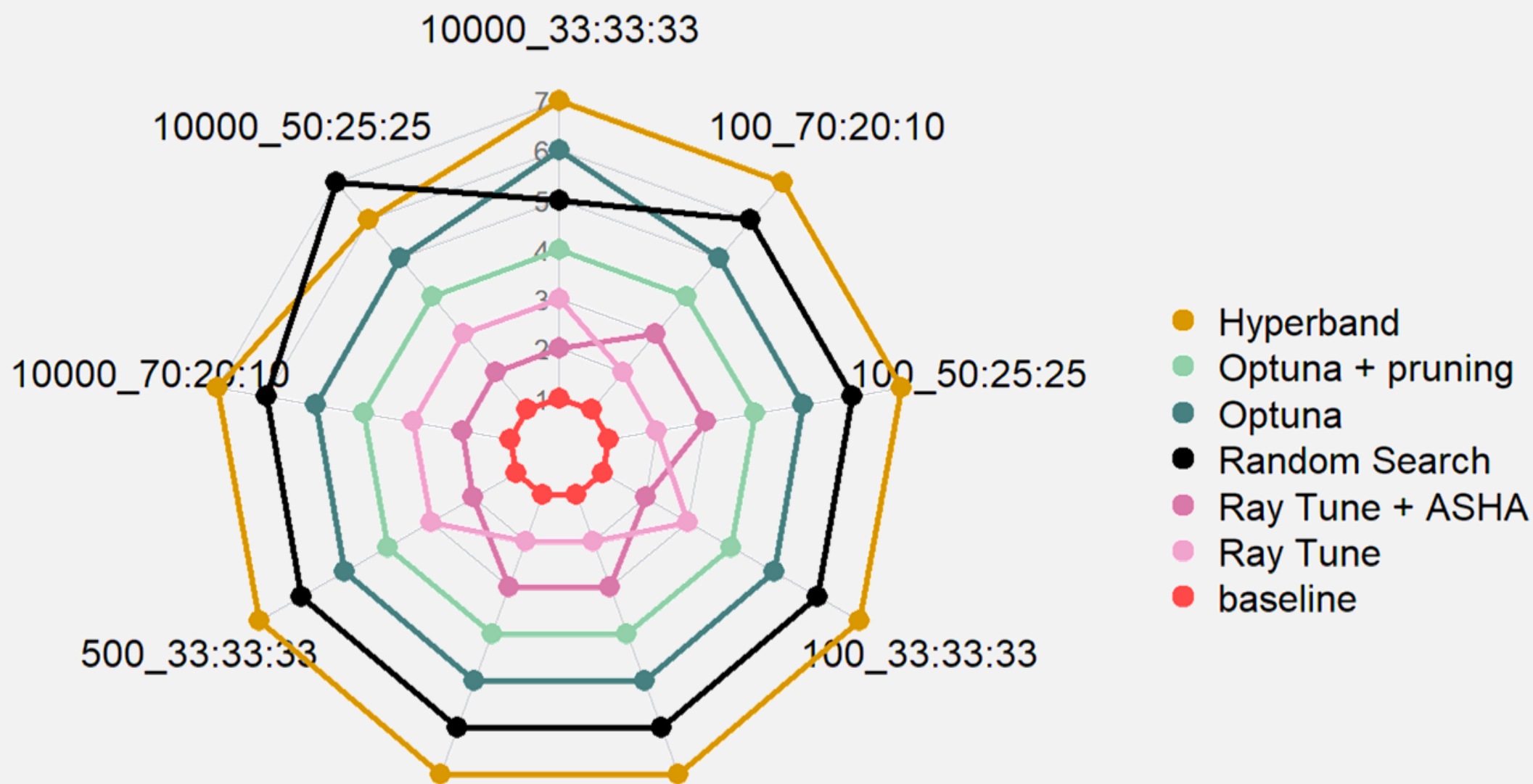
 - Training time



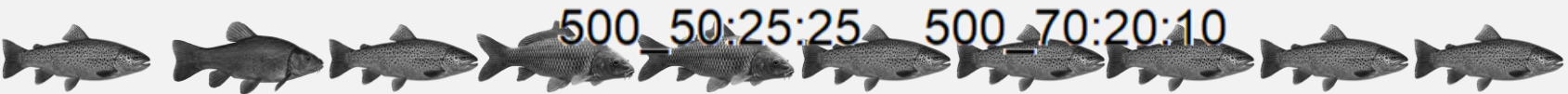
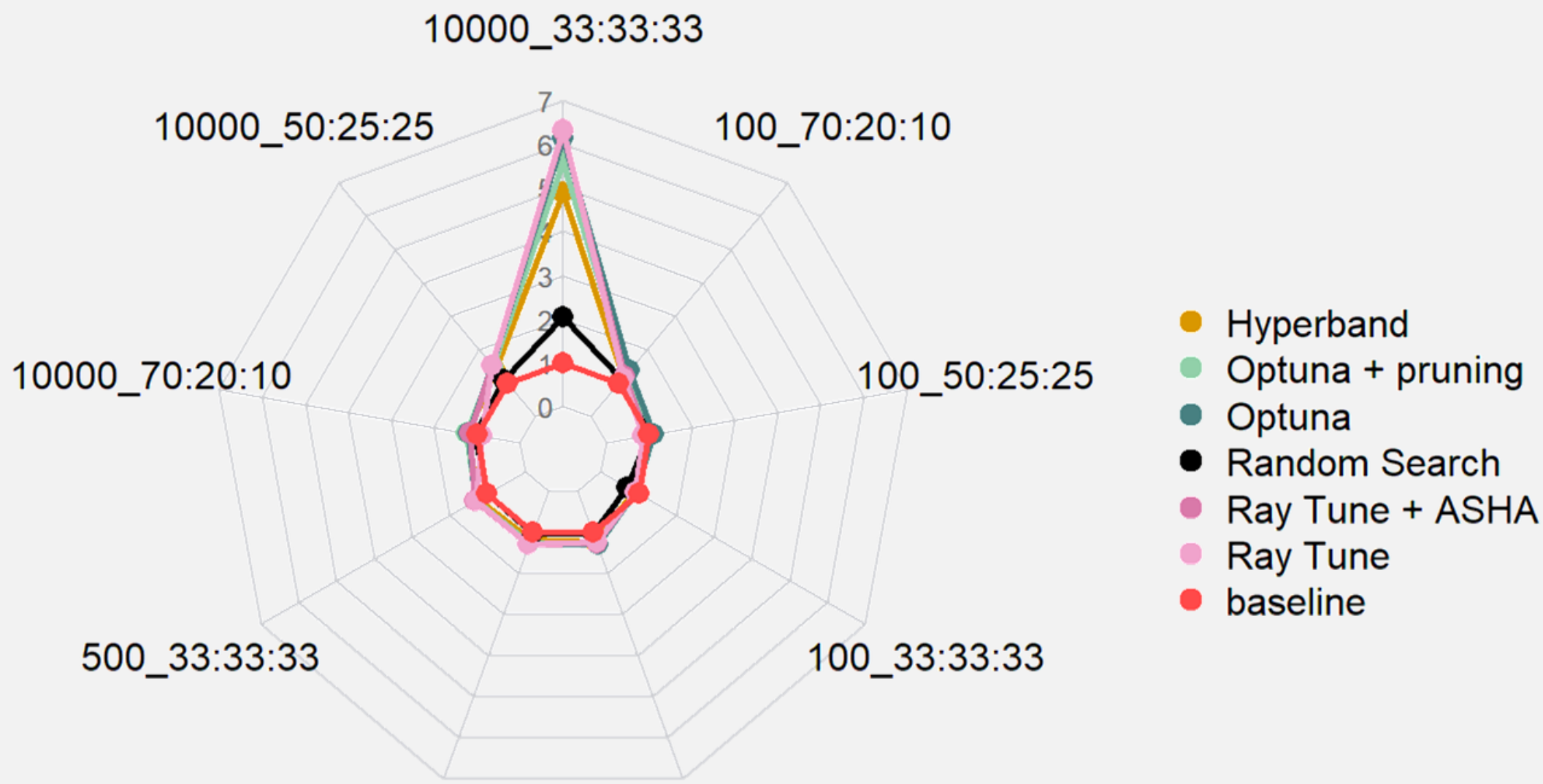
Results – Computing time relative to **baseline**



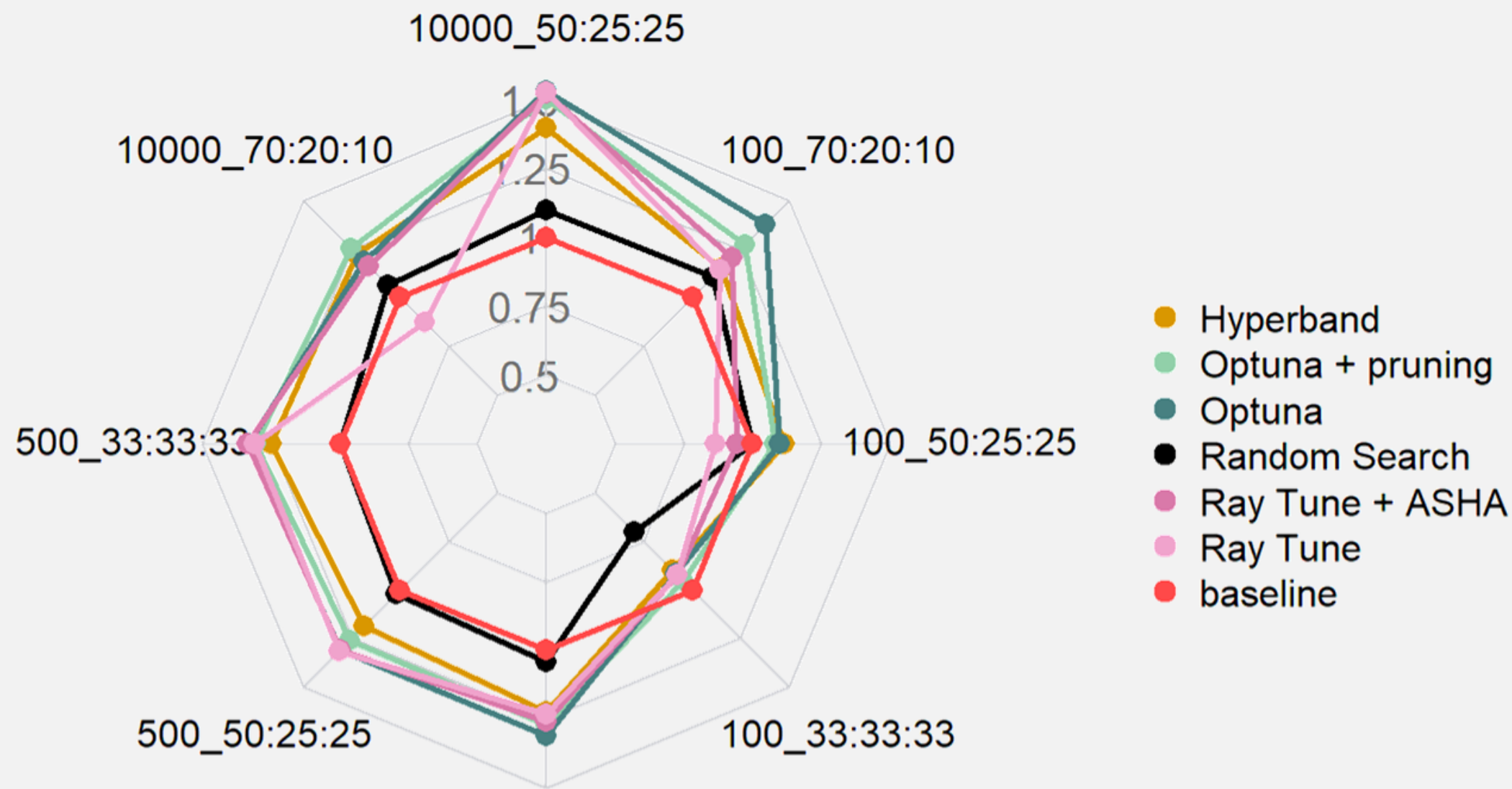
Results – Ranking of computing time



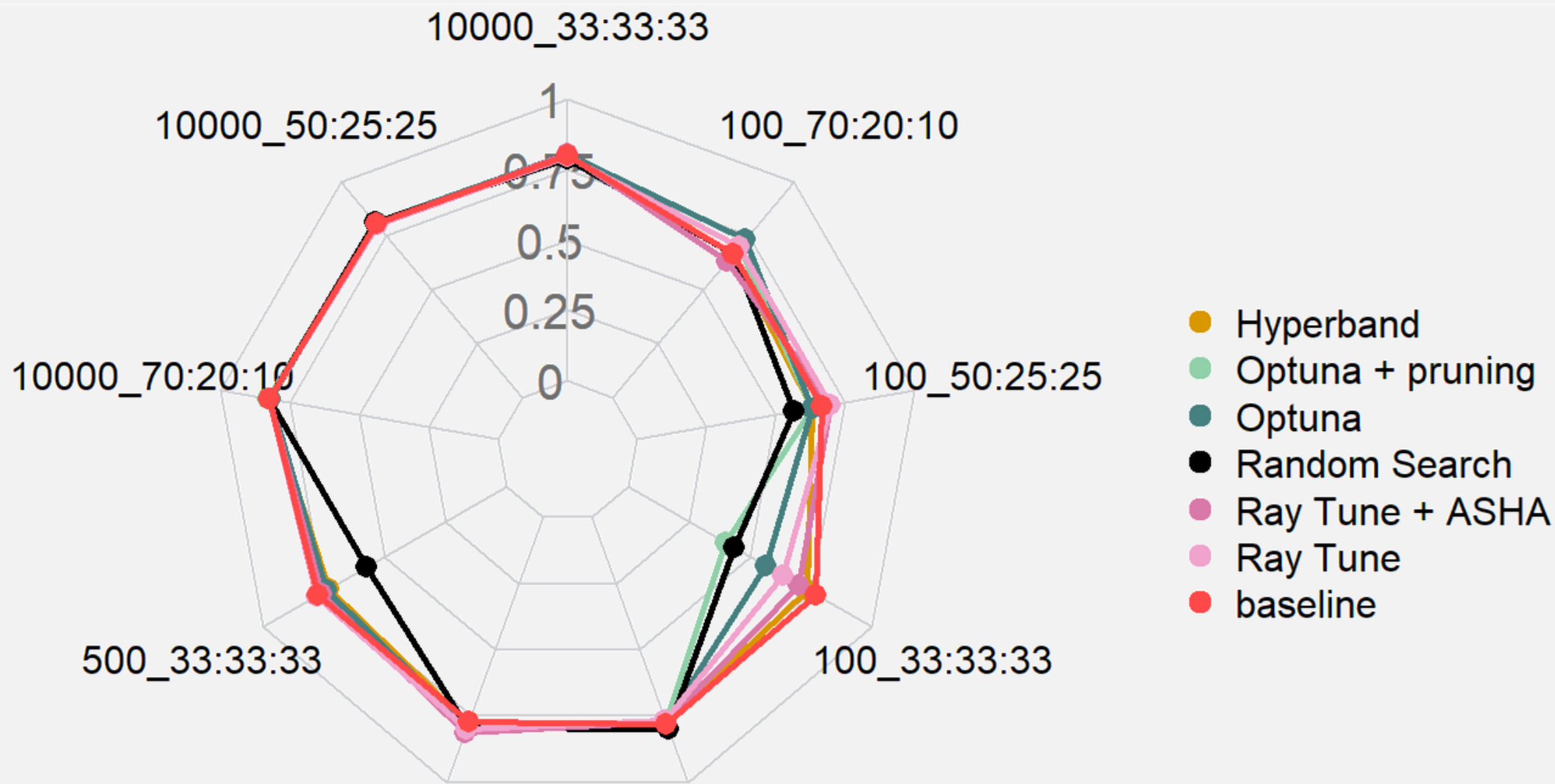
Results – Total memory consumption relative to **baseline**



Results – Total memory consumption relative to **baseline**



Results – F1



Conclusions

1. Optimizing hyperparameters → 12 sec - 1.5 h longer
 - Ray Tune → fastest, Hyperband → slowest
2. Increased memory consumption → no issue
3. ... but
 - No gain in classification quality (F1)
4. ... why?
 - Likelihood over hyperparameters is flat
 - many configurations results in similar classification performance
 - „simple” data set with no correlations between features
5. Optimization only within the fixed architecture
6. More global model optimization via AutoML ???



Thank you for attention

HYPERPARAMETER TUNE RESULTS

class_units: 48, encoding_dim: 10, lr: 0.002661901888489057, units_1: 64, units_2: 8
class_units: 40, encoding_dim: 8, lr: 0.0006647135865318024, units_1: 128, units_2: 8
batch_size: 64, class_units: 16, encoding_dim: 6, **learning_rate: 0.005**, units_1: 32, units_2: 16
class_units: 40, encoding_dim: 12, learning_rate: 0.0009447404851439078, tuner/bracket: 1, tuner/epochs: 17, tuner/initial_epoch: 0, tuner/round: 0, units_1: 80, units_2: 64
class_units: 32, encoding_dim: 6, learning_rate: 0.004539092158444034, units_1: 112, units_2: 56
class_units: 8, encoding_dim: 12, learning_rate: 0.007902373711581125, units_1: 128, units_2: 40
class_units: 64, encoding_dim: 6, lr: 0.001013553491857069859, units_1: 96, units_2: 48
class_units: 16, **16** **UNITS** **116** units_1: 48
class_units: 36, **11** **UNITS** **116** units_2: 32
class_units: 64, encoding_dim: 12, learning_rate: 0.001013553491857069859, units_1: 32, units_2: 64
32 **class_units: 36** **128**
class_units: 84, encoding_dim: 12, learning_rate: 0.007902373711581125, tuner/bracket: 3, tuner/epochs: 17, tuner/initial_epoch: 6, tuner/round: 2, tuner/trial_id: "0034", units_1: 128, units_2: 40
class_units: 56, **0.54** **LEARNING_RATE** **117** units_1: 128, units_2: 40
class_units: 64, **0.54** **LEARNING_RATE** **117** units_1: 96, units_2: 48
class_units: 8, **encoding_dim: 6, lr: 0.009476210031011317064351** units_1: 64, units_2: 48
batch_size: 64, class_units: 16, encoding_dim: 6, learning_rate: 0.005, units_1: 32, units_2: 16
class_units: 16, encoding_dim: 12, learning_rate: 0.006042606682636627, tuner/bracket: 2, tuner/epochs: 17, tuner/initial_epoch: 6, tuner/round: 1, tuner/trial_id: "0058", units_1: 48, units_2: 64
class_units: 16, encoding_dim: 6, learning_rate: 0.000414979578989159, units_1: 80, units_2: 56
class_units: 32, encoding_dim: 6, learning_rate: 0.0013025105444374038, units_1: 112, units_2: 48
class_units: 16, encoding_dim: 10, learning_rate: 0.002369069534544757, units_1: 32, units_2: 32
class_units: 64, encoding_dim: 12, lr: 0.0001175601090023184, units_1: 80, units_2: 16
class_units: 64, encoding_dim: 6, lr: 0.009476210031010521, units_1: 96, units_2: 48
batch_size: 64, class_units: 16, encoding_dim: 6, learning_rate: 0.005, units_1: 32, units_2: 16

