Learning Meta-Features for AutoML

Herilalaina Rakotoarison, Louisot Milijaona,
Andry Rasoanaivo, Michele Sebag, Marc Schoenauer

ICLR 2022

AutoML: Automated Machine Learning

Goal: delivering peak performance on *your* ML task: finding optimal pre-processing, ML algorithm, and hyper-parameters depending on D_{train}, D_{valid}:

$$\text{Find } \theta^* \in \underset{\theta \in \Theta}{\operatorname{argmin}}\, L(\theta, D_{train}, D_{valid}),$$

with Θ the space of ML configurations, and L a loss function.

State-of-art approaches:

- Hyper-parameter optimization[1]
- Meta-learning, i.e. learning to learn:
  - Domain adaptation, Few-shot learning;
  - Learning to describe an ML task:
    - Structured data, via e.g., Deep Neural Networks;
    - Tabular data, via hand-crafted meta-features

[1] e.g. SMAC [Hutter et al., 2011] and HYPEROPT [Bergstra et al., 2011]

AutoML: Automated Machine Learning

Goal: delivering peak performance on *your* ML task: finding optimal pre-processing, ML algorithm, and hyper-parameters depending on D_{train}, D_{valid}:

$$\text{Find } \theta^* \in \operatorname*{argmin}_{\theta \in \Theta} L(\theta, D_{train}, D_{valid}),$$

with Θ the space of ML configurations, and L a loss function.

State-of-art approaches:

- Hyper-parameter optimization[1]
- Meta-learning, i.e. learning to learn:
  - Domain adaptation, Few-shot learning;
  - Learning to describe an ML task:
    - Structured data, via e.g., Deep Neural Networks;
    - Tabular data, via hand-crafted meta-features (This paper)

[1] e.g. SMAC [Hutter et al., 2011] and HYPEROPT [Bergstra et al., 2011]

Hand-crafted meta-features for tabular data

Circa 135 hand-crafted (HC) meta-features[2] (MF) designed by experts since 1980s:

- shallow features: number of instances, number of classes
- statistical features: entropy, average mutual information of features with target
- landmarks: performance of inexpensive classifiers (e.g., Decision Tree)

Limitation: HC MFs insufficiently expressive to support AutoML.

[2] Survey of ML meta-features: [Alcobaca et al., 2020, Rivolli et al., 2022]

Limitation of the hand-crafted meta-features

Given: $\mathcal{X}$, the space of hand-crafted meta-features ($\mathbb{R}^{135}$)
Θ, the configuration space

Limitation of the hand-crafted meta-features

Given: $\mathcal{X}$, the space of hand-crafted meta-features ($\mathbb{R}^{135}$)
Θ, the configuration space

1. Consider datasets A, B, and C, with x_A, x_B, x_C their representation in $\mathcal{X}$

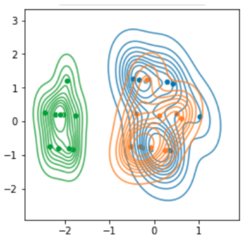
Limitation of the hand-crafted meta-features

Given: $\mathcal{X}$, the space of hand-crafted meta-features ($\mathbb{R}^{135}$)
Θ, the configuration space

1. Consider datasets A, B, and C, with x_A, x_B, x_C their representation in $\mathcal{X}$
2. Let $\Theta_A \subset \Theta$ the set of top config. for A; define (with δ the Dirac function):

$$z_A = \frac{1}{|\Theta_A|} \sum_{\theta \in \Theta_A} \delta_\theta$$

Define similarly z_B and z_C.



Configuration space Θ. Points are top configurations of A, B, and C.

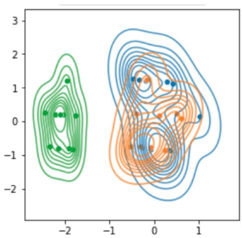
Limitation of the hand-crafted meta-features

Given: $\mathcal{X}$, the space of hand-crafted meta-features ($\mathbb{R}^{135}$)
Θ, the configuration space

1. Consider datasets A, B, and C, with x_A, x_B, x_C their representation in $\mathcal{X}$
2. Let $\Theta_A \subset \Theta$ the set of top config. for A; define (with δ the Dirac function):

$$\mathrm{z}_A = \frac{1}{|\Theta_A|} \sum_{\theta \in \Theta_A} \delta_\theta$$

   Define similarly z_B and z_C.
3. Find nearest neighbor of A w.r.t:



Configuration space Θ. Points are top configurations of A, B, and C.

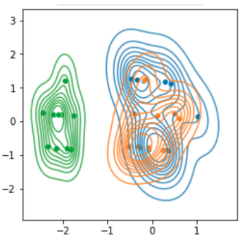
Limitation of the hand-crafted meta-features

Given: $\mathcal{X}$, the space of hand-crafted meta-features ($\mathbb{R}^{135}$)
Θ, the configuration space

1. Consider datasets A, B, and C, with x_A, x_B, x_C their representation in $\mathcal{X}$
2. Let $\Theta_A \subset \Theta$ the set of top config. for A; define (with δ the Dirac function):

$$z_A = \frac{1}{|\Theta_A|} \sum_{\theta \in \Theta_A} \delta_\theta$$

   Define similarly z_B and z_C.
3. Find nearest neighbor of A w.r.t:



Configuration space Θ. Points are top configurations of A, B, and C.

- Euclidean distance over $\mathcal{X} \longrightarrow C$ is the nearest neighbor of A

Limitation of the hand-crafted meta-features

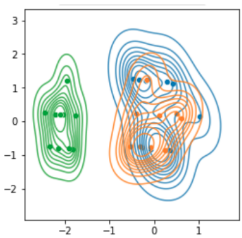
Given: $\mathcal{X}$, the space of hand-crafted meta-features ($\mathbb{R}^{135}$)
Θ, the configuration space

1. Consider datasets A, B, and C, with x_A, x_B, x_C their representation in $\mathcal{X}$
2. Let $\Theta_A \subset \Theta$ the set of top config. for A; define (with δ the Dirac function):

$$\mathrm{z}_A = \frac{1}{|\Theta_A|} \sum_{\theta \in \Theta_A} \delta_\theta$$

   Define similarly z_B and z_C.

3. Find nearest neighbor of A w.r.t:
   - Euclidean distance over $\mathcal{X} \longrightarrow C$ is the nearest neighbor of A
   - Wasserstein distance of $\mathbf{z}$'s $\longrightarrow B$ is the nearest neighbor of A



Configuration space Θ. Points are top configurations of A, B, and C.

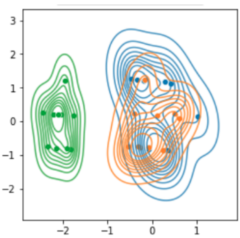
Limitation of the hand-crafted meta-features

Given: $\mathcal{X}$, the space of hand-crafted meta-features ($\mathbb{R}^{135}$)
Θ, the configuration space

1. Consider datasets A, B, and C, with x_A, x_B, x_C their representation in $\mathcal{X}$
2. Let $\Theta_A \subset \Theta$ the set of top config. for A; define (with δ the Dirac function):

$$z_A = \frac{1}{|\Theta_A|} \sum_{\theta \in \Theta_A} \delta_\theta$$

   Define similarly z_B and z_C.
3. Find nearest neighbor of A w.r.t:



Configuration space Θ. Points are top configurations of A, B, and C.

- Euclidean distance over $\mathcal{X}$ $\longrightarrow$ C is the nearest neighbor of A
- Wasserstein distance of $\mathbf{z}$'s $\longrightarrow$ B is the nearest neighbor of A

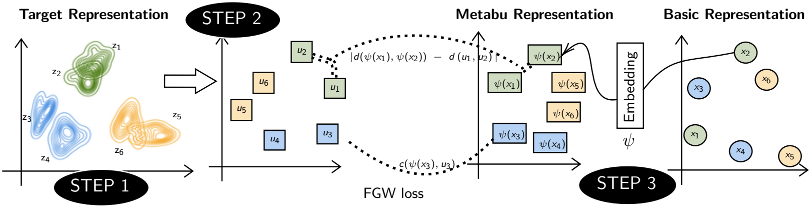
Align topology of HC MFs with the topology of top configurations.

How to proceed ?

METABU: Meta-learning for Tabular Data

With each dataset D, associate:

- Target representation z_D = distribution of top configurations for D (defined on Θ).
- Basic representation x_D = Hand-crafted meta-features for D ($x_D \in \mathcal{X}$).
- METABU representation = linear combinations of HC meta-features.



How to proceed ?

METABU: Meta-learning for Tabular Data

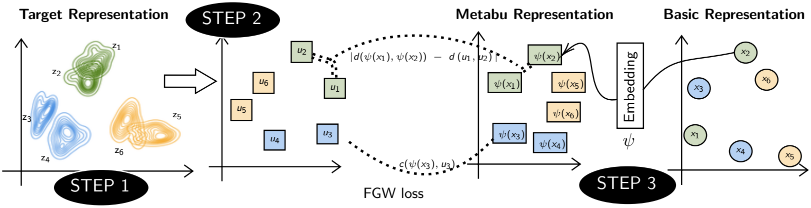
With each dataset D, associate:

- Target representation z_D = distribution of top configurations for D (defined on Θ).
- Basic representation x_D = Hand-crafted meta-features for D ($x_D \in \mathcal{X}$).
- METABU representation = linear combinations of HC meta-features.

Learn METABU meta-features by Optimal Transport such that

METABU neighborhoods ≈ target neighborhoods

.



Empirical assessment of METABU MFs

Configuration spaces: Random Forest, Adaboost, SVM, AutoSklearn.

Baselines (Hand-crafted meta-features)[3]**:** AutoSklearn, SCOT, Landmark, and Random meta-features.

[3] [Feurer et al., 2015, Bardenet et al., 2013, Pfahringer et al., 2000]

[4] [Feurer et al., 2015, Fusi et al., 2018]

Empirical assessment of METABU MFs

Configuration spaces: Random Forest, Adaboost, SVM, AutoSklearn.

Baselines (Hand-crafted meta-features)[3]: AutoSklearn, SCOT, Landmark, and Random meta-features.

Task 1: Assess topology defined by METABU meta-features.

compare METABU-based with target representation-based neighbors;
measure NDCG (information retrieval criterion).

[3] [Feurer et al., 2015, Bardenet et al., 2013, Pfahringer et al., 2000]
[4] [Feurer et al., 2015, Fusi et al., 2018]

Empirical assessment of METABU MFs

Configuration spaces: Random Forest, Adaboost, SVM, AutoSklearn.

Baselines (Hand-crafted meta-features)[3]: AutoSklearn, SCOT, Landmark, and Random meta-features.

Task 1: Assess topology defined by METABU meta-features.

compare METABU-based with target representation-based neighbors; measure NDCG (information retrieval criterion).

Task 2: Assess configurations recommended after METABU.

sample top configurations of METABU-based dataset neighbors; measure average rank of performances.

[3] [Feurer et al., 2015, Bardenet et al., 2013, Pfahringer et al., 2000]

[4] [Feurer et al., 2015, Fusi et al., 2018]

Empirical assessment of METABU MFs

Configuration spaces: Random Forest, Adaboost, SVM, AutoSklearn.

Baselines (Hand-crafted meta-features)[3]: AutoSklearn, SCOT, Landmark, and Random meta-features.

Task 1: Assess topology defined by METABU meta-features.

compare METABU-based with target representation-based neighbors; measure NDCG (information retrieval criterion).

Task 2: Assess configurations recommended after METABU.

sample top configurations of METABU-based dataset neighbors; measure average rank of performances.

Task 3: Assessing METABU on warm-starting optimization algorithms.

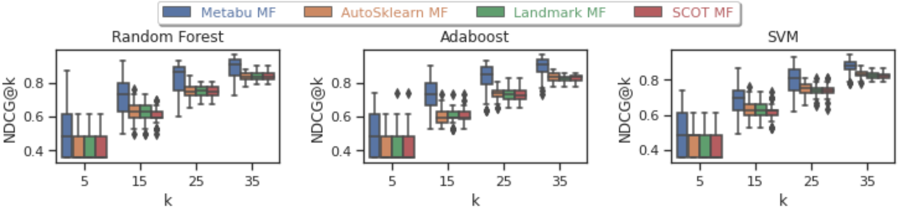
Initialise AutoML optimization (AutoSklearn, PMF)[4] with METABU recommended configurations; measure average rank of performances.

---

[3] [Feurer et al., 2015, Bardenet et al., 2013, Pfahringer et al., 2000]

[4] [Feurer et al., 2015, Fusi et al., 2018]

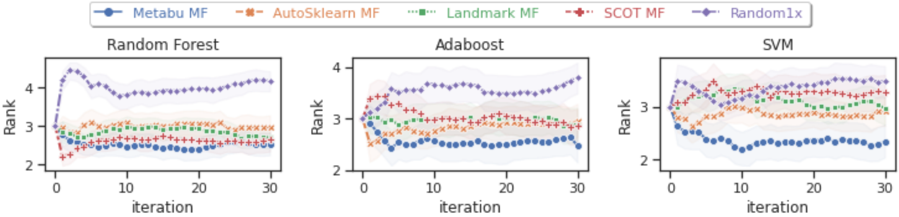
Task 1: Assess topology defined by METABU meta-features



Dataset neighborhood induced by METABU better captures the neighborhood on the target representation.

(NDCG, the higher the better)

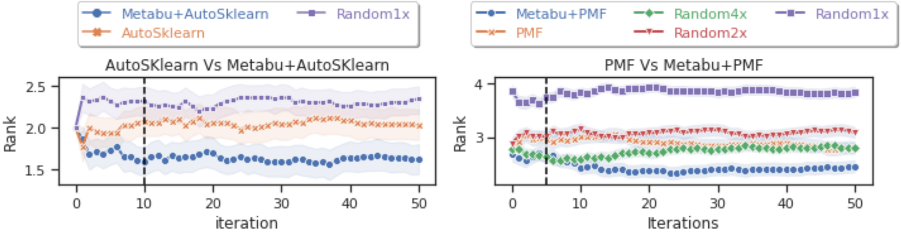
Task 2: Assess configurations recommended after METABU



Configurations, sampled according to the METABU neighborhood, are outperforming the ones sampled by baseline neighborhoods.

(Average rank, the lower the better)

Task 3: Assessing METABU on warm-starting optimization algorithms



Using METABU meta-features to initialize AUTOSKLEARN and PMF search consistently improves over current AUTOSKLEARN and PMF.

(Average rank, the lower the better)

Discussion

Source code publicly available at https://github.com/luxusg1/metabu

MeTaBu:

- learns linear combinations of the HC meta-features.
- captures the topology of target representation, i.e., top configurations.
- outperforms SOTA meta-features on various configuration spaces.

Not discussed:

- Estimating dimension of METABU meta-features ?
- Combatting over-fitting (as benchmark sizes are limited) ?
- Interpreting the found meta-features: what matters for an ML algorithm ?

Come and visit us: Spotlight #6789 and Poster session #6788.

References I

Alcobaca, E., Siqueira, F., Rivolli, A., Garcia, L. P. F., Oliva, J. T., and de Carvalho, A. C. P. L. F. (2020).
MFE: Towards reproducible meta-feature extraction.
JOURNAL OF MACHINE LEARNING RESEARCH, 21.

Bardenet, R., Brendel, M., Kégl, B., and Sebag, M. (2013).
Collaborative hyperparameter tuning.
In *International Conference on Machine Learning*, pages 199–207. PMLR.

Bergstra, J., Bardenet, R., Bengio, Y., and Kégl, B. (2011).
Algorithms for Hyper-Parameter Optimization.
In *Advances in Neural Information Processing Systems*, volume 24. Curran Associates, Inc.

References II

Feurer, M., Klein, A., Eggensperger, K., Springenberg, J., Blum, M., and Hutter, F. (2015).
Efficient and Robust Automated Machine Learning.
In *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc.

Fusi, N., Sheth, R., and Elibol, M. (2018).
Probabilistic Matrix Factorization for Automated Machine Learning.
In *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc.

References III

Hutter, F., Hoos, H. H., and Leyton-Brown, K. (2011).
Sequential Model-Based Optimization for General Algorithm Configuration.
In Coello, C. A. C., editor, *Learning and Intelligent Optimization*, Lecture Notes in Computer Science, pages 507–523, Berlin, Heidelberg. Springer.

Pfahringer, B., Bensusan, H., and Giraud-Carrier, C. G. (2000).
Meta-Learning by Landmarking Various Learning Algorithms.
In Langley, P., editor, *Proceedings of the Seventeenth International Conference on Machine Learning (ICML 2000), Stanford University, Stanford, CA, USA, June 29 - July 2, 2000*, pages 743–750. Morgan Kaufmann.

References IV

Rivolli, A., Garcia, L. P. F., Soares, C., Vanschoren, J., and de Carvalho, A. C. P. L. F. (2022).
Meta-features for meta-learning.
Knowledge-Based Systems, 240:108101.