Cross-data Automatic Feature Engineering via Meta-learning and Reinforcement Learning

Jianyu Zhang¹, Jianye Hao¹, Francoise Fogelman-Soulié²

¹College of Intelligence and Computing, Tianjin University, China
²Hub France IA, Paris, France

PAKDD 2020
May 13th, 2020
Online
Contents

• Problem Overview
• Related Work
• Methods
• Experiments
• Conclusion
Problem Overview

• What is Feature Engineering?
  – Create new features to help machine learning algorithm make better use of the data.

https://en.wikipedia.org/wiki/Feature_engineering
Problem Overview

- Where is the Problem of traditional Feature Engineering?
  - Requires expert knowledge
    - This is hard and time-consuming
- We propos CAFEM:
  - Formulate Feature Engineering by *Feature Transformation Graph* (FTG)
  - Learn to *automatically* generate features for a dataset by Reinforcement Learning (FeL).
  - Extend FeL to cross-data level by Meta-learning.
Related Work

- **Top-down approach**
  - Generate all candidate features, then feature selection; Costly
  - Examples
    - *AFEM* (J. Zhang et al. WISE 2018), *ExploreKit* (Katz et al. ICDM 2016), Data Science Machine *DSM* (Kanter et al. DSAA 2015) and One Button Machine *OneBM* (Lam et al. arXiv 2017).

- **Bottom-up approach**
  - Features are progressively added and evaluated
  - Examples
    - *LFE* (Nargesian et al. IJCAI 2017), *Cognito* (Khurana et al. ICDMW 2016), *FERL* (Khurana et al. AAAI-18)
Background

• Reinforcement Learning (RL)
  – Markov Decision Process
  – States, Actions and Rewards
  – Optimal sequence of actions

• Meta-Learning
  – Quickly train a model for a new task
  – Model-Agnostic Meta-Learning (MAML)
  – Find parameters $\theta$ that is close to all tasks’ optimal parameters
Methods

• Feature Transformation Graph (FTG)
  – Node: an original feature or generated feature
  – Edge: an operator (e.g. log, product) that transforms one/two features to a new feature

• We defined a set of Order-1 (e.g. square) and Order-2 operators (e.g. product)
Methods

• Feature Transformation Graph (FTG)
  – Node: an original feature or generated feature
  – Edge: an operator (e.g. log, product) that transforms one/two features to a new feature
• We defined a set of Order-1 (e.g. square) and Order-2 operators (e.g. product)
Methods

• Learn Feature Engineering by RL (FeL)
  – State:
    • Current FTG
    • We represent current FTG by a set of features
    • Such as # of each operators in FTG, node depth of a feature, average performance improvement of an action.
    • In total, we use 293 features to represent each state.
  – Action
    • Feature Generation: selects an operator and one/two features in FTG, then apply the operator on the features.
    • Feature selection: eliminates an feature from FTG.
Methods

– Reward
  • Performance improvements of classification/regression tasks (evaluation step) after applying an action.

– Budget: Total # of evaluation steps.
  • Evaluation steps is costly.
  • We train RL within the budget.
• Cross-data extension by Meta-Learning (CAFEM)
  – Speed up training by learning on a set of datasets
  – Model-Agnostic Meta-Learning
  – Find parameters $\theta$ on a set of datasets, so that it close to optimal parameters $\theta_i^*$ of all individual datasets.

https://arxiv.org/pdf/1703.03400.pdf
Experiments

• Collect 120 classification/regression datasets from OpenML https://www.openml.org

• 13 transformation operators:
  – Order-1 (Log, Round, Sigmoid, Tanh, Square, Square Root, ZScore, Min-Max- Normalization )
  – Order-2 (Sum, Difference, Product, Division)

• Baseline methods:
  – Random-FeL, Brute-force, LFE, FERL

• Evaluation metrics:
  – F1-Score / 1 - Relevant Absolute Error
## Experiments

- **FeL classification/regression Performance**
  (Random forest with 5-fold CV)

| Datasets        | #Row | #Feature | Baseline | FeL     | BF     | LFE    | RS     | FERL   | FeL     | BF     | LFE    | RS     | FERL   |
|-----------------|------|----------|----------|---------|--------|--------|--------|--------|---------|--------|--------|--------|--------|--------|
| Balance_scale   | 625  | 5        |          | 88.2%   | 88.3%  | 86.4%  | 88.2%  | 88.2%  | 88.6%   | 95.0%  | 97.0%  | 95.1%  | 92.7%  | -      |
| Boston          | 506  | 21       |          | 88.2%   | 90.2%  | 86.7%  | 89.2%  | 89.5%  | 88.7%   | 89.9%  | 85.6%  | 88.2%  | 89.8%  | -      |
| ClimateModel    | 540  | 21       |          | 95.5%   | 96.0%  | 95.6%  | 95.5%  | 95.7%  | 95.9%   | 96.1%  | 95.5%  | 95.5%  | 96.1%  | -      |
| Cpu_small       | 8,192| 13       |          | 86.3%   | 87.1%  | 84.5%  | 85.8%  | 86.6%  | 86.8%   | 87.1%  | 86.2%  | 86.3%  | 87.0%  | -      |
| Credit card     | 14,240| 31      |          | 50.5%   | 68.7%  | 64.8%  | 50.5%  | 63.8%  | 64.0%   | 71.4%  | 65.1%  | 65.1%  | 64.6%  | -      |
| Disclosure_x    | 662  | 4        |          | 44.8%   | 51.7%  | 46.6%  | 46.8%  | 49.7%  | 49.8%   | 51.4%  | 46.4%  | 46.4%  | 51.4%  | 51.8%  |
| Disclosure_z    | 662  | 4        |          | 53.8%   | 57.7%  | 55.6%  | 53.1%  | 55.6%  | 57.0%   | 57.0%  | 53.8%  | 55.0%  | 56.7%  | 56.9%  |
| fri.c1_1000.25  | 1,000| 26       |          | 84.9%   | 87.7%  | 85.8%  | 85.8%  | 86.7%  | 88.0%   | 87.1%  | 77.9%  | 82.1%  | 87.1%  | -      |
| Fri.c2_100.10   | 1,000| 11       |          | 86.3%   | 89.7%  | 85.8%  | 86.8%  | 88.6%  | 89.3%   | 91.0%  | 87.2%  | 86.7%  | 89.3%  | -      |
| Fri.c3_100.5    | 1,000| 6        |          | 88.2%   | 89.2%  | 88.5%  | 88.2%  | 88.4%  | 89.4%   | 90.7%  | 87.3%  | 87.1%  | 89.3%  | -      |
| fri.c3.1000.50  | 1,000| 51       |          | 79.7%   | 83.7%  | 88.5%  | 80.9%  | 80.7%  | 87.8%   | 83.1%  | 88.4%  | 78.3%  | 80.8%  | -      |
| Gina.agnostic   | 3,468| 971      |          | 92.3%   | 92.8%  | 78.9%  | 92.3%  | 92.8%  | 93.5%   | 92.8%  | -      | 92.5%  | 92.8%  | -      |
| Hill-valley     | 1,212| 101      |          | 57.5%   | 61.7%  | 59.2%  | 57.5%  | 60.8%  | 61.1%   | 100%   | 100%   | 57.5%  | 99.9%  | -      |
| Ipd            | 583  | 11       |          | 41.3%   | 45.7%  | 38.7%  | 38.9%  | 43.6%  | 44.9%   | 45.9%  | 45.9%  | 42.4%  | 44.8%  | -      |
| Kcl            | 2,109| 22       |          | 40.4%   | 44.5%  | 35.3%  | 38.9%  | 42.0%  | 42.7%   | 44.4%  | 39.9%  | 38.8%  | 43.4%  | -      |
| openml.589     | 1,000| 25       |          | 66.9%   | 67.7%  | 55.0%  | X      | 67.2%  | 72.6%   | 75.0%  | 76.9%  | X      | 68.1%  | -      |
| Pcc            | 1,458| 38       |          | 47.7%   | 57.0%  | 36.2%  | 45.3%  | 53.8%  | 58.4%   | 58.1%  | 50.1%  | 55.1%  | 56.5%  | -      |
| Pcc3+C14       | 1,563| 38       |          | 25.9%   | 33.4%  | 27.9%  | 23.0%  | 30.3%  | 32.0%   | 33.3%  | 24.6%  | 27.4%  | 31.6%  | -      |
| Spectrometer   | 531  | 103      |          | 77.3%   | 83.9%  | 80.0%  | 75.2%  | 80.4%  | 83.0%   | 82.7%  | 90.8%  | 73.2%  | 81.8%  | -      |
| Strikes        | 625  | 7        |          | 96.6%   | 99.5%  | 98.7%  | 97.8%  | 99.1%  | 98.9%   | 99.5%  | 97.8%  | 93.4%  | 99.4%  | 98.9%  |

PAKDD 2020 May 13th, 2020
Experiments

• Robustness of FeL on learning algorithms
  – Random Forest: 4.2% improvement
  – Logistic Regression: 10.8% improvement
Experiments

• Cross-data Extension (CAFEM) Performance
  – Training on 100 datasets
  – Few shots learning on other 20 datasets.
Conclusion

• Feature engineering influences performances a lot but it is the most time-consuming part of data mining.
• We propose Feature Transformation Graph (FTG) to organize the feature engineering (FE) process and FE learner (FeL) to learn FE by Reinforcement Learning.
• We extend FeL to cross-data level (CAFEM) by meta-learning.
• Our framework out performs state-of-the-art methods.
Thanks for your attention

Q & A