How much data is sufficient to learn high-performing algorithms? Generalization guarantees for data-driven algorithm design

Nina Balcan, Dan DeBlasio, Travis Dick, Carl Kingsford, Tuomas Sandholm, Ellen Vitercik

Data-driven algorithm design

Algorithms often have many tunable parameters

Significant impact on runtime, solution quality, ...

Hand-tuning is time-consuming, tedious, and error prone

Goal: Automate algorithm configuration via machine learning

Input: Training set of typical problem instances from application at hand

Output: Configuration with strong average empirical performance on training set

Prior research proved generalization bounds case

Configuration

Online algorithm configuration:

Sandholm, Gupta, Roughgarden, ITCS’16; Performance is a Piecewise constant

Balcan, Gupta, White, COLT’17; Balcan, Dick, Sandholm, V, ICML’18; Balcan, Dick, White, NeurIPS’18; Balcan, Dick, Lang, ICLR’20; ...

Parameter setting should—ideally—be good on future inputs

Summary of contributions

Broadly applicable theory for deriving generalization bounds

Algorithm’s average performance on training set ≤ ?

Prior research proved generalization bounds case-by-case

Gupta, Roughgarden, ITCS’16; Balcan, Nagarajan, V, White, COLT’17; Balcan, Dick, Sandholm, V, ICML’18; Balcan, Dick, White, NeurIPS’18; Balcan, Dick, Lang, ICLR’20; ...

Clustering algorithm configuration

Integer programming configuration

Greedy algorithm configuration

Computational biology algorithm configuration

Voting mechanism configuration

Recover bounds

Prove novel bounds

We uncover overarching structure linking these seemingly disparate domains

Guarantees apply to any parameterized algorithm where:

Performance is a piecewise-structured function of parameters

Additional references

• Book chapter by Balcan (Cambridge University Press ’20)
• Online algorithm configuration

Exploited that the dual functions are piecewise Lipschitz to provide regret bounds

[Balcan, Dick, V, FOCS’18; Balcan, Dick, Plegdan, UAI’20; Balcan, Dick, Sharma, AISTATS’20]

Primary challenge

Performance is a volatile function of parameters

Complex connection between parameters and performance

Meanwhile, for well-understood functions in machine learning theory: Simple connection between function parameters and value

Running example: Sequence alignment

Gap

A - - - C T G

- G T C A -

Indel | Mismatch

Match

Standard algorithm with parameters $p_1, p_2, p_3 \geq 0$:

Return alignment maximizing:

$\#$ matches $- p_1 \cdot (\#$ mismatches $- p_2 \cdot (\#$ indels $- p_3 \cdot (\#$ gaps)

*There is considerable disagreement among molecular biologists about the correct choice of $p$.*

[Cambridge University Press ‘20]

Model and problem formulation

$\mathbb{R}^d$: Set of all parameters

$\mathcal{X}$: Set of all inputs (e.g., sequence pairs)

$w_p(x) = \text{utility of algorithm parameterized by } p \in \mathbb{R}^d \text{ on input } x$

Runtime, solution quality, ...

Assume $w_p(x) \in [-1, 1]$

Standard assumption: Unknown distribution $\mathcal{D}$ over inputs

Models specific application domain at hand

Generalization bound: Given samples $x_1, ..., x_N \sim \mathcal{D}$, for any $p$,

$$\frac{1}{N} \sum_{i=1}^{N} w_p(x_i) - \mathbb{E}_x[w_p(x)] \leq ?$$

Empirical average utility

Expected utility

Main result

$\mathcal{U} = \{w_p: \mathcal{X} \to \mathbb{R} \mid p \in \mathbb{R}^d\}$

*“Primal”* function class

Typically, prove generalization guarantees by bounding the complexity of $\mathcal{U}$

VC dimension, Rademacher complexity, ...

Challenge: $\mathcal{U}$ is gnarly. E.g., in sequence alignment:

• Each domain element is a pair of sequences

• Unclear how to plot/visualize functions $w_p$

• No obvious notions of Lipschitzness or smoothness to rely on

This is where dual functions come in handy!

$w_p^*(\mathcal{U}) = \text{utility of function parameters}$

$w_p^*(\mathcal{U}) = \{w_p: \mathcal{X} \to \mathbb{R} \mid x \in \mathcal{X}\}$

*“Dual”* function class

Across algorithm configuration, ubiquitously, the duals are piecewise-structured

Theorem

With high probability, for all $p$:

$$w_p^*(\mathcal{U})$$

Average utility on training set - expected utility $= \hat{O}\left(\frac{\dim(\mathcal{U}) + \text{VC-dimension}(\mathcal{U})}{N}\right)$

# boundary functions

Training set size

Lemma: Given $k$ boundaries, how many sign patterns do they make?

$$\frac{(g_1(p_1)) \cdots (g_k(p_k))}{(g_1(p_k)) \cdots (g_k(p_1))} \leq (ck)^{k\text{diam}(\mathcal{U})})$$

Proof idea: Transition to dual and apply Sauer’s lemma: for any $p_1, ..., p_k$

$$\frac{(g_1(p_1)) \cdots (g_k(p_k))}{(g_1(p_k)) \cdots (g_k(p_1))} \leq (ck)^{k\text{diam}(\mathcal{U})})$$

Example application: Sequence alignment

With high probability, for any $p \in \mathbb{R}^d$

$$\text{avg utility on training set - expected utility} = \hat{O}\left(\frac{\text{runtime}}{N}\right)$$

Distance between algorithm’s output and ground-truth alignment