AdaNet: Adaptive Structural Learning of Artificial Neural Networks

Corinna Cortes[1], Xavier Gonzalvo[1], Vitaly Kuznetsov[1], Mehryar Mohri[21], Scott Yang [2]

[1]Google Research

[2]Courant Institute

ICML, 2017/ Presenter: Anant Kharkar

Outline

1. Introduction
   - Motivation & Contributions
2. Theoretical Background
   - Regularizer Derivation
   - Generalization Bounds
3. AdaNet
   - Basics
   - Algorithm
   - Results
   - Variations

Outline

1 Introduction
- Motivation & Contributions

2 Theoretical Background
- Regularizer Derivation
- Generalization Bounds

3 AdaNet
- Basics
- Algorithm
- Results
- Variations

Motivation & Contributions

Motivation

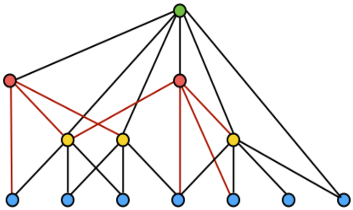
- Lack of theory behind model architecture design
- Usually requires domain knowledge

Contributions

- New regularizer that learns parameters & architecture simultaneously
- Additive model

Context

- Strongly convex optimization problems
- Generic network structure (not just MLP)



Outline

1. Introduction
   - Motivation & Contributions
2. Theoretical Background
   - Regularizer Derivation
   - Generalization Bounds
3. AdaNet
   - Basics
   - Algorithm
   - Results
   - Variations

Regularizer Derivation

Function families:

$$\mathcal{H}_1 = \{x \mapsto \mathbf{u} \cdot \Psi(x) : \mathbf{u} \in \mathbb{R}^{n_0}, \|\mathbf{u}\|_p \leq \Lambda_{1,0}\}$$

$$\mathcal{H}_k = \left\{x \mapsto \sum_{s=1}^{k-1} \mathbf{u}_s \cdot (\varphi_s \circ \mathbf{h}_s)(x) : \mathbf{u}_s \in \mathbb{R}^{n_s}, \|\mathbf{u}_s\|_p \leq \Lambda_{k,s}, h_{k,s} \in \mathcal{H}_s\right\}$$

Definitions:

- x: input
- u: connections to previous layers
- Ψ: feature vector
- φ: activation function

Outline

1 Introduction
- Motivation & Contributions

2 Theoretical Background
- Regularizer Derivation
- Generalization Bounds

3 AdaNet
- Basics
- Algorithm
- Results
- Variations

Generalization Bounds

Rademacher Complexity:

$$\hat{\mathfrak{R}}_S(\mathcal{G}) = \frac{1}{m}\mathop{\mathbb{E}}_{\sigma}\left[\sup_{h\in\mathcal{G}}\sum_{i=1}^{m}\sigma_i h(x_i)\right]$$

Definitions:

- x: input
- h: function expressed by single unit
- σ: Rademacher RV {-1, +1}

Generalization Bounds

Theorem 1:

$$R(f) \leq \hat{R}_{S,\rho}(f) + \frac{4}{\rho}\sum_{k=1}^{l} \|\mathbf{w}_k\|_1 \mathfrak{R}_m(\widetilde{H}_k) + \frac{2}{\rho}\sqrt{\frac{log\ l}{m}} + C(\rho, l, m, \delta)$$

$$C(\rho, l, m, \delta) = \sqrt{\left\lceil \frac{4}{\rho^2} log(\frac{\rho^2 m}{log\ l}) \right\rceil \frac{log\ l}{m} + \frac{log(\frac{2}{\delta})}{2m}}$$

$\frac{4}{\rho}\sum_{k=1}^{l} \|\mathbf{w}_k\|_1 \mathfrak{R}_m(\widetilde{H}_k)$: weighted sum of Rademacher complexities

Generalization Bounds

Corollary 1:

$$R(f) \leq \hat{R}_{S,\rho}(f) + \frac{2}{\rho}\sum_{k=1}^{l} \|w_k\|_1 \left[\bar{r}_\infty \Lambda_k N_k^{\frac{1}{q}} \sqrt{\frac{2 log(2n_0)}{m}}\right]$$

$$+\frac{2}{\rho}\sqrt{\frac{log\ l}{m}} + C(\rho, l, m, \delta)$$

$$C(\rho, l, m, \delta) = \sqrt{\left\lceil \frac{4}{\rho^2} log(\frac{\rho^2 m}{log\ l}) \right\rceil \frac{log\ l}{m} + \frac{log(\frac{2}{\delta})}{2m}}$$

Outline

1 Introduction
- Motivation & Contributions

2 Theoretical Background
- Regularizer Derivation
- Generalization Bounds

3 AdaNet
- Basics
- Algorithm
- Results
- Variations

Basics

- Adaptively grows network structure

Objective function:

$$F(\mathbf{w}) = \frac{1}{m}\sum_{i=1}^{m}\Phi\left(1 - y_i\sum_{j=1}^{N} w_j h_j\right) + \sum_{j=1}^{N}\Gamma_j|w_j|$$

Definitions:

- Φ: nondecreasing convex function (Ex: e^x)
- Γ_j: $\lambda r_j + \beta$
- r_j: $\mathcal{R}_m(\mathcal{H}_{k_j})$ (Rademacher complexity)

This objective function is forcibly convex

Outline

1. Introduction
   - Motivation & Contributions
2. Theoretical Background
   - Regularizer Derivation
   - Generalization Bounds
3. AdaNet
   - Basics
   - Algorithm
   - Results
   - Variations

Algorithm

$\text{AdaNet}(S = ((x_i, y_i)_{i=1}^{m})$

$f_0 \leftarrow 0$

for $t \leftarrow 1$ **to** T **do**

  $\mathbf{h}, \mathbf{h}' \leftarrow \text{WeakLearner}\big(S, f_{t-1}\big)$

  $\mathbf{w} \leftarrow \text{Minimize}\big(F_t(\mathbf{w}, \mathbf{h})\big)$

  $\mathbf{w}' \leftarrow \text{Minimize}\big(F_t(\mathbf{w}, \mathbf{h}')\big)$

  if $F_t(\mathbf{w}, \mathbf{h}') \leq F_t(\mathbf{w}', \mathbf{h}')$ **then**

    $\mathbf{h}_t \leftarrow \mathbf{h}$

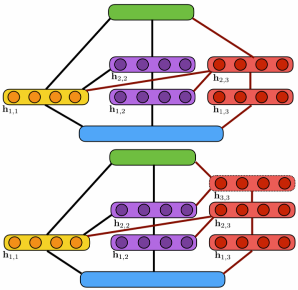
  else $\mathbf{h}_t \leftarrow \mathbf{h}'$

  if $F(\mathbf{w}_{t-1} + \mathbf{w}^*) < F(\mathbf{w}_{t-1})$ **then**

    $f_{t-1} \leftarrow f_t + \mathbf{w}^* \cdot \mathbf{h}_t$

  else return f_{t-1}

return f_T



Outline

1. Introduction
   - Motivation & Contributions
2. Theoretical Background
   - Regularizer Derivation
   - Generalization Bounds
3. AdaNet
   - Basics
   - Algorithm
   - Results
   - Variations

Results

Label pair	ADANET	LR	NN	NN-GP
deer-truck	**0.9372 ± 0.0082**	0.8997 ± 0.0066	0.9213 ± 0.0065	0.9220 ± 0.0069
deer-horse	**0.8430 ± 0.0076**	0.7685 ± 0.0119	0.8055 ± 0.0178	0.8060 ± 0.0181
automobile-truck	**0.8461 ± 0.0069**	0.7976 ± 0.0076	0.8063 ± 0.0064	0.8056 ± 0.0138
cat-dog	**0.6924 ± 0.0129**	0.6664 ± 0.0099	0.6595 ± 0.0141	0.6607 ± 0.0097
dog-horse	**0.8350 ± 0.0089**	0.7968 ± 0.0128	0.8066 ± 0.0087	0.8087 ± 0.0109

Label pair	ADANET		NN	NN-GP
	1st layer	2nd layer		
deer-truck	990	0	2048	1050
deer-horse	1475	0	2048	488
automobile-truck	2000	0	2048	1595
cat-dog	1800	25	512	155
dog-horse	1600	0	2048	1273

Outline

1. Introduction
   - Motivation & Contributions
2. Theoretical Background
   - Regularizer Derivation
   - Generalization Bounds
3. AdaNet
   - Basics
   - Algorithm
   - Results
   - Variations

Variations

- AdaNet.R: $\mathcal{R}(\mathbf{w}, \mathbf{h}) = \Gamma_h \|w\|_1$ regularization in obj.
- AdaNet.P: subnetworks only connected to previous subnetwork
- AdaNet.D: dropout on subnetwork connections
- AdaNet.SD: std dev of final layers instead of Rademacher complexity