Classifier Cascade for Minimizing Feature Evaluation Cost
Minmin Chen
Kilian Weinberger, Olivier Chapelle
Machine Learning for Real-World Applications
Average Computational Cost during Test Time

- hundreds of thousands of documents within a few milliseconds

- millions of messages per day, thus less than 10 milliseconds per email
Differences and Observations

- Computational cost is evaluated on average per test instances
Differences and Observations

- Features are computed on demand, and vary on costs
Goal

- How to build classifiers to achieve high precision with low test time complexity?
Showcase Application: Web Search Ranking

The largest web search ranking competition to date

- All 8 winners out of the 1,055 teams used Gradient Boosted Regression Trees (GBRT) or its variants

\[ D = \{(x_1, y_1), (x_2, y_2), \ldots , (x_n, y_n)\} \subset \mathcal{R}^d \times \{0, 1, 2, 3, 4\} \]
Showcase Application: Web Search Ranking

- The largest web search ranking competition to date
- All 8 winners out of the 1,055 teams used Gradient Boosted Regression Trees (GBRT) or its variants

\[ D = \{(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\} \subset \mathcal{R}^d \times \{0, 1, 2, 3, 4\} \]

0,1,2: irrelevant
3,4: relevant
Gradient Boosted Regression Tree (GBRT)

$\alpha + \alpha + \alpha + \ldots + \alpha + \alpha \ (\alpha = 0.05)$
Feature Evaluation Costs

\[
\alpha \text{ } 1 \text{ } 2 \text{ } 4 \text{ } 7 \text{ } 10 \text{ } 6 + \alpha \text{ } 8 \text{ } 1 \text{ } 5 \text{ } 11 \text{ } 9 \text{ } 3 \text{ } 1 = 0.05
\]

\[
(\alpha = 0.05)
\]
Feature Evaluation Costs

\( \alpha + \alpha + \alpha + \ldots + \alpha \)  

(\( \alpha = 0.05 \))

\begin{align*}
\alpha &= 1 \\
\alpha &= 5 \\
\alpha &= 20 \\
\alpha &= 50 \\
\alpha &= 100 \\
\alpha &= 150 \\
\alpha &= 200 \\
\end{align*}

Minmin Chen  
Classifier Cascade for Minimizing Feature Evaluation Cost  
Washington University in St. Louis

Thursday, January 12, 12
Feature Evaluation Costs

\[
\alpha \cdot 1 + \alpha \cdot 11 + \alpha \cdot 9 + \cdots + \alpha + 1
\]

\(\alpha = 0.05\)

\[
C = 5 + 20 + 1 + 20 + 1 + 5 + 0 + 1
\]
Feature Evaluation Costs

\( \alpha \)

\( + \alpha \)

\( + \alpha \)

\( + \cdots + \alpha \)

\( (\alpha = 0.05) \)

\[ C = 5 + 20 + 1 + 20 + 1 + 5 + 0 + 1 \]

A feature becomes free after it is computed for the first time

Minmin Chen

Classifier Cascade for Minimizing Feature Evaluation Cost

Washington University in St. Louis
Post Processing

\[ \alpha + \alpha + \alpha + \alpha + \cdots + \alpha + \alpha + \alpha \]

\( (\alpha = 0.05) \)

\[ c = 0 \]
\[ c = 1 \]
\[ c = 5 \]
\[ c = 20 \]
\[ c = 50 \]
\[ c = 100 \]
\[ c = 150 \]
\[ c = 200 \]

\[ C = 52 + 1 \]
\[ C = 326 + 1 \]
\[ C = 0 + 1 \]
Post Processing

\[ \alpha \cdot \frac{1}{3} + \alpha \cdot \frac{1}{3} + \alpha \cdot \frac{1}{3} + \ldots + \alpha \cdot \frac{1}{3} + \alpha \cdot \frac{1}{3} \]

\[ (\alpha = 0.05) \]

\[ c = 0 \]
\[ c = 1 \]
\[ c = 5 \]
\[ c = 20 \]
\[ c = 50 \]
\[ c = 100 \]
\[ c = 150 \]
\[ c = 200 \]

\[ C = 52 + 1 \]

\[ C = 326 + 1 \]

\[ C = 0 + 1 \]
Post processing classifiers to reduce their amortized time complexity without sacrificing the performance much.

Minmin Chen                                Classifier Cascade for Minimizing Feature Evaluation Cost                                Washington University in St. Louis
Single Stage Sparse Solution

\[ \alpha \cdot 1 + \alpha \cdot 1 \cdot 2 \cdot 4 \cdot 7 \cdot 10 \cdot 6 + \alpha \cdot 1 \cdot 2 \cdot 5 \cdot 11 \cdot 9 + \alpha \cdot 3 \cdot 11 \cdot 3 \cdot 5 \cdot 9 + \alpha \cdot \alpha \]
Single Stage Sparse Solution
Single Stage Sparse Solution

\[ \beta_1 = 0.1 \quad \beta_2 = 0.0 \quad \beta_3 = 0.02 \quad \theta_{T-1} = 0.05 \quad \theta_T = 0.0 \]
Single Stage Sparse Solution

\[
\begin{align*}
\beta_1 &= 0.1 \\
\beta_2 &= 0.0 \\
\beta_3 &= 0.02 \\
\beta_{T-1} &= 0.05 \\
\beta_T &= 0.0
\end{align*}
\]
Single Stage Sparse Solution

\[ \frac{1}{2} \sum_{i=1}^{n} \left( \sum_{t=1}^{T} \beta_t h_t(x_i) - y_i \right)^2 + \lambda \sum_{t=1}^{T} e_t \delta(\beta_t \neq 0) + \sum_{\alpha=1}^{d} c_\alpha \delta(\sum_{t=1}^{T} F_{\alpha t} \delta(\beta_t \neq 0) \neq 0) + \rho |\beta| \]
Single Stage Sparse Solution

\[ \beta_1 + \beta_2 + \beta_3 + \ldots + \beta_{T-1} + \beta_T \]

\[ \beta_1 = 0.1 \]
\[ \beta_2 = 0.0 \]
\[ \beta_3 = 0.02 \]
\[ \beta_{T-1} = 0.05 \]
\[ \beta_T = 0.0 \]

Loss

\[ \frac{1}{2} \sum_{i=1}^{n} \left( \sum_{t=1}^{T} \beta_t h_t(x_i) - y_i \right)^2 + \lambda \]

Tree cost

\[ \sum_{t=1}^{T} e_t \delta(\beta_t \neq 0) + \sum_{\alpha=1}^{d} c_{\alpha} \delta \left( \sum_{t=1}^{T} F_{\alpha t} \delta(\beta_t \neq 0) \neq 0 \right) + \rho \]

Feature cost

Reg.

Non-continuous 0-1 function

Minmin Chen
Classifier Cascade for Minimizing Feature Evaluation Cost
Washington University in St. Louis

Thursday, January 12, 12
Single Stage Sparse Solution

\[ \beta_1 = 0.1 \]
\[ \beta_2 = 0.0 \]
\[ \beta_3 = 0.02 \]
\[ \beta_{T-1} = 0.05 \]
\[ \beta_T = 0.0 \]

\[
\frac{1}{2} \sum_{i=1}^{n} \left( \sum_{t=1}^{T} \beta_t h_t(x_i) - y_i \right)^2 + \lambda
\]

Tree cost
\[
\sum_{t=1}^{T} e_t \delta(\beta_t \neq 0) + \sum_{\alpha=1}^{d} c_\alpha \delta(\sum_{t=1}^{T} F_\alpha t \delta(\beta_t \neq 0) \neq 0) + \rho \left| \beta \right|
\]

Feature cost
\[
\sum_{t=1}^{T} e_t |\beta_t| + \sum_{\alpha=1}^{d} c_\alpha \sqrt{\sum_{t=1}^{T} F_\alpha t \beta_t^2}
\]

Non-continuous 0-1 function
\[
\frac{1}{2} \sum_{i=1}^{n} \left( \sum_{t=1}^{T} \beta_t h_t(x_i) - y_i \right)^2 + \lambda \sum_{t=1}^{T} e_t |\beta_t| + \sum_{\alpha=1}^{d} c_\alpha \sqrt{\sum_{t=1}^{T} F_{\alpha t} \beta_t^2} + \rho |\beta|
\]
\[
\frac{1}{2} \sum_{i=1}^{n} \left( \sum_{t=1}^{T} \beta_t h_t(x_i) - y_i \right)^2 + \lambda \sum_{t=1}^{T} e_t |\beta_t| + \sum_{\alpha=1}^{d} c_\alpha \sqrt{\sum_{t=1}^{T} F_{\alpha t} \beta_t^2} + \rho |\beta|
\]
\[
\frac{1}{2} \sum_{i=1}^{n} \left( \sum_{t=1}^{T} \beta_t h_t(x_i) - y_i \right)^2 + \lambda \sum_{t=1}^{T} e_t |\beta_t| + \sum_{\alpha=1}^{d} c_{\alpha} \sqrt{\sum_{t=1}^{T} F_{\alpha t} \beta_t^2} + \rho |\beta|
\]

Non-differentiable

Introducing scaling factors to make it differentiable
(Chapelle and Keerthi, 2004)
\[
\frac{1}{2} \sum_{i=1}^{n} \left( \sum_{t=1}^{T} \beta_t h_t(x_i) - y_i \right)^2 + \lambda \sum_{t=1}^{T} e_t |\beta_t| + \sum_{\alpha=1}^{d} c_\alpha \sqrt{\sum_{t=1}^{T} F_{\alpha t} \beta_t^2} + \rho |\beta|
\]

1. jointly convex on both the weight vector and the scaling factors;

Introducing scaling factors to make it differentiable
(Chapelle and Keerthi, 2004)

Non-differentiable
\[
\frac{1}{2} \sum_{i=1}^{n} \left( \sum_{t=1}^{T} \beta_t h_t(x_i) - y_i \right)^2 + \lambda \sum_{t=1}^{T} e_t |\beta_t| + \sum_{\alpha=1}^{d} c_{\alpha} \sqrt{\sum_{t=1}^{T} F_{\alpha t} \beta_t^2} + \rho |\beta|
\]

Non-differentiable

Introducing scaling factors to make it differentiable
(Chapelle and Keerthi, 2004)

1. jointly convex on both the weight vector and the scaling factors;
2. quadratic problem (closed form solution) at each iteration.
\[
\frac{1}{2} \sum_{i=1}^{n} \left( \sum_{t=1}^{T} \beta_t h_t(x_i) - y_i \right)^2 + \lambda \left( \sum_{t=1}^{T} e_t |\beta_t| \right) + \sum_{\alpha=1}^{d} c_{\alpha} \sqrt{\sum_{t=1}^{T} F_{\alpha t} \beta_t^2} + \rho |\beta|
\]

1. jointly convex on both the weight vector and the scaling factors;
2. quadratic problem (closed form solution) at each iteration.

Fast training: 5,000 weak classifiers, 150,000 inputs, 13 seconds
Method

- Is it necessary to let every points go through the entire ensemble of trees?
  - Re-order the trees to allow “easy” inputs to be classified primarily on cheap features and fewer trees than “difficult” inputs
  - Highly imbalanced data  \(\longrightarrow\)  Rule out irrelevant documents earlier
  - Reduce the average computation time tremendously
Classifier Cascade

Minmin Chen

Classifier Cascade for Minimizing Feature Evaluation Cost

Washington University in St. Louis
Classifier Cascade

Stack multiple re-weighted classifiers into an ordered cascade
Classifier Cascade

Stack multiple re-weighted classifiers into an ordered cascade

Early exit

$$\sum_{t=1}^{T} \beta_t h_t(x_t) < \theta_t$$
Stack multiple re-weighted classifiers into an ordered cascade

Early exit

\[ \sum_{t=1}^{T} \beta_t^1 h_t(x_i) < \theta_1 \]

Early exit

\[ \sum_{t=1}^{T} \beta_t^2 h_t(x_i) < \theta_2 \]
Classifier Cascade

Stack multiple re-weighted classifiers into an ordered cascade

\[ \sum_{t=1}^{T} \beta_t^1 h_t(x_i) < \theta_1 \]

\[ \sum_{t=1}^{T} \beta_t^2 h_t(x_i) < \theta_2 \]

\[ \sum_{t=1}^{T} \beta_t^{K-1} h_t(x_i) < \theta_{K-1} \]
How to order the trees?

**Greedy?**

*(cheap features first)*
How to order the trees?

*Greedy? (cheap features first)*

Joint optimization on the ordering and the weights
How to order the trees? **Greedy?** (cheap features first)

Joint optimization on the ordering and the weights

**Loss**

\[
\frac{1}{2} \sum_{i=1}^{n} \sum_{t=1}^{T} \beta_t h_t(x_i) - y_i)^2 + \lambda \sum_{t=1}^{T} e_t |\beta_t| + \sum_{\alpha=1}^{d} c_\alpha \sum_{t=1}^{T} \beta^2_t + \rho |\beta|
\]

**Tree cost**

\[
\sum_{t=1}^{T} e_t |\beta_t|
\]

**Feature cost**

\[
\sum_{\alpha=1}^{d} c_\alpha \sum_{t=1}^{T} \beta^2_t
\]

**Reg.**

\[
\rho |\beta|
\]

**Loss**

\[
\frac{1}{2} \sum_{i=1}^{n} \sum_{k=1}^{K} q_i^k \sum_{t=1}^{T} \beta^k_t h_t(x_i) - y_i)^2 + \rho \sum_{k=1}^{K} |\beta^k|
\]

**Tree cost**

\[
\sum_{t=1}^{T} e_t \sum_{k=1}^{K} (\beta^k_t d_k)^2
\]

**Feature cost**

\[
\sum_{\alpha=1}^{d} c_\alpha \sum_{k=1}^{K} \sum_{t=1}^{T} (F_{\alpha t} \beta^k_t d_k)^2
\]

\[
\beta^1 \beta^2 \cdots \beta^K
\]

\[
\theta_1 \theta_2 \cdots \theta_K
\]

Minmin Chen

Classifier Cascade for Minimizing Feature Evaluation Cost

Washington University in St. Louis

Thursday, January 12, 12
Cronus: Cyclic Optimization

- **Initialization:** similar to the single stage formulation

- **Re-Optimization:**
Cronus: Cyclic Optimization

- **Initialization:** similar to the single stage formulation

- **Re-Optimization:**
Cronus: Cyclic Optimization

- **Initialization:** similar to the single stage formulation

  \[ \beta^1, \theta_1 \rightarrow 2 \beta^2, \theta_2 \rightarrow \text{Early exit} \]

- **Re-Optimization:**
Cronus: Cyclic Optimization

- **Initialization:** similar to the single stage formulation

  1. $\beta^1, \theta_1$
  2. $\beta^2, \theta_2$
  3. $\beta^k, \theta_k$

  Early exit

- **Re-Optimization:**
Cronus: Cyclic Optimization

- **Initialization:** similar to the single stage formulation

  1. $\beta^1, \theta_1$
  2. $\beta^2, \theta_2$
  ... (repeated for $k$)
  $\beta^k, \theta_k$

- **Early exit** at each stage $i$ (from $1$ to $k$)

- **Re-Optimization:**

  Minmin Chen
  Classifier Cascade for Minimizing Feature Evaluation Cost
  Washington University in St. Louis

Thursday, January 12, 12
Cronus: Cyclic Optimization

- **Initialization**: similar to the single stage formulation

- **Re-Optimization**:

Minmin Chen
Classifier Cascade for Minimizing Feature Evaluation Cost
Washington University in St. Louis
Experimental Results

- Dataset: Yahoo learning to rank;
  - 700 features (costs 1~200)
  - 5,000 weak classifiers (trees)
  - 141,397 training inputs
  - 146,769 testing inputs

- Precision@5: among the top 5 ranked documents, how many of them are relevant to the query;

- Comparisons:
  - GBRT
  - GBRT with early exit
  - Greedy ordering of features
  - Cronus

![Graph showing precision@5 and test-time cost for various classifier settings.](http://tinyurl.com/rtrank)

- Figure 2: The precision@5 and the test-time cost of various classifier settings.

- Figure 3: The fraction of test-inputs remaining per stage.

- Loss + $\lambda$ Cost + $\rho$ Reg.

Minmin Chen
Classifier Cascade for Minimizing Feature Evaluation Cost
Washington University in St. Louis

Thursday, January 12, 12
Experimental Results

- Dataset: Yahoo learning to rank;
  - 700 features (costs 1~200)
  - 5,000 weak classifiers (trees)
  - 141,397 training inputs
  - 146,769 testing inputs
- Precision@5: among the top 5 ranked documents, how many of them are relevant to the query;
- Comparisons:
  - GBRT
  - GBRT with early exit
  - Greedy ordering of features
  - Cronus

![Graph showing precision@5 vs. test cost](graph.png)

Minmin Chen  
Classifier Cascade for Minimizing Feature Evaluation Cost  
Washington University in St. Louis

Thursday, January 12, 12
Experimental Results

- Early stages use primarily cheap features;
- A few expensive features are used in the early stages to rule out more data points.

![Graph showing features used vs. stage](image1)

![Graph showing test-inputs remaining vs. stage](image2)

![Graph showing feature cost vs. stage](image3)
Conclusion

- Controlling the operational cost of machine learning algorithms is of great importance;
- Introduce a novel algorithm, Cronus, to build classifiers to trade-off prediction accuracy and runtime cost.
  - Optimize the order of feature extraction globally;
  - Provide an elegant and efficient method for initialization and parameter tuning.
Conclusion

- Controlling the operational cost of machine learning algorithms is of great importance;
- Introduce a novel algorithm, Cronus, to build classifiers to trade-off prediction accuracy and runtime cost.
  - Optimize the order of feature extraction globally;
  - Provide an elegant and efficient method for initialization and parameter tuning.

Thank you!
Questions?