Gradient Boosted Normalizing Flows

Robert Giaquinto and Arindam Banerjee

University of Minnesota, Twin Cities

December 7, 2020
Normalizing Flows

Figure: Normalizing flows construct flexible distributions via smooth, invertible mappings.

- Exact likelihoods, data generation.
- Recent trend: deeper, more complex transformations.
Gradient Boosted Normalizing Flows

- A *wider* alternative
- Iteratively add new flow components
- New component fit to residuals of previous fixed components
- Density estimation
- Variational inference

Related: Rosset and Segal 2002; Guo et al. 2016; Grover and Ermon 2018; Dinh et al. 2019; Cornish et al. 2020
Density Estimation with Multiplicative GBNF

- Weighted combination of fixed and new components.

\[
\mathcal{F} = -\frac{1}{N} \sum_{i=1}^{N} \left[ \left( \log(G_{K}^{(c-1)}(x_i)) + \rho_c \log(g_{K}^{(c)}(x_i)) \right) - \log \Gamma_{(c)} \right]
\]

- \( \Gamma_{(c)} \) partition function
- See paper for additive mixture formulation
Density Estimation with Multiplicative GBNF

- Weighted combination of fixed and new components.

\[
\mathcal{F} = -\frac{1}{N} \sum_{i=1}^{N} \left[ \left( \log(G^{(c-1)}_{K}(x_i)) + \rho_c \log(g^{(c)}_{K}(x_i)) \right) - \log \Gamma(c) \right]
\]

- $\Gamma(c)$ partition function
- See paper for additive mixture formulation

Training GBNF:

Component $c=1$: Fit with traditional objective — no boosting!
Component $c > 1$: Have fixed components $G^{(c-1)}_{K}$, and new component $g^{(c)}_{K}$

1. Train $g^{(c)}_{K}$ via Frank-Wolfe linear approximation
2. Optimize weight $\rho_c \in [0, 1]$
Advantages of Gradient Boosted Normalizing Flows

- Compliments existing flow models

---

1Note: Variational inference requires *analytically* invertible flows (see paper).
Advantages of Gradient Boosted Normalizing Flows

- Compliments existing flow models
- Resembles mixture
  - Easier! Just optimize $g_k^{(c)}$

---

$\begin{array}{|c|c|c|}
\hline
K & \text{Data} & \text{RealNVP} & \text{GBNF} \\
\hline
1 & & & \\
\hline
4 & & & \\
\hline
\end{array}$

$\begin{array}{|c|c|c|}
\hline
K & \text{Data} & \text{RealNVP} & \text{GBNF} \\
\hline
2 & & & \\
\hline
8 & & & \\
\hline
\end{array}$

Note: Variational inference requires *analytically* invertible flows (see paper).
Advantages of Gradient Boosted Normalizing Flows

- Compliments existing flow models\(^1\)
- Resembles mixture
  - Easier! Just optimize \(g^{(c)}_K\)
- Exchange flexibility for training cycles

\(\begin{array}{ccc}
K & \text{Data} & \text{RealNVP} & \text{GBNF} \\
1 & & & \\
4 & & & \\
2 & & & \\
8 & & & \\
\end{array}\)

\(^1\)Note: Variational inference requires \textit{analytically} invertible flows (see paper).
Advantages of Gradient Boosted Normalizing Flows

- Compliments existing flow models\(^1\)
- Resembles mixture
  - Easier! Just optimize \(g^{(c)}_K\)
- Exchange flexibility for training cycles
- Prediction/sampling in parallel

\(^1\)Note: Variational inference requires *analytically* invertible flows (see paper).
See Our Paper for More!

- Training components
- Analysis of objectives
- Unique challenges ("Decoder Shock")
- Experiments

Thank you!


Fit $g^{(c)}_K$ based on functional gradient descent, yields:

$$g^{(c)}_K = \arg \max_{g_K \in G_K} \mathbb{E}_{p^*} [\log g_K(x)] - \log \mathbb{E}_{G^{(c-1)}_K} [g_K(x)]$$
Fit $g_{K}^{(c)}$ based on *functional* gradient descent, yields:

$$g_{K}^{(c)} = \arg \max_{g_{K} \in G_{K}} \mathbb{E}_{p^*} [\log g_{K}(x)] - \log \mathbb{E}_{G_{K}^{(c-1)}} [g_{K}(x)]$$

Solution given by:

$$g_{K}^{(c)}(x) = \frac{p^*(x)}{G_{K}^{(c-1)}(x)}$$
Optimizing a New (Multiplicative) Boosting Component

- Fit $g_K^{(c)}$ based on functional gradient descent, yields:

$$g_K^{(c)} = \arg \max_{g_K \in G_K} \mathbb{E}_{p^*} [\log g_K(x)] - \log \mathbb{E}_{G_{K}^{(c-1)}} [g_K(x)]$$

- Solution given by:

$$g_K^{(c)}(x) = \frac{p^*(x)}{G_{K}^{(c-1)}(x)}$$

- How will new $g_K^{(c)}$ behave?
Fit $g_K^{(c)}$ based on functional gradient descent, yields:

$$g_K^{(c)} = \arg \max_{g_K \in G_K} \mathbb{E}_{p^*} [\log g_K(x)] - \log \mathbb{E}_{G_K^{(c-1)}} [g_K(x)]$$

Solution given by:

$$g_K^{(c)}(x) = \frac{p^*(x)}{G_K^{(c-1)}(x)}$$

How will new $g_K^{(c)}$ behave?

- Large $G_K^{(c-1)} \implies$ ignore, already covered
- Small $G_K^{(c-1)} \implies$ attractive!
Decoder Shock

Gradient boosting flows unlike decision trees

- Gradient boosting flows unlike decision trees
Decoder Shock

- Gradient boosting flows unlike decision trees
- Decoder shared by all components
Decoder Shock

- Gradient boosting flows unlike decision trees
- Decoder shared by all components
- What happens when we begin training a new component?
Decoder Shock

Figure: Loss on the test set decreases steadily as we add new components. The validation loss, however, jumps when a new component is introduced due to a sudden change in the distribution of samples passed to the decoder (aka “decoder shock”).
Reasons for Decoder Shock

New component = sudden shift in distribution of samples, why?

1. Samples coming from new component
2. Objective’s regularization $KL(q(z | x) || p(z))$ annealed from 0 to 1
   $\implies$ No regularization (temporally)
   $\implies$ Model is free to find very flexible posterior

Solution:
Reasons for Decoder Shock

New component = sudden shift in distribution of samples, why?

1. Samples coming from new component

2. Objective’s regularization $KL(q(z \mid x) \parallel p(z))$ annealed from 0 to 1
   - No regularization (temporally)
   - Model is free to find very flexible posterior

Solution:

- Blend in samples from fixed components
- Helps “remember” previous components
- $G_{1:K}^{(c-1)}$ still fixed