Improved Dropout for Shallow and Deep Learning

Zhe Li
Joint work with Prof. Gong and Prof. Yang

The University of Iowa

Wednesday 29th March, 2017

Outline

The success of deep learning

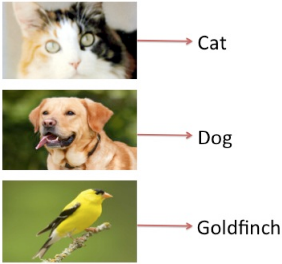
- Image Classification

The success of deep learning

- Image Classification

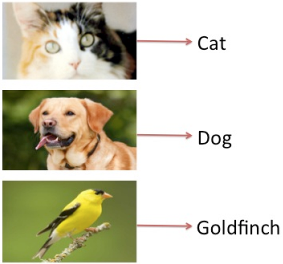
The success of deep learning

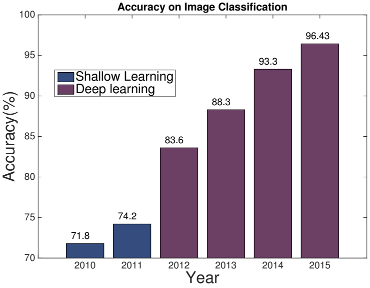
- Image Classification



The success of deep learning

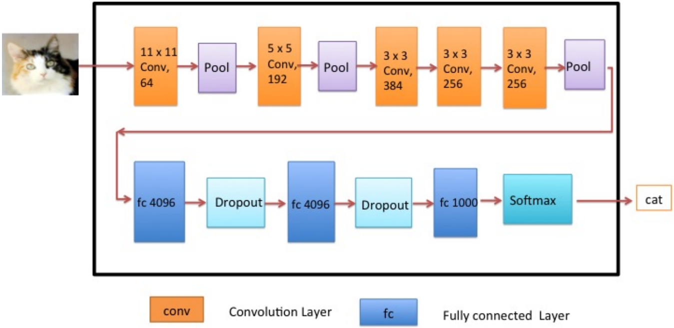
- Image Classification





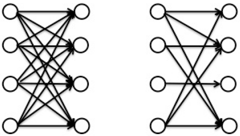
Deep Neural Network

The classical example: AlexNet [A Krizhevsky, et .al, 2012]



Dropout Layer

- Dropout Layer: Uniformly at randomly drop out features.

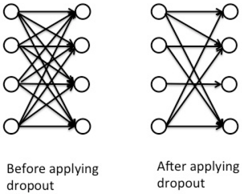


Before applying dropout

After applying dropout

Dropout Layer

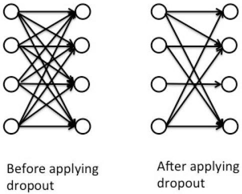
- Dropout Layer: Uniformly at randomly drop out features.



- Is uniformly dropout optimal?

Dropout Layer

- Dropout Layer: Uniformly at randomly drop out features.



- Is uniformly dropout optimal?
  - Answered the above question in this work.

Improved Dropout

Improved Dropout

- Dropping out the output of the neuron based on multinomial distribution computed from the training data.

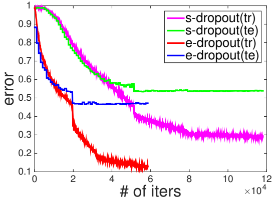


Figure: Evolutional dropout vs standard dropout on CIFAR100 datasets for deep learning

Problem Setup

- Let $(\mathbf{x}, y)$ denote a feature vector and a label, where $\mathbf{x} \in \mathbb{R}^d$ and $y \in \mathcal{Y}$.

Problem Setup

- Let $(\mathbf{x}, y)$ denote a feature vector and a label, where $\mathbf{x} \in \mathbb{R}^d$ and $y \in \mathcal{Y}$.
- Denote by $\mathcal{P}$ the joint distribution of $(\mathbf{x}, y)$ and by $\mathcal{D}$ the marginal distribution of $\mathbf{x}$.

Problem Setup

- Let $(\mathbf{x}, y)$ denote a feature vector and a label, where $\mathbf{x} \in \mathbb{R}^d$ and $y \in \mathcal{Y}$.
- Denote by $\mathcal{P}$ the joint distribution of $(\mathbf{x}, y)$ and by $\mathcal{D}$ the marginal distribution of $\mathbf{x}$.
- The goal is to learn a linear prediction function ($f(x) = \mathbf{w}^\top \mathbf{x}$) that minimizes the expected risk (considering loss function $\ell(\cdot, y)$):

$$\min_{\mathbf{w} \in \mathbb{R}^d} \mathcal{L}(\mathbf{w}) \triangleq \mathrm{E}_{\mathcal{P}}[\ell(\mathbf{w}^\top \mathbf{x}, y)] \tag{1}$$

Problem Setup

- Denote by $\epsilon \sim \mathcal{M}$ a dropout noise vector of dimension d.

Problem Setup

- Denote by $\epsilon \sim \mathcal{M}$ a dropout noise vector of dimension d.
- The corrupted feature vector is given by $\widehat{\mathbf{x}} = \mathbf{x} \circ \epsilon$, where the operator $\circ$ represents the element-wise multiplication.

Problem Setup

- Denote by $\epsilon \sim \mathcal{M}$ a dropout noise vector of dimension d.
- The corrupted feature vector is given by $\widehat{\mathbf{x}} = \mathbf{x} \circ \epsilon$, where the operator $\circ$ represents the element-wise multiplication.
- Denote by $\widehat{\mathcal{P}}$ the joint distribution of the new data $(\widehat{\mathbf{x}}, y)$ and by $\widehat{\mathcal{D}}$ the marginal distribution of $\widehat{\mathbf{x}}$.

Problem Setup

- Denote by $\epsilon \sim \mathcal{M}$ a dropout noise vector of dimension d.
- The corrupted feature vector is given by $\widehat{\mathbf{x}} = \mathbf{x} \circ \epsilon$, where the operator $\circ$ represents the element-wise multiplication.
- Denote by $\widehat{\mathcal{P}}$ the joint distribution of the new data $(\widehat{\mathbf{x}}, y)$ and by $\widehat{\mathcal{D}}$ the marginal distribution of $\widehat{\mathbf{x}}$.
- With the corrupted data, the risk minimization becomes

$$\min_{\mathbf{w} \in \mathbb{R}^d} \widehat{\mathcal{L}}(\mathbf{w}) \triangleq \mathrm{E}_{\widehat{\mathcal{P}}}[\ell(\mathbf{w}^\top(\mathbf{x} \circ \epsilon), y)] \tag{2}$$

Multinomial Dropout

Definition 1

A **multinomial dropout** is defined as $\widehat{\mathbf{x}} = \mathbf{x} \circ \epsilon$, where $\epsilon_i = \frac{m_i}{kp_i}, i \in [d]$ and $\{m_1, \ldots, m_d\}$ follow a multinomial distribution $Mult(p_1, \ldots, p_d; k)$ with $\sum_{i=1}^{d} p_i = 1$ and $p_i \geq 0$.

Multinomial Dropout

Definition 1

A **multinomial dropout** is defined as $\widehat{\mathbf{x}} = \mathbf{x} \circ \epsilon$, where $\epsilon_i = \frac{m_i}{kp_i}, i \in [d]$ and $\{m_1, \ldots, m_d\}$ follow a multinomial distribution $Mult(p_1, \ldots, p_d; k)$ with $\sum_{i=1}^{d} p_i = 1$ and $p_i \geq 0$.

- Ability of using non-uniformly sampling probabilities for different features.

Multinomial Dropout

Definition 1

A **multinomial dropout** is defined as $\widehat{\mathbf{x}} = \mathbf{x} \circ \epsilon$, where $\epsilon_i = \frac{m_i}{kp_i}, i \in [d]$ and $\{m_1, \ldots, m_d\}$ follow a multinomial distribution $Mult(p_1, \ldots, p_d; k)$ with $\sum_{i=1}^{d} p_i = 1$ and $p_i \geq 0$.

- Ability of using non-uniformly sampling probabilities for different features.
- Easy to control the level of dropout by varying the value of k.

Multinomial Dropout

- Dropout is a data-dependent regularizer.

Multinomial Dropout

- Dropout is a data-dependent regularizer.

Proposition 1

If $\ell(z, y) = \log(1 + \exp(-yz))$, then

$$\mathrm{E}_{\widehat{\mathcal{P}}}[\ell(\mathbf{w}^\top \widehat{\mathbf{x}}, y)] = \mathrm{E}_{\mathcal{P}}[\ell(\mathbf{w}^\top \mathbf{x}, y)] + R_{\mathcal{D},\mathcal{M}}(\mathbf{w})$$

where $\mathcal{M}$ denotes the distribution of ϵ and

$$R_{\mathcal{D},\mathcal{M}}(\mathbf{w}) = \mathrm{E}_{\mathcal{D},\mathcal{M}}\left[\log \frac{\exp(\mathbf{w}^\top \frac{\mathbf{x}\circ\epsilon}{2}) + \exp(-\mathbf{w}^\top \frac{\mathbf{x}\circ\epsilon}{2})}{\exp(\mathbf{w}^\top \mathbf{x}/2) + \exp(-\mathbf{w}^\top \mathbf{x}/2)}\right].$$

Learning with Multinomial Dropout

Learning with Multinomial Dropout

Learning with Multinomial Dropout

Learning with Multinomial Dropout

- Give the initial solution $\mathbf{w}_1$.

Learning with Multinomial Dropout

Learning with Multinomial Dropout

- Give the initial solution $\mathbf{w}_1$.
- Update the model at t^{th} iteration:

$$\mathbf{w}_{t+1} = \mathbf{w}_t - \eta_t \nabla \ell(\mathbf{w}_t^\top (\mathbf{x}_t \circ \boldsymbol{\epsilon}_t), y_t) \tag{3}$$

Learning with Multinomial Dropout

Learning with Multinomial Dropout

- Give the initial solution $\mathbf{w}_1$.
- Update the model at t^{th} iteration:

$$\mathbf{w}_{t+1} = \mathbf{w}_t - \eta_t \nabla \ell(\mathbf{w}_t^\top(\mathbf{x}_t \circ \boldsymbol{\epsilon}_t), y_t) \tag{3}$$

- Output the final solution:

$$\widehat{\mathbf{w}}_n = \frac{1}{n}\sum_{t=1}^{n} \mathbf{w}_t$$

Outline

Improved dropout for Shallow Learning

Theorem 1:

Let $\mathcal{L}(\mathbf{w})$ be the expected risk of $\mathbf{w}$ defined in (1). Assume $\mathrm{E}_{\widehat{\mathcal{D}}}[\|\mathbf{x} \circ \boldsymbol{\epsilon}\|_2^2] \leq B^2$ and $\ell(z, y)$ is convex and G-Lipschitz continuous. For any $\|\mathbf{w}_*\|_2 \leq r$, by appropriately choosing η, we can have

$$\mathrm{E}[\mathcal{L}(\widehat{\mathbf{w}}_n) + R_{\mathcal{D},\mathcal{M}}(\widehat{\mathbf{w}}_n)] \leq \mathcal{L}(\mathbf{w}_*) + R_{\mathcal{D},\mathcal{M}}(\mathbf{w}_*) + \frac{GBr}{\sqrt{n}}$$

How to prove the above theorem?

Improved dropout for Shallow Learning

Theorem 1:

Let $\mathcal{L}(\mathbf{w})$ be the expected risk of $\mathbf{w}$ defined in (1). Assume $\mathrm{E}_{\widehat{\mathcal{D}}}[\|\mathbf{x} \circ \boldsymbol{\epsilon}\|_2^2] \leq B^2$ and $\ell(z, y)$ is convex and G-Lipschitz continuous. For any $\|\mathbf{w}_*\|_2 \leq r$, by appropriately choosing η, we can have

$$\mathrm{E}[\mathcal{L}(\widehat{\mathbf{w}}_n) + R_{\mathcal{D},\mathcal{M}}(\widehat{\mathbf{w}}_n)] \leq \mathcal{L}(\mathbf{w}_*) + R_{\mathcal{D},\mathcal{M}}(\mathbf{w}_*) + \frac{GBr}{\sqrt{n}}$$

How to prove the above theorem?

- Standard SGD analysis.

Improved dropout for Shallow Learning

Theorem 1:

Let $\mathcal{L}(\mathbf{w})$ be the expected risk of $\mathbf{w}$ defined in (1). Assume $\mathrm{E}_{\widehat{\mathcal{D}}}[\|\mathbf{x} \circ \epsilon\|_2^2] \leq B^2$ and $\ell(z, y)$ is convex and G-Lipschitz continuous. For any $\|\mathbf{w}_*\|_2 \leq r$, by appropriately choosing η, we can have

$$\mathrm{E}[\mathcal{L}(\widehat{\mathbf{w}}_n) + R_{\mathcal{D},\mathcal{M}}(\widehat{\mathbf{w}}_n)] \leq \mathcal{L}(\mathbf{w}_*) + R_{\mathcal{D},\mathcal{M}}(\mathbf{w}_*) + \frac{GBr}{\sqrt{n}}$$

How to prove the above theorem?

- Standard SGD analysis.
- Dropout is a data-dependent regularizer.

Improved dropout for Shallow Learning

- Minimizing the term $\mathrm{E}_{\widehat{\mathcal{D}}}[\|\mathbf{x} \circ \boldsymbol{\epsilon}\|_2^2]$ and the relaxed upper bound of term $R_{\mathcal{D},\mathcal{M}}(\mathbf{w}_*)$ yields the optimal sampling probabilities:

$$p_i^* = \frac{\sqrt{\mathrm{E}_{\mathcal{D}}[x_i^2]}}{\sum_{j=1}^{d} \sqrt{\mathrm{E}_{\mathcal{D}}[x_j^2]}}, i = 1, \ldots, d \quad (4)$$

Improved dropout for Shallow Learning

- Minimizing the term $\mathrm{E}_{\widehat{\mathcal{D}}}[\|\mathbf{x} \circ \boldsymbol{\epsilon}\|_2^2]$ and the relaxed upper bound of term $R_{\mathcal{D},\mathcal{M}}(\mathbf{w}_*)$ yields the optimal sampling probabilities:

$$p_i^* = \frac{\sqrt{\mathrm{E}_{\mathcal{D}}[x_i^2]}}{\sum_{j=1}^{d} \sqrt{\mathrm{E}_{\mathcal{D}}[x_j^2]}}, i = 1, \ldots, d \qquad (4)$$

- Can we compute the above probability for dropout?

Improved dropout for Shallow Learning

- Minimizing the term $\mathrm{E}_{\widehat{\mathcal{D}}}[\|\mathbf{x} \circ \boldsymbol{\epsilon}\|_2^2]$ and the relaxed upper bound of term $R_{\mathcal{D},\mathcal{M}}(\mathbf{w}_*)$ yields the optimal sampling probabilities:

$$p_i^* = \frac{\sqrt{\mathrm{E}_{\mathcal{D}}[x_i^2]}}{\sum_{j=1}^d \sqrt{\mathrm{E}_{\mathcal{D}}[x_j^2]}}, i = 1, \ldots, d \tag{4}$$

- Can we compute the above probability for dropout?
  - ✗

Improved dropout for Shallow Learning

- Practically, we use the empirical second-order statistics to compute the probabilities:

$$p_i = \frac{\sqrt{\frac{1}{n}\sum_{j=1}^{n}[[\mathbf{x}_j]_i^2]}}{\sum_{i'=1}^{d}\sqrt{\frac{1}{n}\sum_{j=1}^{n}[[\mathbf{x}_j]_{i'}^2]}}, i = 1, \ldots, d \tag{5}$$

Outline

Improved dropout for deep learning

- Could we directly use the above idea to Deep Learning?

Improved dropout for deep learning

- Could we directly use the above idea to Deep Learning?
  - ✗

Improved dropout for deep learning

- Could we directly use the above idea to Deep Learning?
  - ✗
- Why not?

Improved dropout for deep learning

- Could we directly use the above idea to Deep Learning?
  - ✗
- Why not?
  - Too expensive to compute dropout probablity from all examples.

Improved dropout for deep learning

- Could we directly use the above idea to Deep Learning?
  - ✗
- Why not?
  - Too expensive to compute dropout probablity from all examples.
- How to address this issue?

Improved dropout for deep learning

- Could we directly use the above idea to Deep Learning?
  - ✗
- Why not?
  - Too expensive to compute dropout probablity from all examples.
- How to address this issue?
  - Use a mini-batch of examples to calculate the dropout probablity.

Improved dropout for deep learning

- Let $X^l = (\mathbf{x}_1^l, \ldots, \mathbf{x}_m^l)$ denote the outputs of the l^{th} layer for a mini-batch of m examples, calculate the probabilities for dropout by

$$p_i^l = \frac{\sqrt{\frac{1}{m}\sum_{j=1}^{m}[[\mathbf{x}_j^l]_i^2]}}{\sum_{i'=1}^{d}\sqrt{\frac{1}{m}\sum_{j=1}^{m}[[\mathbf{x}_j^l]_{i'}^2]}}, i = 1, \ldots, d \tag{6}$$

Improved dropout for deep learning

Evolutional Dropout for Deep Learning

Input: a batch of outputs of a layer: $X^l = (\mathbf{x}_1^l, \ldots, \mathbf{x}_m^l)$ and dropout level parameter $k \in [0, d]$

Output: $\widehat{X}^l = X^l \circ \Sigma^l$

Compute sampling probabilities by (6)

For $j = 1, \ldots, m$

Sample $\mathbf{m}_j^l \sim Mult(p_1^l, \ldots, p_d^l; k)$

Construct $\epsilon_j^l = \frac{\mathbf{m}_j^l}{k\mathbf{p}^l} \in \mathbb{R}^d$, where $\mathbf{p}^l = (p_1^l, \ldots, p_d^l)^\top$

Let $\Sigma^l = (\epsilon_1^l, \ldots, \epsilon_m^l)$ and compute $\widehat{X}^l = X^l \circ \Sigma^l$

Figure: Evolutional Dropout applied to a layer over a mini-batch

Outline

Experimental Results for Shallow Learning

Training/test error between standard and improved dropout

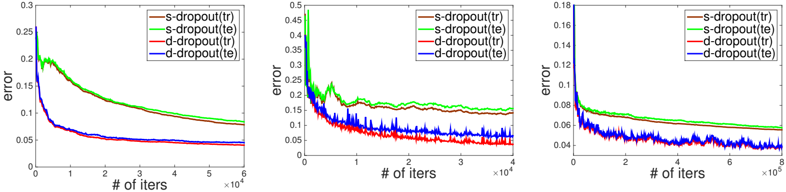


Figure: data-dependent dropout vs. standard dropout on three datasets (real-sim, news20 and RCV1) for logistic regression

Experimental Results for Deep Learning

- Implemented in CudaConvNet Library.

Experimental Results for Deep Learning

- Implemented in CudaConvNet Library.
- Using four benchmark datasets: MNIST, SVHN, CIFAR10, CIFAR100.

Experimental Results for Deep Learning

- Implemented in CudaConvNet Library.
- Using four benchmark datasets: MNIST, SVHN, CIFAR10, CIFAR100.
- Different neural network stuctures from the existing literatures.

Experimental Results for Deep Learning

- Implemented in CudaConvNet Library.
- Using four benchmark datasets: MNIST, SVHN, CIFAR10, CIFAR100.
- Different neural network stuctures from the existing literatures.
- Training strategy.

Experimental Results for Deep Learning

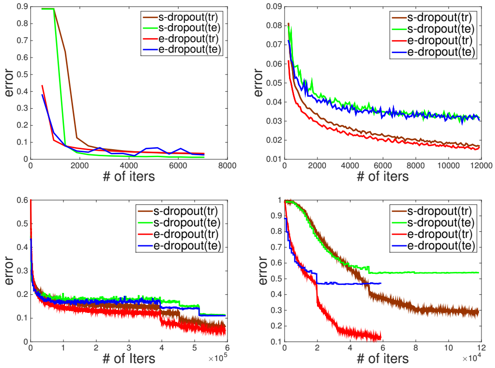


Figure: Evolutional dropout vs. standard dropout on four benchmark datasets (MNIST, SVHN, CIFAR-10 and CIFAR-100) for deep learning

Experimental Results for Deep Learning

Compared to Batch Normalization

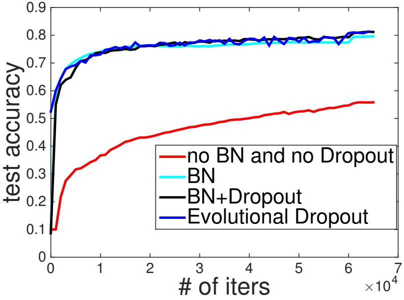


Figure: Evolutional dropout vs BN on CIFAR-10.

Outline

Conclusion

- Proposed a multinomial dropout for shallow learning.

Conclusion

- Proposed a multinomial dropout for shallow learning.
- Demonstrated that this proposed distribution-dependent dropout leads to a faster convergence and a smaller generalization error through the risk bound analysis.

Conclusion

- Proposed a multinomial dropout for shallow learning.
- Demonstrated that this proposed distribution-dependent dropout leads to a faster convergence and a smaller generalization error through the risk bound analysis.
- Proposed an efficient evolutional dropout for deep learning.

Conclusion

- Proposed a multinomial dropout for shallow learning.
- Demonstrated that this proposed distribution-dependent dropout leads to a faster convergence and a smaller generalization error through the risk bound analysis.
- Proposed an efficient evolutional dropout for deep learning.
- Justified the proposed dropouts for both shallow and deep learning empirically.

Question?