Semi-supervised Dropout Training

Baylearn 2013
Stefan Wager, Sida Wang, Percy Liang
The basics of dropout training

- Introduced by Hinton et al. in “Improving neural networks by preventing co-adaptation of feature detectors”
- For each example, randomly select features
  - zero them
  - compute the gradient, make an update
  - repeat
Empirically successful

• Dropout is important in some recent successes
  • won the ImageNet challenge [Krizhevsky et al., 2012]
  • won the Merck challenge [Dahl et al., 2012]

• Improved performance on standard datasets
  • images: MNIST, CIFAR, ImageNet, etc.
  • document classification: Reuters, IMDB, Rotten Tomatoes, etc.
  • speech: TIMIT, GlobalPhone, etc.
Lots of related works already

Variants

- DropConnect [Wan et al., 2013]
- Maxout networks [Goodfellow et al., 2013]

Analytical integration

- Fast Dropout [Wang and Manning, 2013]
- Marginalized Corrupted Features [van der Maaten et al., 2013]

Many other works report empirical gains
Theoretical understanding?

• Dropout as adaptive regularization
  • feature noising -> interpretable penalty term

\[
\text{Loss( Dropout(data) )} = \text{Loss(data)} + \text{Regularizer(data)}
\]

• Semi-supervised learning
  • feature dependent, label independent regularizer:

\[
\text{Regularizer(Unlabeled data)}
\]
Dropout for Log-linear Models

- Log likelihood (e.g., softmax classification):

\[ \log p(y|x; \theta) = x^T \theta_y - A(x^T \theta) \]

\[ \theta = [\theta_1, \theta_2, \ldots, \theta_K] \]
Dropout for Log-linear Models

- Log likelihood (e.g., softmax classification):
  \[
  \log p(y|x; \theta) = x^T \theta_y - A(x^T \theta)
  \]
  \[
  \theta = [\theta_1, \theta_2, \ldots, \theta_K]
  \]

- Dropout: \[
  \tilde{x}_j = \begin{cases} 
  2x_j & \text{with } p=0.5 \\
  0 & \text{otherwise}
  \end{cases}
  \]
  \[
  \mathbb{E}[\tilde{x}] = x
  \]

- Dropout objective:
  \[
  \mathbb{E}[\log p(y|\tilde{x}; \theta)] = \mathbb{E}[\tilde{x}^T \theta_y] - \mathbb{E}[A(\tilde{x}^T \theta)]
  \]
  \[
  - \text{Loss(Dropout(data))}
  \]
  \[
  \text{Loss(data)} + \text{Regularizer(data)}
  \]
Dropout for Log-linear Models

- We can rewrite the dropout log-likelihood

\[
\mathbb{E}[\log p(y|\tilde{x}; \theta)] - \mathbb{E}[\tilde{x}^T \theta_y] - \mathbb{E}[A(\tilde{x}^T \theta)] = -\text{Loss}(\text{Dropout(data)})
\]

\[
\log p(y|x; \theta) = x^T \theta_y - A(x^T \theta)
\]

\[
\mathbb{E}[\log p(y|x; \theta)] = \log p(y|x; \theta) - (\mathbb{E}[A(\tilde{x}^T \theta)] - A(x^T \theta))
\]

- Dropout reduces to a regularizer

\[
R(\theta, x) = \mathbb{E}[A(\tilde{x}^T \theta)] - A(x^T \theta)
\]
Second-order delta method

Take the Taylor expansion

\[ A(s) \approx A(s_0) + (s - s_0)^T A'(s_0) + (s - s_0)^T \frac{A''(s_0)}{2} (s - s_0) \]
Second-order delta method

Take the Taylor expansion

\[ A(s) \approx A(s_0) + (s - s_0)^T A'(s_0) + (s - s_0)^T \frac{A''(s_0)}{2} (s - s_0) \]

Substitute \( s = \tilde{s} \overset{\text{def}}{=} \theta^T \tilde{x} \), \( s_0 = \mathbb{E}[\tilde{s}] \)

Take expectations to get the quadratic approximation:

\[ R^q(\theta, x) = \frac{1}{2} \mathbb{E}[(\tilde{s} - s)^T \nabla^2 A(s)(\tilde{s} - s)] \]

\[ = \frac{1}{2} \text{tr}(\nabla^2 A(s) \text{Cov}(\tilde{s})) \]
Example: logistic regression

- The quadratic approximation

\[ R^q(\theta, x) = \frac{1}{2} A''(x^T \theta) \text{Var}[\tilde{x}^T \theta] \]
Example: logistic regression

- The quadratic approximation
  \[ R^q(\theta, x) = \frac{1}{2} A''(x^T \theta) \text{Var}[\tilde{x}^T \theta] \]

- \( A''(x^T \theta) = p(1 - p) \) represents uncertainty:
  \[ p = p(y|x; \theta) = (1 + \exp(-yx^T \theta))^{-1} \]
Example: logistic regression

- The quadratic approximation

\[ R^q(\theta, x) = \frac{1}{2} A''(x^T \theta) \text{Var}[\tilde{x}^T \theta] \]

- \(A''(x^T \theta) = p(1 - p)\) represents uncertainty:

\[ p = p(y|x; \theta) = (1 + \exp(-yx^T \theta))^{-1} \]

- \(\text{Var}[\tilde{x}^T \theta] = \sum_j \theta_j^2 x_j^2\) is L₂-regularization after normalizing the data
The regularizers

- Dropout on Linear Regression
  \[ R^q(\theta) = \frac{1}{2} \sum_j \theta_j^2 \sum_i x_j^{(i)2} \]

- Dropout on Logistic Regression
  \[ R^q(\theta) = \frac{1}{2} \sum_j \theta_j^2 \sum_i p_i(1 - p_i)x_j^{(i)2} \]

- Multiclass, CRFs [Wang et al., 2013]
Dropout intuition

\[ R_q(\theta) = \frac{1}{2} \sum_j \theta_j^2 \sum_i p_i (1 - p_i) x_i^{(i)^2} \]

- Regularizes “rare” features less, like AdaGrad: there is actually a more precise connection [Wager et al., 2013]
- Big weights are okay if they contribute only to confident predictions
- Normalizing by the diagonal Fisher information
Semi-supervised Learning

- These regularizers are label-independent
  - but can be data adaptive in interesting ways
- labeled dataset \( \mathcal{D} = \{x_1, x_2, \ldots, x_n\} \)
- unlabeled data \( \mathcal{D}_{\text{unlabeled}} = \{u_1, u_2, \ldots, u_n\} \)
- We can better estimate the regularizer

\[
R_\ast(\theta, \mathcal{D}, \mathcal{D}_{\text{unlabeled}}) \defeq \frac{n}{n + \alpha m} \left( \sum_{i=1}^{n} R(\theta, x_i) + \alpha \sum_{i=1}^{m} R(\theta, u_i) \right).
\]

for some tunable \( \alpha \).
Semi-supervised intuition

\[ R^q(\theta) = \frac{1}{2} \sum_j \theta_j^2 \sum_i p_i (1 - p_i) x_j^{(i)2} \]

- Like other semi-supervised methods:
  - transductive SVMs [Joachims, 1999]
  - entropy regularization [Grandvalet and Bengio, 2005]
  - EM: guess a label [Nigam et al., 2000]
  - want to make confident predictions on the unlabeled data
- Get a better estimate of the Fisher information
IMDB dataset [Maas et al., 2011]

- 25k examples of positive reviews
- 25k examples of negative reviews
- Half for training and half for testing
- 50k unlabeled reviews also containing neutral reviews
- 300k sparse unigram features
- ~5 million sparse bigram features
Experiments: semi-supervised

- Add more unlabeled data (10k labeled) improves performance
Experiments: semi-supervised

- Add more labeled data (40k unlabeled) improves performance
## Quantitative results on IMDB

<table>
<thead>
<tr>
<th>Method \ Settings</th>
<th>Supervised</th>
<th>Semi-sup.</th>
</tr>
</thead>
<tbody>
<tr>
<td>MNB - unigrams with SFE [Su et al., 2011]</td>
<td>83.62</td>
<td>84.13</td>
</tr>
<tr>
<td>Vectors for sentiment analysis [Maas et al., 2011]</td>
<td>88.33</td>
<td>88.89</td>
</tr>
<tr>
<td>This work: dropout + unigrams</td>
<td>87.78</td>
<td>89.52</td>
</tr>
<tr>
<td>This work: dropout + bigrams</td>
<td>91.31</td>
<td>91.98</td>
</tr>
</tbody>
</table>
Experiments: other datasets

<table>
<thead>
<tr>
<th>Dataset \ Settings</th>
<th>L₂</th>
<th>Drop</th>
<th>+Unlbl</th>
</tr>
</thead>
<tbody>
<tr>
<td>Subjectivity [Peng and Lee, 2004]</td>
<td>88.96</td>
<td>90.85</td>
<td>91.48</td>
</tr>
<tr>
<td>Rotten Tomatoes [Peng and Lee, 2005]</td>
<td>73.49</td>
<td>75.18</td>
<td>76.56</td>
</tr>
<tr>
<td>20-newsgroups</td>
<td>82.19</td>
<td>83.37</td>
<td>84.71</td>
</tr>
<tr>
<td>CoNLL-2003</td>
<td>80.12</td>
<td>80.90</td>
<td>81.66</td>
</tr>
</tbody>
</table>
• Our arXiv paper [Wager et al., 2013] has more details, including the relation to AdaGrad

• Our EMNLP paper [Wang et al., 2013] extends this framework to structured prediction

• Our ICML paper [Wang and Manning, 2013] applies a related technique to neural networks and provides some negative examples
CRF sequence tagging

- CoNLL 2003 Named Entity Recognition

<table>
<thead>
<tr>
<th>Dataset \ Settings</th>
<th>None</th>
<th>L₂</th>
<th>Drop</th>
</tr>
</thead>
<tbody>
<tr>
<td>CoNLL 2003 Dev</td>
<td>89.40</td>
<td>90.73</td>
<td>91.86</td>
</tr>
<tr>
<td>CoNLL 2003 Test</td>
<td>84.67</td>
<td>85.82</td>
<td>87.42</td>
</tr>
</tbody>
</table>
• Our arXiv paper [Wager et al., 2013] has more details, including the relation to AdaGrad

• Our EMNLP paper [Wang et al., 2013] extends this framework to structured prediction

• Our ICML paper [Wang and Manning, 2013] applies a related technique to neural networks and provides some negative examples

• Thanks! Any questions?
Dropout vs. $L_2$

- Can be much better than all settings of $L_2$
- Part of the gain comes from normalization

![Graph showing the effect of $L_2$ regularization strength on test set performance](graph.png)

Table 2: Classification performance and transductive learning results on some standard datasets.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>$K$</th>
<th>$L_2$ only</th>
<th>$L_2$+Quadratic dropout</th>
<th>$L_2$+Gaussian dropout</th>
</tr>
</thead>
<tbody>
<tr>
<td>CoNLL</td>
<td>5</td>
<td>80.12</td>
<td>80.90</td>
<td>81.66</td>
</tr>
<tr>
<td>20news</td>
<td>20</td>
<td>82.19</td>
<td>83.37</td>
<td>84.71</td>
</tr>
<tr>
<td>RCV1</td>
<td>4</td>
<td>95.90</td>
<td>96.03</td>
<td>96.11</td>
</tr>
<tr>
<td>R21578</td>
<td>65</td>
<td>92.24</td>
<td>92.24</td>
<td>92.58</td>
</tr>
<tr>
<td>TDT2</td>
<td>30</td>
<td>97.91</td>
<td>98.00</td>
<td>98.12</td>
</tr>
</tbody>
</table>

Table 3: Semisupervised learning results on some standard datasets.

A third (33%) of the full dataset was used for training, a third for testing, and the rest as unlabeled.

Over a broad range of $\lambda$ values, we find that dropout plus $L_2$ regularization performs far better than using just $L_2$ regularization for any value of $\lambda$. We see that Gaussian dropout appears to perform slightly better than the quadratic approximation discussed in this paper. However, our quadratic approximation extends easily to the multiclass case and to structured prediction in general, while Gaussian dropout does not. Thus, it appears that our approximation presents a reasonable trade-off between...
Example: linear least squares

- The loss function is $f(\theta \cdot x) = 1/2(\theta \cdot x - y)^2$
- Let $X = \theta \cdot \tilde{x}$ where $\tilde{x}_j = 2z_j x_j$, $z_j = \text{Bernoulli}(0.5)$

$$
\mathbb{E}[f(X)] = f(\mathbb{E}[X]) + \frac{f''(\mathbb{E}[X])}{2} \text{Var}[X]
= 1/2(\theta \cdot x - y)^2 + 1/2 \sum_j x_j^2 \theta_j^2
$$

- The total regularizer is

$$
R^q(\theta) = \frac{1}{2} \sum_j \theta_j^2 \sum_i x_j^{(i)2}
$$

- This is just L2 applied after data normalization
## Quantitative results on IMDB

<table>
<thead>
<tr>
<th>Method \ Settings</th>
<th>Supervised</th>
<th>Semi-sup.</th>
</tr>
</thead>
<tbody>
<tr>
<td>MNB - unigrams with SFE [Su et al., 2011]</td>
<td>83.62</td>
<td>84.13</td>
</tr>
<tr>
<td>MNB – bigrams</td>
<td>86.63</td>
<td>86.98</td>
</tr>
<tr>
<td>Vectors for sentiment analysis [Maas et al., 2011]</td>
<td>88.33</td>
<td>88.89</td>
</tr>
<tr>
<td>NBSVM – bigrams [Wang and Manning, 2012]</td>
<td>91.22</td>
<td>-</td>
</tr>
<tr>
<td>This work: dropout + unigrams</td>
<td>87.78</td>
<td>89.52</td>
</tr>
<tr>
<td>This work: dropout + bigrams</td>
<td>91.31</td>
<td><strong>91.98</strong></td>
</tr>
</tbody>
</table>