Maxout Networks

Hien Quoc Dang
Outline

- Introduction
- Maxout Networks
  - Description
  - A Universal Approximator & Proof
- Experiments with Maxout
- Why does Maxout work?
- Conclusion
Introduction

- Generalization
  - Adding noise
  - Training multiple models and use the average model of those

- Dropout
  - Drop a hidden unit with probability of 0.5
  - Maximal $2^h$ models ($2^{64} = 1.8 \times 10^{19}$)
  - Approximation to geometric mean
  - Fast averaging technique (divide weights by 2)

- Maxout *(Goodfellow et al)*
  - Facilitate dropout’s optimization
  - Improve accuracy of dropout’s fast approximate model averaging technique
Idea of Maxout

- Traditional activation functions

Threshold function

Sigmoid function
Idea of Maxout

- Do not use a fixed activation function
- But learn the activation function
- Piecewise Linear Function
  - Can approximate any continuous function (*Stone-Weierstrass*)
  - Linear almost everywhere, except \( k-1 \) points
Idea of Maxout

- Maxout unit
  - k linear models
  
  Output is the maximal value from k models from the given input x

- Formal:
  \[
  h_i(x) = \max_{j \in \{1, k\}} z_{ij}
  \]

Where
\[
  z_{ij} = x^T W_{ij} + b_{ij}
\]

- \( W \in \mathbb{R}^{d \times m \times k} \) and \( b \in \mathbb{R}^{m \times k} \)

- \( m \): number of hidden units
- \( d \): size of input vector (x)
- \( k \): number of linear models
Idea of Maxout

Rectifier

Absolute value

Quadratic

$h_i(x)$

$x$

$k=2$

$k=2$

$k=5$
Maxout: universal approximator

- Maxout networks with two hidden units:

```
\begin{tikzpicture}
  \node[latent] (v) at (0,0) {$v$};
  \node[latent] (z1) at (1,-1) {$z_1$};
  \node[latent] (z2) at (2,-1) {$z_2$};
  \node[latent] (h1) at (1,-2) {$h_1$};
  \node[latent] (h2) at (2,-2) {$h_2$};
  \node[latent] (g) at (1.5,-3) {$g$};
  \node[obs] (w1) at (0,-3) {$W_1 = 1$};
  \node[obs] (w2) at (1.5,-3) {$W_2 = -1$};

  \edge {v} {z1, z2, h1, h2, g, w1, w2} ;
\end{tikzpicture}
```
Maxout: universal approximator

- Universal approximator theorem:
  
  Any continuous function $f$ can be approximated arbitrarily well on a compact domain $C \subset \mathbb{R}^n$ by a maxout network with two maxout hidden units.

- Proof
  
  (Wang, 2004) Any continuous function can be expressed as a difference of 2 convex functions
  
  $$g(x) = h_1(x) - h_2(x)$$  \hspace{1cm} (1)

  (Stone-Weierstrass) Any continuous function can be approximated by a piecewise linear function
  
  $$|f(x) - g(x)| < \varepsilon$$  \hspace{1cm} (2)
## Experiment on benchmark datasets

<table>
<thead>
<tr>
<th>Name</th>
<th>Classes</th>
<th>Training</th>
<th>Test</th>
<th>Image</th>
<th>Color</th>
</tr>
</thead>
<tbody>
<tr>
<td>MNIST</td>
<td>10</td>
<td>60 000</td>
<td>10 000</td>
<td>28x28</td>
<td>Grayscale</td>
</tr>
<tr>
<td>CIFAR-10</td>
<td>10</td>
<td>50 000</td>
<td>10 000</td>
<td>32x32</td>
<td>Color</td>
</tr>
<tr>
<td>CIFAR-100</td>
<td>100</td>
<td>50 000</td>
<td>10 000</td>
<td>32x32</td>
<td>Color</td>
</tr>
<tr>
<td>SVHN</td>
<td>10</td>
<td>73 257</td>
<td>26 032</td>
<td>32x32</td>
<td>Color</td>
</tr>
</tbody>
</table>

- SVHN dataset also consists of 521,131 additional samples
**MNIST**

- *Permutation invariant MNIST*

- Maxout multilayer perceptron (MLP):
  - Two *maxout layers* followed by a *softmax layer*
  - Dropout
  - Training/Validation/Test: 50,000/10,000/10,000 samples

- Error rate: 0.94%

- This is the best result without pre-training
Without permutation invariant restriction

Best model consists of:
- 3 convolutional maxout hidden layers with spatial max pooling
- Followed by a softmax layer

Error rate is 0.45%

There are better results by augmenting standard dataset
Preprocessing
- Global contrast normalization
- ZCA whitening

Best model consists of
- 3 convolutional maxout layers
- A fully connected maxout layer
- A fully connected softmax layer

Error rate
- Without data augmentation 11.68 %
- With data augmentation 9.35 %
CIFAR-100

- Use the same hyperparameters as in CIFAR-10

- Error rates
  - Without retraining using entire training set: 41.48 %
  - With retraining: 38.57 %
SVHN

- Local contrast normalization preprocessing
- 3 convolutional maxout hidden layers
- 1 maxout layer
- Followed by a softmax layer

- Error rate is 2.47%

Local contrast normalization
(Zeiler&Fergus 2013)
Comparison to rectifiers

Comparison of large rectifier networks to maxout

- Maxout
- Rectifier, no channel pooling
- Rectifier + channel pooling
- Large rectifier, no channel pooling

Validation set error for best experiment vs. training epochs

0.160
0.155
0.150
0.145
0.140
0.135
0.130
0.125
0
100
200
300
400
500
600
700
800
Training epochs
What does Maxout work?

- Enhance accuracy of dropout model averaging technique
- Maxout using with dropout improves optimization
- Maxout improves bagging training style on deeper layer
Model Averaging

- Dropout performs model averaging
- Comparing of geometric mean of sample’s subsets and full model of dropout with half of the weight W
- Maxout improves accuracy of dropout
Model Averaging

- Kullback-Leibler divergence between geometric mean of sample’s subset and dropout averaged model

- The approximation is more accurate for maxout units
Optimization

- Maxout works better than max pooled rectified linear units
  - Small model on large dataset
    - 2 convolutional layers
    - Training with big SVHN dataset (600,000 samples)
  - Error rate
    - Maxout error : 5.1%
    - Rectifier error : 7.3%
Optimization

- Maxout works better than max pooled rectified linear units
- Comparison on network depth

![Graph showing MNIST classification error versus network depth]
Saturation

- **Maxout:**
  - Rate of sign switches is equals
  - $>99.99\%$ filters used
- **Rectifier:**
  - "death rate" is bigger than "birth rate"
  - $40\%$ filters are unused
Conclusion

- A new activation function which is suited with dropout
- Proof of a universal approximator with 2 maxout hidden units
- Maxout model benefits more from dropout than other activation functions
- Set new state of the art on 4 benchmark datasets
References

- Goodfellow et al., Maxout Networks, Proceedings of International Conference on Machine Learning (ICML), 2013