Batch Normalization

Seungmok Lee

2020.02.25

Online Lecture Recommendation

- Famous, free lecture by Andrew Ng
  - https://www.edwith.org/deeplearningai1
- From introduction of deep learning, to CNN.
- Easy, detailed and well-constructed lecture. Credible lecturer.
- Highly recommendable lecture!

Batch Normalization

- Idea from Google, in 2015.
- Normalize input before each layer, so that they fit to activation function.
  - Actually, I'm not understanding it well.

Input: Values of x over a mini-batch: $\mathcal{B} = \{x_{1...m}\}$;
Parameters to be learned: γ, β
Output: $\{y_i = \mathrm{BN}_{\gamma,\beta}(x_i)\}$

$$\mu_{\mathcal{B}} \leftarrow \frac{1}{m}\sum_{i=1}^{m} x_i \qquad \text{// mini-batch mean}$$

$$\sigma_{\mathcal{B}}^2 \leftarrow \frac{1}{m}\sum_{i=1}^{m}(x_i - \mu_{\mathcal{B}})^2 \qquad \text{// mini-batch variance}$$

$$\widehat{x}_i \leftarrow \frac{x_i - \mu_{\mathcal{B}}}{\sqrt{\sigma_{\mathcal{B}}^2 + \epsilon}} \qquad \text{// normalize}$$

$$y_i \leftarrow \gamma \widehat{x}_i + \beta \equiv \mathrm{BN}_{\gamma,\beta}(x_i) \qquad \text{// scale and shift}$$

Algorithm 1: Batch Normalizing Transform, applied to activation x over a mini-batch.

https://arxiv.org/abs/1502.03167

Batch Normalization

- Now, it is considered as a standard method to speed up the training, and to regularize over-fitting.
- Many people say dropout is not necessary if batch normalization was applied.

Batch Normalization in Keras

- It is really easy to implement in Keras.
  - just ‘import’ and ‘add’

```
from keras.layers.normalization import BatchNormalization
model.add(Dense(1024))
model.add(BatchNormalization())
model.add(Activation('relu'))
```

  - no hyper parameters need to be set.
  - Some people say it is not clear where to add BN layer, before or after the activation function.

Experiment

- Batch Normalization

```
loss: 0.7510 - accuracy: 0.7623

loss: 0.5866 - accuracy: 0.8245

loss: 0.5133 - accuracy: 0.8443

loss: 0.4620 - accuracy: 0.8584
```

- Dropout 0.3

```
loss: 0.7565 - accuracy: 0.7796

loss: 0.6057 - accuracy: 0.8144

loss: 0.5380 - accuracy: 0.8244

loss: 0.4986 - accuracy: 0.8299
```

2 Dense layers, having 1024, 768 neurons, respectively. Batch normalization really speeds up, and reduces over-fitting. I recommend you to use batch normalization, instead of dropout!

Energy Separation

- My networks have bad behavior in energy.
  - It seems they classify signal / background only using energy.
- I'm trying to separate my data set by energy, and train one by one.

Data Augmentation

- Powerful walk through in image recognition.
  - AlexNet, VGGNet, ResNet
- Crop, reflect, rotate and modify color to extend data set.
- It never fails, generally successes.

Data Augmentation in Cosine

- I think we can move the waveform forward and backward, unless a pulse is cut.
- Looking for more idea!