Optimisation

Ali Akbar Septiandri

Universitas Al Azhar Indonesia

October 21, 2018

Daftar isi

Bahan Bacaan

1. Ruder, S. (2016) An overview of gradient descent optimization algorithms. URL: `http://ruder.io/optimizing-gradient-descent/index.html`
2. Ruder, S. (2017) Optimization for Deep Learning Highlights in 2017. URL: `http://ruder.io/deep-learning-optimization-2017/`
3. Karpathy, A. (2017) Neural Networks Part 2: Setting up the Data and the Loss. URL: `http://cs231n.github.io/neural-networks-2/`
4. Karpathy, A. (2017) Neural Networks Part 3: Learning and Evaluation. URL: `http://cs231n.github.io/neural-networks-3/`

Preprocessing

Best Practices

- Normalisasi data per fitur, i.e. $\mu = 0$ dan nilainya antara $[-1, 1]$
- Inisialisasi *weights* $\sim \mathcal{N}(0, \sqrt{2/n})$ dengan n adalah jumlah neuron masukan
- Gunakan regularisasi L2 dan dropout
- Gunakan *batch normalisation*

Sumber: `http://cs231n.github.io/neural-networks-2/`

Weight Decay

- *Weight decay* atau peluruhan bobot bekerja seperti pegas

Weight Decay

- *Weight decay* atau peluruhan bobot bekerja seperti pegas
- Kalau data latih memberikan gaya yang besar pada suatu bobot, maka nilainya akan mengalahkan nilai peluruhan

Weight Decay

- *Weight decay* atau peluruhan bobot bekerja seperti pegas
- Kalau data latih memberikan gaya yang besar pada suatu bobot, maka nilainya akan mengalahkan nilai peluruhan
- Jika gayanya tidak konsisten ke suatu arah, maka bobotnya akan meluruh ke nol

Weight Decay

- *Weight decay* atau peluruhan bobot bekerja seperti pegas
- Kalau data latih memberikan gaya yang besar pada suatu bobot, maka nilainya akan mengalahkan nilai peluruhan
- Jika gayanya tidak konsisten ke suatu arah, maka bobotnya akan meluruh ke nol
- Konsistensi $\sim$ pola, i.e. bukan *noise*

Weight Decay

- *Weight decay* atau peluruhan bobot bekerja seperti pegas
- Kalau data latih memberikan gaya yang besar pada suatu bobot, maka nilainya akan mengalahkan nilai peluruhan
- Jika gayanya tidak konsisten ke suatu arah, maka bobotnya akan meluruh ke nol
- Konsistensi $\sim$ pola, i.e. bukan *noise*
- Menentukan jumlah parameter yang efektif

L2 Regularisation

- Menambahkan

$$E_{\mathbf{w}} = \frac{1}{2}\lambda \sum_{j=1}^{D} w_j^2$$

  dengan λ adalah hyperparameter yang kita tentukan nilainya

- Turunan parsialnya

$$\frac{\partial E_{\mathbf{w}}}{\partial w_j} = \lambda w_j$$

- Jika digabungkan dengan *loss function* yang dipakai

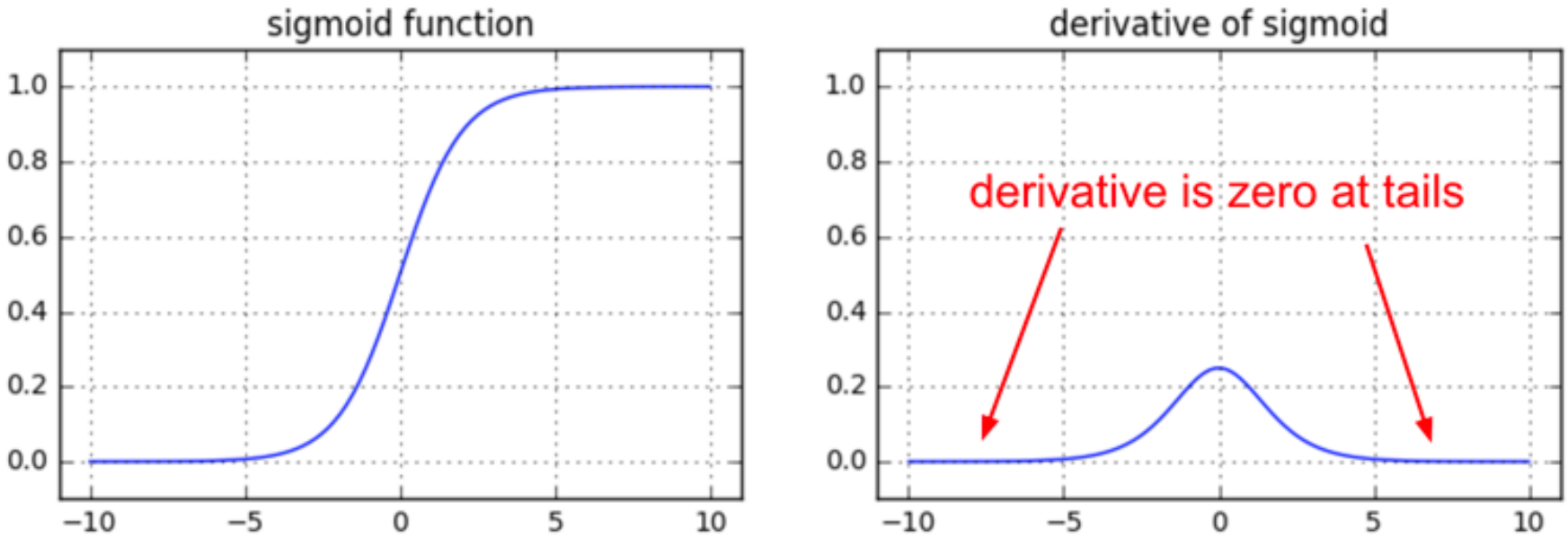
$$\frac{\partial E}{\partial w_j} = \frac{\partial (E_{train} + E_{\mathbf{w}})}{\partial w_j} = \frac{\partial E_{train}}{\partial w_j} + \lambda w_j$$

Batch Normalisation

- Meminjam ide *preprocessing* untuk dilakukan di *tiap layer*
- Mempercepat proses pelatihan
- Biasanya diletakkan setelah *dense layer* dan sebelum *non-linearities*
- Salah satu konsep penting di ResNet [He et al., 2016]

Activation Functions

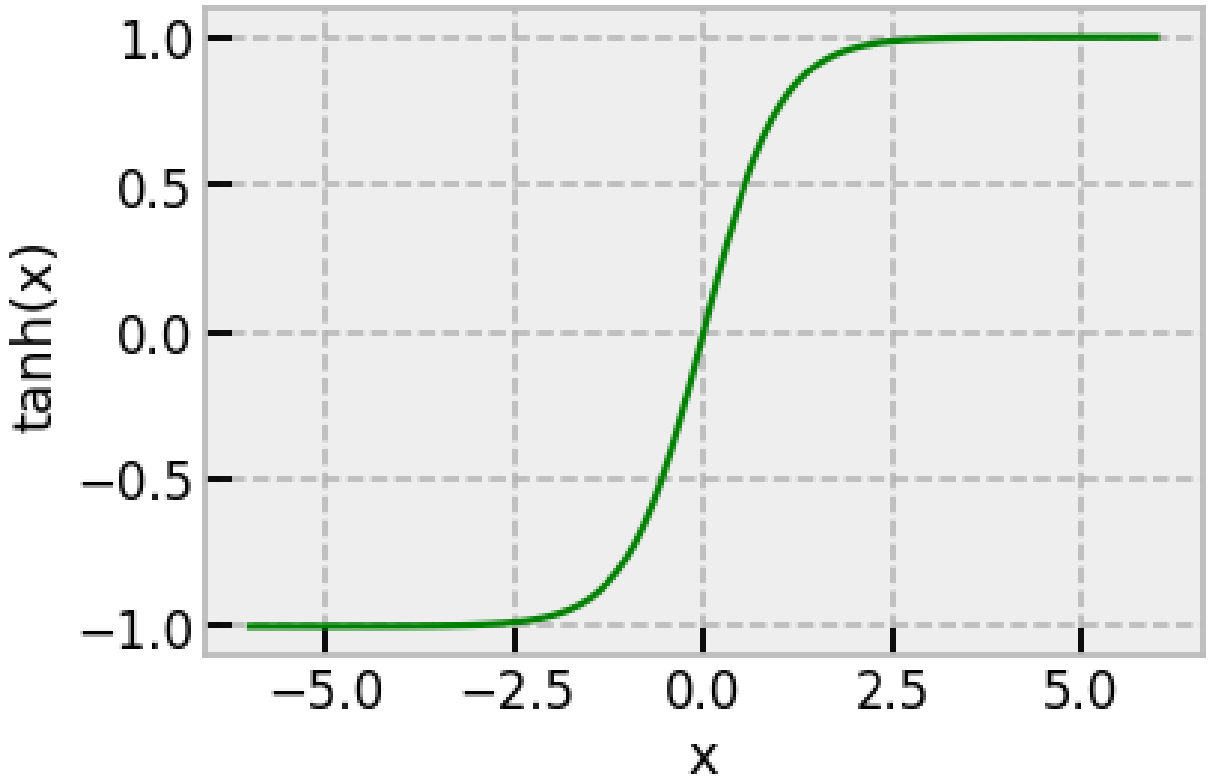
Vanishing Gradients



Gambar: Nilai gradien dari sigmoid yang cepat jenuh [Karpathy, 2016]

Apa alternatif yang bisa dipakai?

Hyperbolic Tangent (tanh)



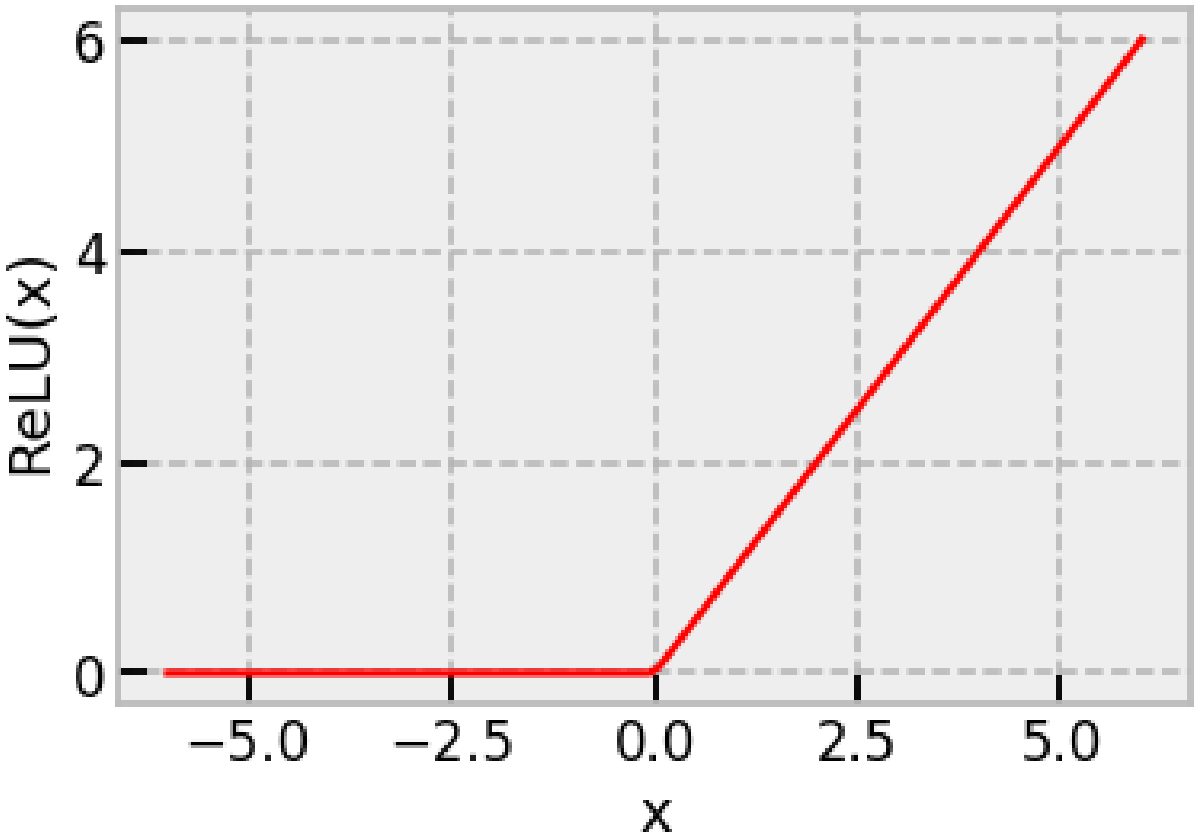
Apa bedanya dengan sigmoid?

Tanh

- Memberikan jangkauan $[-1, 1]$
- Gradient yang lebih besar[1]
- Sifat lainnya mirip dengan sigmoid

[1] Catatan: Akan sangat tergantung inisialisasi weights

ReLU

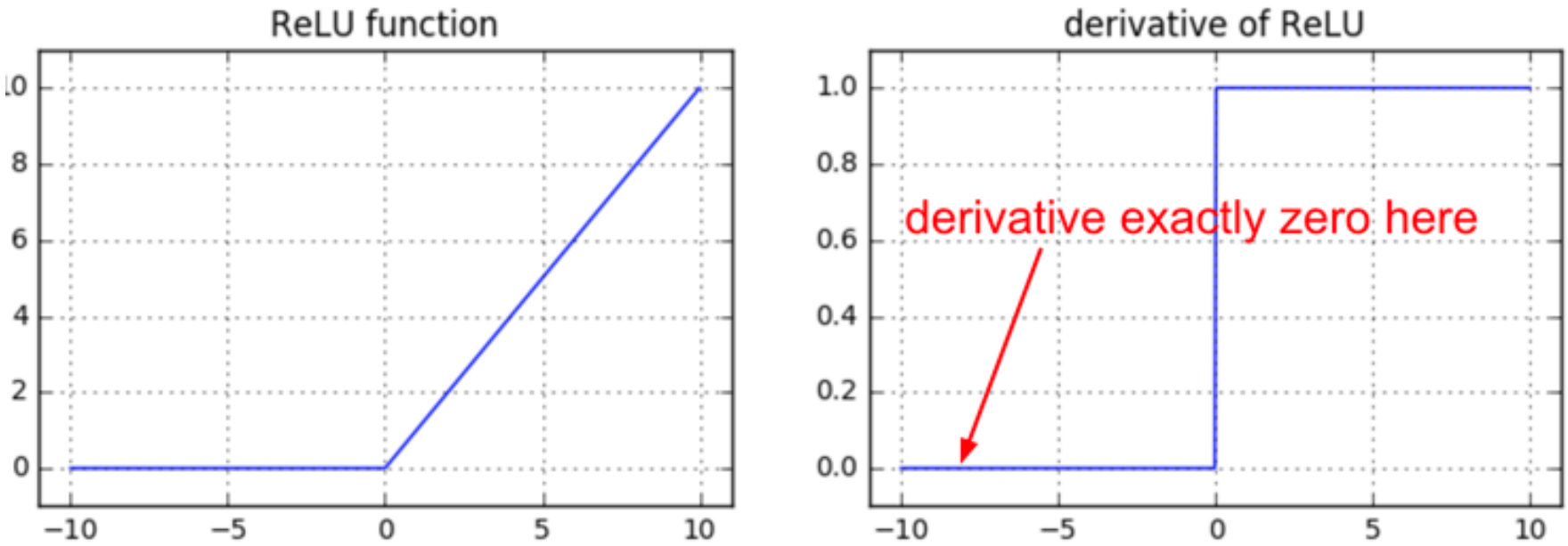


Gambar: Rectified linear unit [Nair, V. & Hinton, G.E., 2010]

ReLU

- Bentuknya $ReLU(x) = max(0, x)$
- Hasil empiris pada kasus suara dan gambar lebih baik daripada sigmoid atau tanh
- Tidak ada titik jenuh pada positif
- Mungkin "mati"

Dying ReLUs



Gambar: Mungkin terjadi kasus keluarannya selalu nol [Karpathy, 2016]

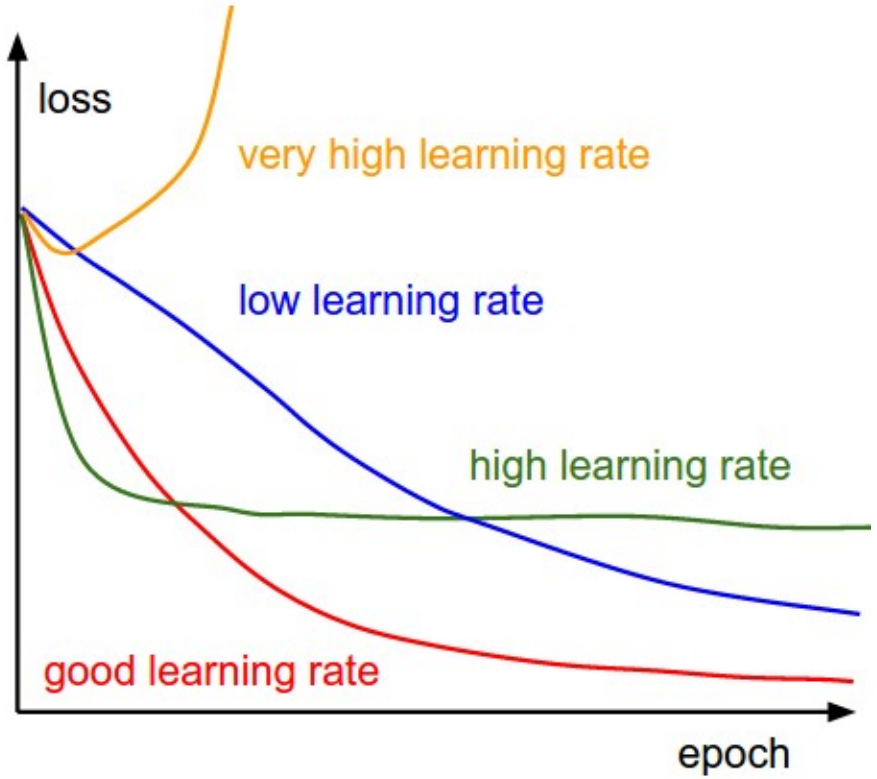
Optimisation

Sanity Check

Sebelum melatih model lebih lanjut (yang biasanya butuh waktu lama), pastikan beberapa hal ini:

- Nilai *loss* inisialisasi sudah benar, e.g. 2.302 pada CIFAR-10 dengan softmax
- Menambahkan regularisasi $\rightarrow$ meningkatkan *loss*
- *Overfit a tiny subset of data*

Learning Rate - Loss



Gambar: Efek nilai η terhadap loss [Karpathy, 2017]

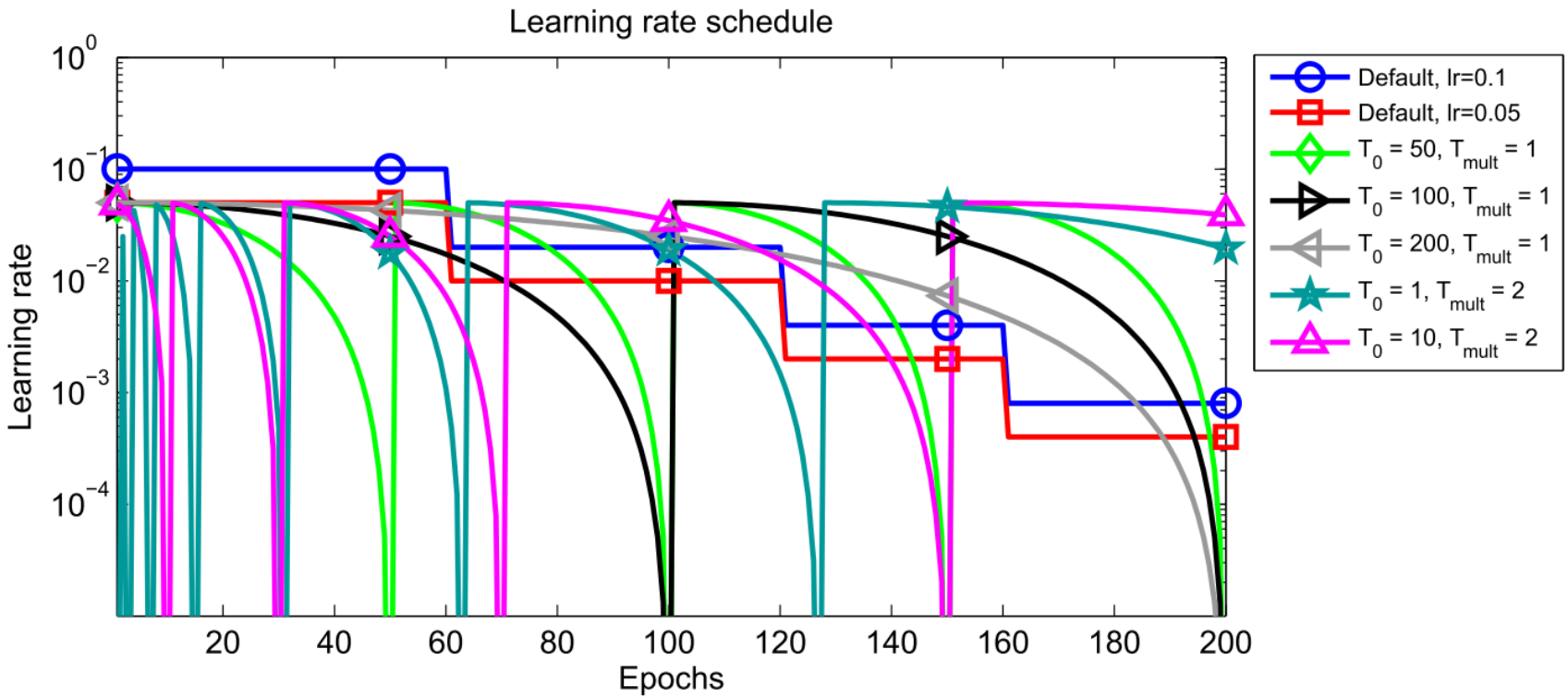
Alternatif dari SGD

- Momentum
- Nesterov accelerated gradient
- Adagrad
- Adadelta
- RMSProp
- Adam
- AdaMax
- ...

Referensi: `http://ruder.io/optimizing-gradient-descent/`

Rekomendasi:
SGD+Nesterov Momentum atau Adam

SGD with Restarts

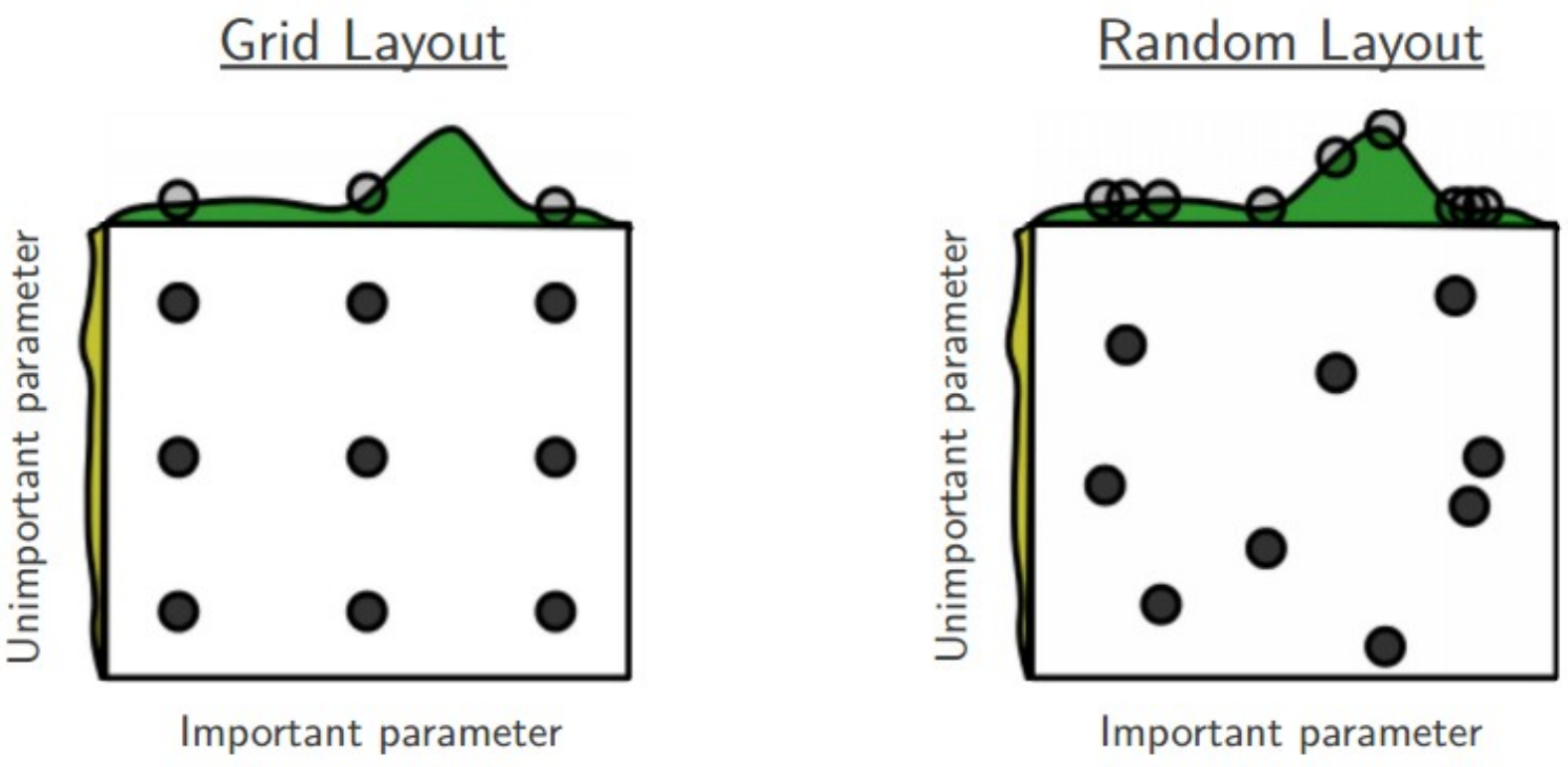


Gambar: SGDR dengan *warm restarts* (Loshchilov and Hutter, 2017)

Best Practices

- Gunakan validation set alih-alih cross-validation
- Coba beberapa nilai *hyperparameters* dalam skala log, e.g. $\eta \in [10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}]$
- Gunakan *random search* alih-alih *grid search* (Bergstra & Bengio, 2012)

Grid vs Random Search



Gambar: Beberapa *hyperparameters* lebih penting dibandingkan yang lain (Bergstra & Bengio, 2012)

Referensi

Andrej Karpathy (20 December 2016)

Yes you should understand backprop

https://medium.com/@karpathy/yes-you-should-understand-backprop-e2f06eab496b

Andrej Karpathy (2017)

Neural Networks Part 3: Learning and Evaluation.

http://cs231n.github.io/neural-networks-3/

Vinod Nair & Geoffrey E. Hinton (2010)

Rectified linear units improve restricted boltzmann machines

ICML (pp. 807-814)

He, K., Zhang, X., Ren, S. & Sun, J.(2016)

Deep residual learning for image recognition

CVPR (pp. 770-778)

Terima kasih