CS224n assignment 3 (Written)

1-(a)-(i)

① How using m stops the updates from varying as much?

: Adam is an incorporated form of Momentum and RMSProp. The term m is a modified term of velocity. The update rule of momentum (below) contains accumulating the gradients. Not only making steps with respect to current gradients, it also makes steps with respect to previous gradient history. Considering both current and previous gradients, step size will be larger if there are successive gradients with same directions, while smaller with different directions — making variance low.

$$\mathbf{m} \leftarrow \beta_1 \mathbf{m} + (1-\beta_1)\nabla_{\boldsymbol{\theta}} J_{\text{minibatch}}(\boldsymbol{\theta})$$
$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \alpha \mathbf{m}$$

② Why low variance may be helpful to learning?

: Because it helps fast convergence toward loss minima.

1-(a)-(ii)

① Since Adam divides the update by $\sqrt{\mathbf{v}}$, which of the model parameters will get larger updates?

: $\sqrt{\mathbf{v}}$ means the squared root sum of all their historical squared values. And according to the update rule below, step size will be larger if $\sqrt{\mathbf{v}}$ gets smaller. Small $\sqrt{\mathbf{v}}$ means small sum of all historical squared gradients. Therefore, model parameters with small partial derivatives will get larger updates.

$$\mathbf{m} \leftarrow \beta_1 \mathbf{m} + (1-\beta_1)\nabla_{\boldsymbol{\theta}} J_{\text{minibatch}}(\boldsymbol{\theta})$$
$$\mathbf{v} \leftarrow \beta_2 \mathbf{v} + (1-\beta_2)(\nabla_{\boldsymbol{\theta}} J_{\text{minibatch}}(\boldsymbol{\theta}) \odot \nabla_{\boldsymbol{\theta}} J_{\text{minibatch}}(\boldsymbol{\theta}))$$
$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \alpha \odot \mathbf{m}/\sqrt{\mathbf{v}}$$

② Why might this help learning?

: It can help solving the saddle-point / local-minima problems.

* Intuition!
편미분 값이 커서 Loss에 민감한 parameter는 천천히 학습되도록 설정 (vice versa)

1-(b)

(i) What must γ equal in terms of p_{drop}?

The value γ should have the value that can make the expected value h_{drop} h.
Therefore, we can compute the expected value as below:

$$E_{p_{drop}}[h_{drop}]_i = \gamma \cdot 0 \cdot h_i p_{drop} + \gamma \cdot 1 \cdot h_i (1-p_{drop})$$
$$= \gamma \cdot h_i (1-p_{drop}) = h_i$$

In conclusion, the value γ must be

$$\gamma = \frac{1}{1-p_{drop}}$$

(ii) Why should we apply dropout during training but not during evaluation?

: Applying dropout during training means we're ensembling each specific network (including dropout). On evaluation process, we check the ensembled result, so we do not apply dropout during evaluation.