My solutions for written questions:

1-c)

Placeholder variables allows us to create a computational graph, without needing the data. Data then can be fed into these placeholder variables through feed dictionaries.

1-e)

Tensorflow is a software for Automatic Differentiation (AD). By "automatic differentiation," we mean that we only need to define the forward pass of our model, or more accurately, the computational graph of our model. Then the backward pass and calculation of all derivatives will be done automatically by Tensorflow.

2-a)

Stack	Buffer	New dependency	transition
[ROOT]	[I, parsed, this, sentence, correctly]		Initial Config.
[ROOT, I]	[parsed, this, sentence, correctly]		SHIFT
[ROOT, I, parsed]	[this, sentence, correctly]		SHIFT
[ROOT, parsed]	[this, sentence, correctly]	parsed $\rightarrow$ I	LEFT-ARC
[ROOT, parsed, this]	[sentence, correctly]		SHIFT
[ROOT, parsed, this, sentence]	[correctly]		SHIFT
[ROOT, parsed, sentence]	[correctly]	sentence $\rightarrow$ this	LEFT-ARC
[ROOT, parsed]	[correctly]	parsed $\rightarrow$ sentence	RIGHT-ARC
[ROOT, parsed, correctly]	[]		SHIFT
[ROOT, parsed]	[]	parsed $\rightarrow$ correctly	RIGHT-ARC
[ROOT]	[]	ROOT $\rightarrow$ parsed	RIGHT-ARC

2-b)

In a sentence, each word has to be shifted onto the stack and then reduced away. In this way, a sentence containing n word, parsed in 2n steps_e.g., In the above example sentence, we saw that the sentence (with 5 words) was parsed in 10 steps.

2-f)

we want to determine γ such that: $\mathbb{E}_{Pdrop}[h_{drop}]_i = h_i$

$\mathbb{E}_{Pdrop}[h_{drop}]_i = \mathbb{E}_{Pdrop}[\gamma \mathrm{dh}]_i = \mathbb{E}_{Pdrop}[\gamma d_i h_i] = p_{drop}(0) + (1 - p_{drop})\gamma h_i$

So we want to be sure that:

$\mathbb{E}_{Pdrop}[h_{drop}]_i = (1 - p_{drop})\gamma h_i = h_i$

So γ must be equal to $1/(1 - p_{drop})$

2-g)

i) Using m, the rolling average of the gradients, each update of **m** will be almost equal to the previous one due to the equation and the fact that $\beta_1 = 0.9$. Only $1 - \beta_1 = 0.1$ portion of m changes in each step. It means that updates vary just slightly in each step. So the movement of the rolling average of gradients toward a local optimum is smoother and faster with less bouncing.

ii) In order to improve the learning process, or reaching to local optimums faster, we want to slow down the learning in direction of parameters that have larger gradients (by assigning smaller updates to them) and speed up the learning in direction of parameters that have smaller gradients. Loss function with respect to parameters that have smaller gradients is flatter, so we want these parameters to get larger updates so that we can reach to local optimums faster.

2-h)

Train loss: 0.0625

Best dev UAS: 88.45

Test UAS: 88.96

3-a)

Recall that $\boldsymbol{\mathcal{Y}}^{(t)}$ is a one-hot vector. Supposing that k-th element of this vector is the only nanozero element, we then can write:

$$CE\left(\mathcal{Y}^{(t)}, \hat{\mathcal{Y}}^{(t)}\right) = -\log\left(\hat{\mathcal{Y}}_k^{(t)}\right) = \log(\frac{1}{\hat{\mathcal{Y}}_k^{(t)}})$$

Similarly we can rewrite perplexity as:

$$PP^{(t)}\left(\mathcal{Y}^{(t)}, \hat{\mathcal{Y}}^{(t)}\right) = \frac{1}{\hat{\mathcal{Y}}_k^{(t)}}$$

Comparing two equations above:

$$CE\left(\mathcal{Y}^{(t)}, \hat{\mathcal{Y}}^{(t)}\right) = \log(PP^{(t)}\left(\mathcal{Y}^{(t)}, \hat{\mathcal{Y}}^{(t)}\right))$$

So minimizing the arithmetic mean of the cross-entropy will also minimize the geometric mean of the perplexity.

If the model predictions were completely random, we would expect $\mathbb{E}\left[\hat{\mathcal{Y}}_k^{(t)}\right] = \frac{1}{|V|}$ so we expect that

$$PP^{(t)}\left(\mathcal{Y}^{(t)}, \hat{\mathcal{Y}}^{(t)}\right) = \frac{1}{\mathbb{E}\left[\hat{\mathcal{Y}}_k^{(t)}\right]} = \frac{1}{\frac{1}{|V|}} = |V|$$

Based on the third equation, cross-entropy is equal to logarithm of perplexity, So:

$$CE\left(\mathcal{Y}^{(t)}, \hat{\mathcal{Y}}^{(t)}\right) = \log(|V|)$$

Then If $|V| = 10000$:

$$CE\left(\mathcal{Y}^{(t)}, \hat{\mathcal{Y}}^{(t)}\right) = \log(10000)$$

3-b)

We Introduce two intermediate variable as below:

$$\mathrm{Z}^{(t)} = h^{(t-1)}H + e^{(t)}I + b_1 \quad \in \mathbb{R}^{D_h}$$

$$\theta^{(t)} = h^{(t)}U + b_2 \quad \in \mathbb{R}^{|V|}$$

So we can rewrite the equations as:

$$h^{(t)} = sigmoid\left(\mathrm{Z}^{(t)}\right) = \sigma(\mathrm{Z}^{(t)}) \quad \in \mathbb{R}^{D_h}$$

$$\hat{y}^{(t)} = softmax(\theta^{(t)}) \quad \in \mathbb{R}^{|V|}$$

We can also define the error signals as:

$$e^{(t)} = x^{(t)}L = L_{x^{(t)}} \quad \in \mathbb{R}^{d}$$

$$\delta_1^{(t)} = \frac{\partial J^{(t)}}{\partial \theta^{(t)}} = \hat{y}^{(t)} - y^{(t)} \quad \in \mathbb{R}^{|V|}$$

$$\delta_2^{(t)} = \frac{\partial J^{(t)}}{\partial \mathrm{Z}^{(t)}} = \frac{\partial J^{(t)}}{\partial \theta^{(t)}}\frac{\partial \theta^{(t)}}{\partial h^{(t)}}\frac{\partial h^{(t)}}{\partial \mathrm{Z}^{(t)}} = \delta_1^{(t)}U^T \odot h^{(t)} \odot (1 - h^{(t)}) \quad \in \mathbb{R}^{D_h}$$

Note that in equation above we use $\frac{d\sigma(z)}{dz} = \sigma'(z) = \sigma(z)\left(1 - \sigma(z)\right)$

Now we can compute the derivative w.r.t the previous hidden layer values:

$$\frac{\partial J^{(t)}}{\partial b_2} = \frac{\partial J^{(t)}}{\partial \theta^{(t)}}\frac{\partial \theta^{(t)}}{\partial b_2} = \delta_1^{(t)} \quad \in \mathbb{R}^{|V|}$$

$$\frac{\partial J^{(t)}}{\partial L_{x^{(t)}}} = \frac{\partial J^{(t)}}{\partial \mathrm{Z}^{(t)}}\frac{\partial \mathrm{Z}^{(t)}}{\partial e^{(t)}}\frac{\partial e^{(t)}}{\partial L_{x^{(t)}}} = \delta_2^{(t)}I^T \quad \in \mathbb{R}^{d}$$

$$\left.\frac{\partial J^{(t)}}{\partial I}\right|_{(t)} = \frac{\partial \mathrm{Z}^{(t)}}{\partial I}\frac{\partial J^{(t)}}{\partial \mathrm{Z}^{(t)}} = (e^{(t)})^T\delta_2^{(t)} \quad \in \mathbb{R}^{d\times D_h}$$

$$\left.\frac{\partial J^{(t)}}{\partial H}\right|_{(t)} = \frac{\partial \mathrm{Z}^{(t)}}{\partial H}\frac{\partial J^{(t)}}{\partial \mathrm{Z}^{(t)}} = (h^{(t-1)})^T\delta_2^{(t)} \quad \in \mathbb{R}^{D_h\times D_h}$$

$$\frac{\partial J^{(t)}}{\partial h^{(t-1)}} = \frac{\partial J^{(t)}}{\partial \mathrm{Z}^{(t)}}\frac{\partial \mathrm{Z}^{(t)}}{\partial h^{(t-1)}} = \delta_2^{(t)}H^T \quad \in \mathbb{R}^{D_h}$$

3-c)

For convenience:

$$\sigma'\left(\mathrm{Z}^{(t-1)}\right) = diag\left(\sigma\left(\mathrm{Z}^{(t-1)}\right) \odot \left(1-\sigma\left(\mathrm{Z}^{(t-1)}\right)\right)\right) = diag(h^{(t-1)} \odot \left(1-h^{(t-1)}\right))$$

So we can write derivatives as:

$$\frac{\partial J^{(t)}}{\partial L_{x^{(t-1)}}} = \frac{\partial J^{(t)}}{\partial h^{(t-1)}} \frac{\partial h^{(t-1)}}{\partial \mathrm{Z}^{(t-1)}} \frac{\partial \mathrm{Z}^{(t-1)}}{\partial e^{(t-1)}} \frac{\partial e^{(t-1)}}{\partial L_{x^{(t-1)}}} = \delta^{(t-1)} \sigma'(\mathrm{Z}^{(t-1)}) I^T \qquad \in \mathbb{R}^d$$

$$\left.\frac{\partial J^{(t)}}{\partial I}\right|_{(t-1)} = \frac{\partial \mathrm{Z}^{(t-1)}}{\partial I} \frac{\partial h^{(t-1)}}{\partial \mathrm{Z}^{(t-1)}} \frac{\partial J^{(t)}}{\partial h^{(t-1)}} = (e^{(t-1)})^T \delta^{(t-1)} \sigma'(\mathrm{Z}^{(t-1)}) \qquad \in \mathbb{R}^{d \times D_h}$$

$$\left.\frac{\partial J^{(t)}}{\partial H}\right|_{(t-1)} = \frac{\partial \mathrm{Z}^{(t-1)}}{\partial H} \frac{\partial h^{(t-1)}}{\partial \mathrm{Z}^{(t-1)}} \frac{\partial J^{(t)}}{\partial h^{(t-1)}} = (h^{(t-2)})^T \delta^{(t-1)} \sigma'(\mathrm{Z}^{(t-1)}) \qquad \in \mathbb{R}^{D_h \times D_h}$$

3-d)

We need three main matrix-vector multiplication for forward propagation:

$h^{(t-1)}H = (1 \times D_h)(D_h \times D_h) \rightarrow number\ of\ operations \approx D_h D_h$

$e^{(t)}I = (1 \times d)(d \times D_h) \ \rightarrow number\ of\ operations \approx dD_h$

$h^{(t)}U = (1 \times D_h)(D_h \times |V|) \rightarrow number\ of\ operations \approx D_h|V|$

So:

$O(D_h|V| + dD_h + D_h D_h)$

For each forward path we have one backward path. So for τ timesteps, we have as much operation as:

$O(\tau(D_h|V| + dD_h + D_h D_h))$

Assuming $|V| \gg D_h$, The slowest part is $O(D_h|V|)$ term because