Deep-Learning-TAU-0510-7255-Spring-2020

Homework 2

Ophir Shurany 304867716

Tim Mironov 321297111

Theory: Question 1

a. To handle variable length **input** sequences, we can do the following:
   i. **Padding**: Padding can be used, whereby you would have to fix the length of each sample (either to the length of the longest sample, or to a fixed length — longer samples would be trimmed or filtered somehow to fit into that length)
   ii. **Tokenize (encoder**): create a state vector of the input by process the inputs one by one by the encoder RNN, while disregarding the output. The RNN should stop by an agreed EOS sign.
   iii. **Length Information**: set the *sequence_length* argument when calling the *dynamic_rnn()* (or *static_rnn()*) function; it must be a 1D tensor indicating the length of the input sequence for each instance.

b. To handle variable length **output** sequences, we can do the following:
   i. **Tokenize (decoder)**: Define a special output called an end-of-sequence token (EOS token). Any output past the EOS should be ignored
   ii. **Length Information**: If you know in advance what length each sequence will have (for example if you know that it will be the same length as the input sequence), then you can set the *sequence_length* parameter as described above (a.II)
   iii. **Last Relevant Output**: For sequence classification, we want to feed the last output of the recurrent network into a predictor, e.g. a softmax layer. While taking the last frame worked well for fixed-sized sequences, we do not have to select the last relevant frame.

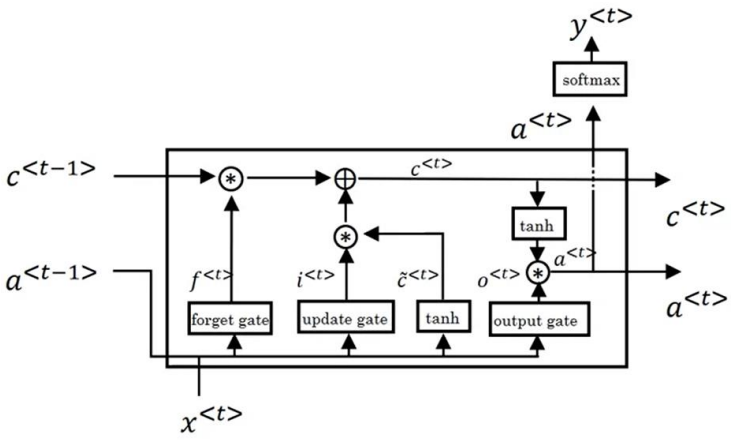
Theory: Question 2

The advantages of the GRU:

   i. **Architecture**: GRU is a simpler model and so it is easier to build a much bigger network.
   ii. **Computations**: GRU has only has two gates (LSTM has three gates), so computationally, it runs a bit faster than LSTM.

Theory: Question 3

LSTM structure:



LSTM equations:

$$\tilde{c}^{<t>} = \tanh(W_c \times [a^{<t-1>}, x^{<t>}] + b_c)$$
$$\Gamma_u = \sigma(W_u \times [a^{<t-1>}, x^{<t>}] + b_u)$$
$$\Gamma_f = \sigma\left(W_f \times [a^{<t-1>}, x^{<t>}] + b_f\right)$$
$$\Gamma_o = \sigma(W_o \times [a^{<t-1>}, x^{<t>}] + b_o)$$
$$c^{<t>} = \Gamma_u * \tilde{c}^{<t>} + \Gamma_f * c^{<t-1>}$$
$$a^{<t>} = \Gamma_o * \tanh(c^{<t>})$$

Where W_a is a compressed notation $[W_{aa} \ ; \ W_{ax}] \rightarrow DIM\{W_a\} = (n, n+d) \underbrace{= n^2 + nd}_{\#parameters}$

$$DIM\{[a^{<t-1>}, x^{<t>}]\} = (n+d, 1) \Rightarrow DIM\left\{\tilde{c}^{<t>}, \Gamma_{u,f,o}\right\} = (n, 1)$$

$$\rightarrow DIM\{b\} = (n, 1) \quad \underbrace{= n}_{\#parameters}$$

Total parameters calculating for 3 gates and 1 cell:

$$4 \cdot (W_a + b) = \boxed{4 \cdot \left(n^2 + nd + n\right)}$$
$$for\ n = d = 200 \Rightarrow 320{,}800\ \#\ parameters$$

Theory: Question 4

The GRU equations are defined as follows:

$$z_t = \sigma\left(W_{xz}^T \cdot x_t + W_{hz}^T \cdot h_{t-1} + b_z\right) = \sigma(\widetilde{z_t})$$
$$r_t = \sigma\left(W_{xr}^T \cdot x_t + W_{hr}^T \cdot h_{t-1} + b_r\right) = \sigma(\widetilde{r_t})$$
$$g_t = \tanh\left(W_{xg}^T \cdot x_t + W_{hg}^T \cdot (r_t \otimes h_{t-1}) + b_g\right) = \tanh(\widetilde{g_t})$$
$$h_t = z_t \otimes h_{t-1} + (1 - z_t) \otimes g_t$$
$$and\ \sigma(x) = \frac{1}{1+e^{-x}}$$
$$\rightarrow \frac{\partial\sigma}{\partial x} = \frac{e^{-x}}{(1+e^{-x})^2} \equiv \frac{1+e^{-x}}{(1+e^{-x})^2} - \frac{1}{(1+e^{-x})^2} = \frac{1}{1+e^{-x}} - \frac{1}{(1+e^{-x})^2}$$
$$\Rightarrow \boxed{\frac{\partial\sigma}{\partial x} = \sigma(x)\left(1-\sigma^2(x)\right)}$$

Given is a two iteration GRU network with a defined loss function:

$$\epsilon_t = \frac{1}{2}(h_t - y_t)^2$$

The gradient $\frac{\partial\epsilon_2}{\partial h_2}$ is given. The BP gradients of the second time stamp are calculated as follows:

a) $\frac{\partial\epsilon_2}{\partial W_{xz}} = \frac{\partial\epsilon_2}{\partial h_2}\frac{\partial h_2}{\partial z_2}\frac{\partial z_2}{\partial W_{xz}} = \frac{\partial\epsilon_2}{\partial h_2}(h_1 - g_2)\cdot\sigma(\widetilde{z_2})(1-\sigma(\widetilde{z_2})^2)\cdot x_2$

$where\ \widetilde{z_2} = W_{xz}^T \cdot x_2 + W_{hz}^T \cdot h_1 + b_z$

b) $\frac{\partial\epsilon_2}{\partial W_{hz}} = \frac{\partial\epsilon_2}{\partial h_2}\frac{\partial h_2}{\partial z_2}\frac{\partial z_2}{\partial W_{hz}} = \frac{\partial\epsilon_2}{\partial h_2}(h_1 - g_2)\sigma(\widetilde{z_2})(1-\sigma(\widetilde{z_2})^2)\cdot h_1$

$where\ \widetilde{z_2} = W_{xz}^T \cdot x_2 + W_{hz}^T \cdot h_1 + b_z$

c) $\frac{\partial\epsilon_2}{\partial W_{xg}} = \frac{\partial\epsilon_2}{\partial h_2}\frac{\partial h_2}{\partial g_2}\frac{\partial g_2}{\partial W_{xg}} = \frac{\partial\epsilon_2}{\partial h_2}(1-z_2)\cdot(1-\tanh^2(\widetilde{g_2}))\cdot x_2$

$where\ \widetilde{g_2} = W_{xg}^T \cdot x_2 + W_{hg}^T \cdot (r_2 \otimes h_1) + b_g$

d) $\frac{\partial\epsilon_2}{\partial W_{hg}} = \frac{\partial\epsilon_2}{\partial h_2}\frac{\partial h_2}{\partial g_2}\frac{\partial g_2}{\partial W_{hg}} = \frac{\partial\epsilon_2}{\partial h_2}(1-z_2)(1-\tanh^2(\widetilde{g_2}))\cdot r_2\cdot h_1$

$where\ \widetilde{g_2} = W_{xg}^T \cdot x_2 + W_{hg}^T \cdot (r_2 \otimes h_1) + b_g$

e) $\frac{\partial\epsilon_2}{\partial W_{xr}} = \frac{\partial\epsilon_2}{\partial h_2}\frac{\partial h_2}{\partial g_2}\frac{\partial g_2}{\partial r_2}\frac{\partial r_2}{\partial W_{xr}} = \frac{\partial\epsilon_2}{\partial h_2}(1-z_2)(1-\tanh^2(\widetilde{g_2}))\left(W_{hg}^T \cdot h_1\right)\sigma(\widetilde{r_2})(1-\sigma(\widetilde{r_2})^2)\cdot x_2$

$where\ \widetilde{g_2} = W_{xg}^T \cdot x_2 + W_{hg}^T \cdot (r_2 \otimes h_1) + b_g$

$and\ \widetilde{r_2} = W_{xr}^T \cdot x_2 + W_{hr}^T \cdot h_1 + b_r$

f) $\frac{\partial\epsilon_2}{\partial W_{hr}} = \frac{\partial\epsilon_2}{\partial h_2}\frac{\partial h_2}{\partial g_2}\frac{\partial g_2}{\partial r_2}\frac{\partial r_2}{\partial W_{hr}} = \frac{\partial\epsilon_2}{\partial h_2}(1-z_2)(1-\tanh^2(\widetilde{g_2}))\left(W_{hg}^T \cdot h_1\right)\sigma(\widetilde{r_2})(1-\sigma(\widetilde{r_2})^2)\cdot h_1$

$where\ \widetilde{g_2} = W_{xg}^T \cdot x_2 + W_{hg}^T \cdot (r_2 \otimes h_1) + b_g$

$and\ \widetilde{r_2} = W_{xr}^T \cdot x_2 + W_{hr}^T \cdot h_1 + b_r$

Practical Part:

Architecture:

As the general guidelines for the proposed architecture we used the referenced article "Recurrent Neural Network Regularization" by Zaremba. As requested the trained models used two layers, each with 200 hidden units.

Similar to the article the clip norm technique was applied on the gradients during the training and decaying learning rate schedule was used. In the LSTM and GRU models, where Dropout had to be tested, the technique was applied only on the non-recurrent connections, as suggested in the article.

Four different models were totally tested:

1. LSTM Based with Drop Out:
   - 2 Hidden levels with DO of 0.4 on non-recurrent connections.
   - Trained for 70 epochs with decaying learning rate.
   - SGD optimizer was used with “clip norm” on 5 and momentum of 0.8

2. LSTM Based non-regularized:
   - 2 Hidden levels, no dropout applied.
   - Trained for 70 epochs with decaying learning rate.
   - SGD optimizer was used with “clip norm” on 5 and momentum of 0.8

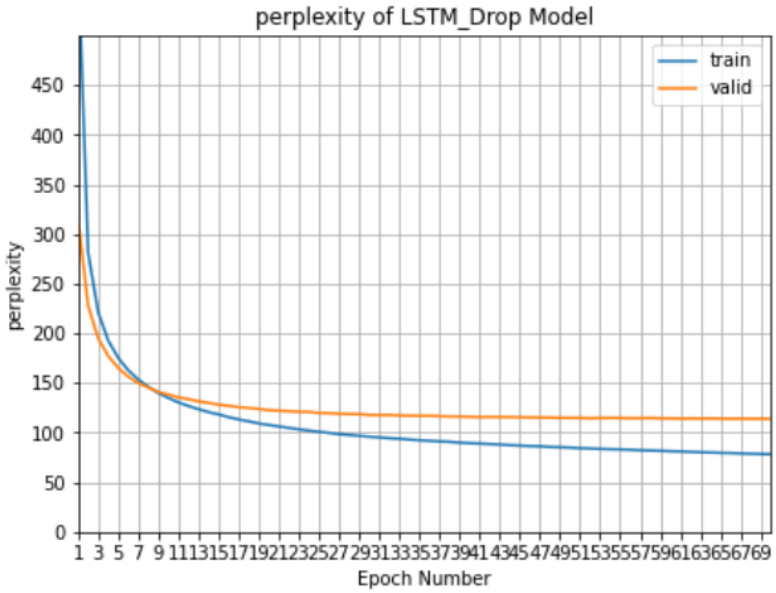
3. GRU Based with Drop Out:
   - 2 hidden levels with 0.4 dropout.
   - Decaying learning rate.

4. GRU Based non-regularized:
   - 2 hidden levels without dropout.
   - Decaying learning rate.

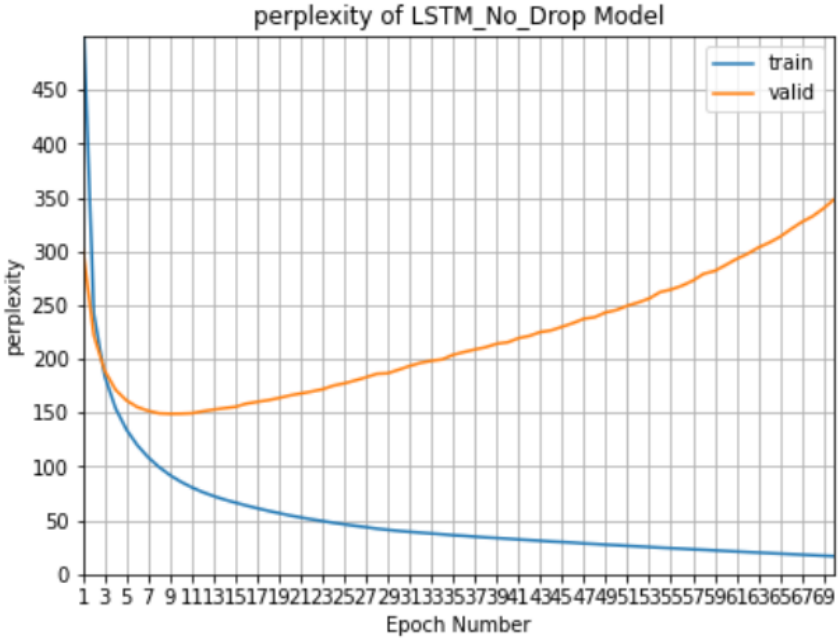
All models were trained on a sequence length of 25 and mini-batches of 20, using Colab GPUs.

Training Graphs:

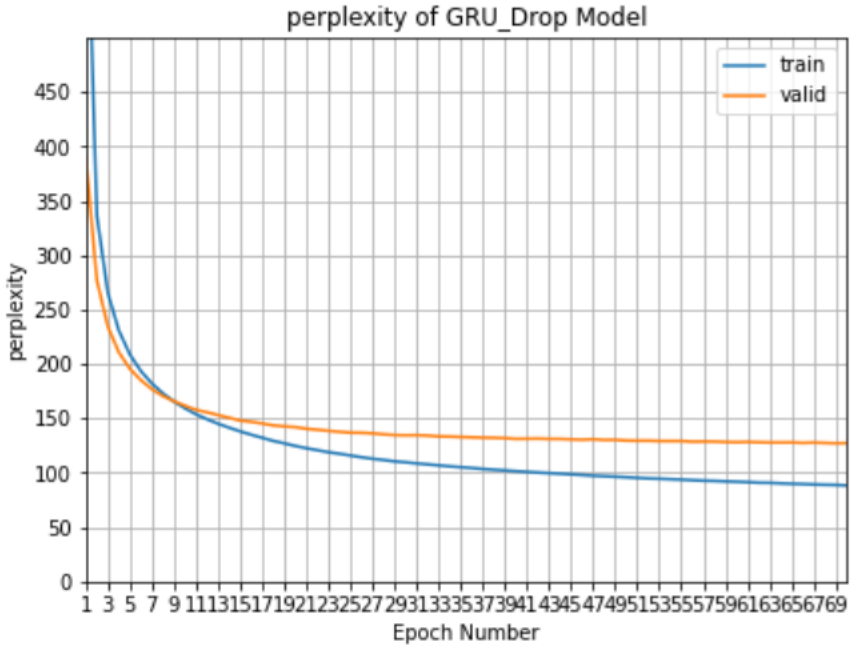
1. LSTM Based with Drop Out:
   The model was trained for 70 epochs with Dropout rate of 0.4 applied on non-recurrent connections. The custom Learning Scheduler fixed the learning rate on 1 for 15 first epochs with subsequent decay of 0.98 per epoch:



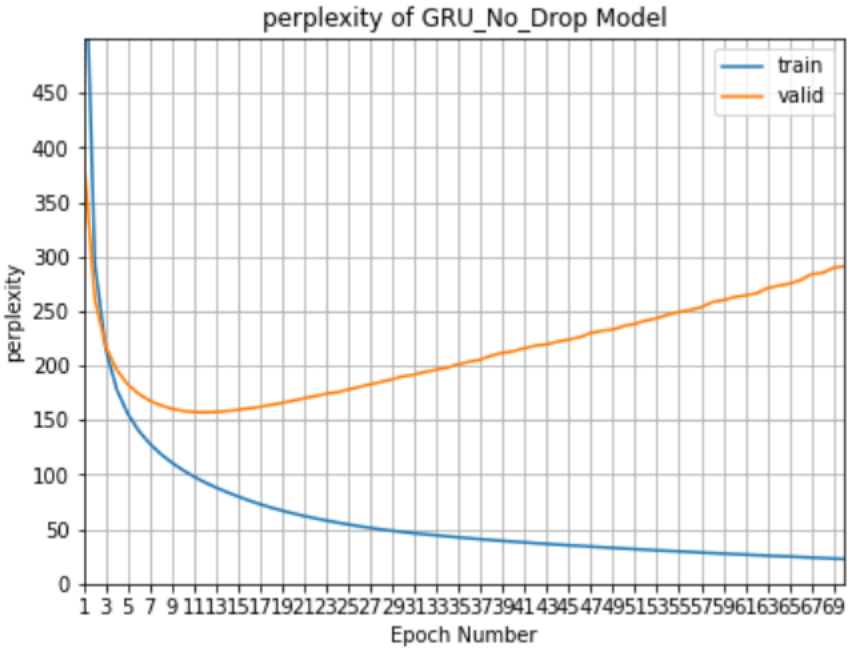
2. LSTM Based non-regularized:
   This model was trained using the same setup as previous, but without the Drop Out applied. The best weights were save on epoch 15 since after that the model quickly entered over-training. The remaining epochs history was saved and the model fit was not terminated only to serve the purposes of presenting the "over fitting" pattern:



3. GRU Based with Drop Out:
   This model was trained using dropout rate of 0.4 with decaying learning rate by the ratio of 0.98. First 15 epochs were trained with fixed rate of 1 and a total training was of 70 epochs.



4. GRU Based non-regularized:
   This model was trained using the same architecture as in previous model, but without applying the dropout technique.



Results:

The proposed models were trained and their best weights according to the validation set were saved. The models were reconstructed and evaluated again to confirm the training and validation set scores of perplexity and to evaluate the models on un-seen testing set:

	Train	Validation	Test
LSTM Dropout	49.85	113.83	106.74
LSTM No DO	90.19	149.03	143.03
GRU Dropout	59.71	127.12	120.87
GRU No DO	92.46	157.19	150.97

Conclusions:

We can see that overall the **LSTM based models achieved better results** on the dataset than the GRU based models: perplexity of ~106/143 vs. ~120/150 on the test dataset. This might be a result of the known tradeoff between GRU and LSTM – complexity vs. computational time, since GRU has simpler architecture than LSTM, but its calculations tend to be faster. Another reason might be an insufficient hyper parameters exploration, which ideally would require more learning schedules testing along with multiple optimizer variations.

Another obvious conclusion from the experiment results is the **superiority of the "drop out applied" models** over the non-regularized models. We see a 40 decrease in perplexity for LSTM and almost a 30 improvement of the GRU for the test dataset. In addition a **clear over fitting pattern** is emerged for both models when trained without DO for >50 number of epochs. Contrarily to the fact that 200 hidden units' models are considered "small" – these models are prone to over-fitting as well; and **drop out technique applied on non-recurrent connections** can improve the results significantly.