Advanced Methods in NLP– Spring 2017

Assignment #2

Idan Rejwan (204085948)
Avi Caciularu (203056585)

April 2017

Question 1

a) Implemented in code.

b) Implemented in code.
Our results: For λ_1 and λ_2 interpolation parameters greed search (λ_3 is just the complementary), we got the following table:

λ_2, λ_1	0	0.1	0.2	0.3	0.4
0	189.624432	104.210141	87.508771	79.029286	74.226364
0.1	115.664846	82.634682	71.625713	65.419878	61.614176
0.2	94.729557	72.622319	64.312655	59.467058	56.499031
0.3	82.687762	66.030838	59.421234	55.554704	53.305675
0.4	74.629917	61.318553	55.934771	52.887356	51.411267
0.5	68.859828	57.858234	53.474881	51.271615	51.056755
0.6	64.620749	55.376331	51.972257	51.076132	
0.7	61.563957	53.844768	51.850421		
0.8	59.627504	53.719830			
0.9	59.293246				

λ_2, λ_1	0.5	0.6	0.7	0.8	0.9
0	71.732161	71.140803	72.652028	77.439629	90.453036
0.1	59.390019	58.488045	59.059968	62.137834	
0.2	54.885269	54.580978	56.248181		
0.3	52.407144	53.319851			
0.4	51.752504				
0.5					
0.6					
0.7					
0.8					
0.9					

We get that the combination $\lambda_1 = 0.4, \lambda_2 = 0.5, \lambda_3 = 1 - \lambda_2 - \lambda_1 = 0.1$ gives the best results, meaning the model takes unigram probabilities into a smaller consideration.

In total, we get: #trigrams: 413540

#bigrams: 122930

#unigrams: 2000

#tokens: 1118296

For $\lambda_1 = 0.4, \lambda_2 = 0.5, \lambda_3 = 1 - \lambda_2 - \lambda_1 = 0.1$ we get:

#perplexity: 51.0567551405

Question 2

a) First we substitute $\hat{\mathbf{y}} = \text{softmax}(\theta)$ in the loss function to get:

$$CE(\mathbf{y}, \hat{\mathbf{y}}) = -\sum_i y_i \log\left(\frac{\exp(\theta_i)}{\sum_j \exp(\theta_j)}\right) = -\sum_i y_i\Big(\theta_i - \log\big(\sum_j \exp(\theta_j)\big)\Big)$$

Now we derive it with regard to θ:

$$\frac{\partial J}{\partial \theta_k} = -y_k + \sum_i y_i \frac{\exp(\theta_k)}{\sum_j \exp(\theta_j)} = -y_k + \Big(\sum_i y_i\Big)\,\text{softmax}(\theta)_k$$

Now, recall that $\mathbf{y}$ is an one-hot vector with one in the correct word $y_o = 1$. Thus in the second term $\sum_i y_i = 1$ and we have:

$$\frac{\partial J}{\partial \theta_k} = \begin{cases} \text{softmax}(\theta)_o - 1 & k = o \\ \text{softmax}(\theta)_k & k \neq o \end{cases}$$

Thus, using vector notation (recall that $\hat{\mathbf{y}} = \text{softmax}(\theta)$):

$$\frac{\partial J}{\partial \theta} = \hat{\mathbf{y}} - \mathbf{y}$$

b) For one hidden layer neural network we have:

$$\theta = \mathbf{h}\mathbf{W_2} + \mathbf{b_2}, \;\; \mathbf{h} = \sigma(\mathbf{z}), \;\; \mathbf{z} = \mathbf{x}\mathbf{W_1} + \mathbf{b_1}$$

We use chain rule:

$$\frac{\partial J}{\partial \mathbf{x}} = \frac{\partial J}{\partial \theta}\frac{\partial \theta}{\partial \mathbf{h}}\frac{\partial \mathbf{h}}{\partial \mathbf{z}}\frac{\partial \mathbf{z}}{\partial \mathbf{x}} \qquad (*)$$

where:

$$\frac{\partial \theta}{\partial \mathbf{h}} = \mathbf{W}_2^T, \;\; \frac{\partial \mathbf{h}}{\partial \mathbf{z}} = \sigma(z)\big(1 - \sigma(z)\big), \;\; \frac{\partial \mathbf{z}}{\partial \mathbf{x}} = \mathbf{W}_1^T$$

Substitute all in $(*)$ to get:

$$\frac{\partial J}{\partial \mathbf{x}} = \bigg(\Big((\hat{\mathbf{y}} - \mathbf{y})\mathbf{W_2^T}\Big) \circ \Big(\sigma(\mathbf{z})\big(\mathbf{1} - \sigma(\mathbf{z})\big)\Big)\bigg)\mathbf{W_1^T}$$

where $\circ$ denotes element-wise product.

c) Implemented in code.

d) Implemented in code.

The results that we got:

Total parameters learned: 104550

Training examples: 1118296

The perplexity we got on the dev dataset: 111.31