CS 4248
Natural Language Processing

Professor NG Hwee Tou

Department of Computer Science
School of Computing
National University of Singapore
nght@comp.nus.edu.sg

Materials

- NNM4NLP Chapter 9

Language Modeling

- Language modeling: The task of assigning a probability to a sequence of words, or equivalently, assigning a probability of a word following a sequence of words

$$P(w_i|w_1, w_2, \dots, w_{i-1}) = \frac{P(w_1, w_2, \dots, w_{i-1}, w_i)}{P(w_1, w_2, \dots, w_{i-1})}$$

$$P(w_1, \dots, w_i)$$
$$= P(w_1) \times P(w_2|w_1) \times \cdots \times P(w_i|w_1, \dots, w_{i-1})$$

Limitations of N-Gram LMs

- Requires intricate and manually designed smoothing schemes
- Computationally expensive to scale to larger N-gram LMs
  - Number of observed n-grams grows at least multiplicatively when n-gram size increases by 1.
- Lack of generalization across contexts
  - Having observed *black car* and *blue car* does not influence the estimate of *red car*

Neural Language Models

- Alleviate the need for manually designing smoothing schemes
- Allow conditioning on increasingly large context sizes with only a linear increase in the number of parameters
- Support generalization across different contexts

Neural Language Model

- Input to multilayer perceptron: a sequence of k words
- Output: a probability distribution over the next word

Neural Language Model

$$v(w) = \boldsymbol{E}_{[w]}$$

$$\boldsymbol{x} = [v(w_1); v(w_2); \ldots; v(w_k)]$$

$$\boldsymbol{h} = g(\boldsymbol{x}\boldsymbol{W}^{\mathbf{1}} + \boldsymbol{b}^{\mathbf{1}})$$

$$\widehat{\boldsymbol{y}} = P(w_i | w_{1:k}) = LM(w_{1:k}) = \mathrm{softmax}(\boldsymbol{h}\boldsymbol{W}^{\mathbf{2}} + \boldsymbol{b}^{\mathbf{2}})$$

$$w_i \in V \quad \boldsymbol{E} \in \mathbb{R}^{|V| \times d_w} \quad \boldsymbol{W}^{\mathbf{1}} \in \mathbb{R}^{k \cdot d_w \times d_{\mathrm{hid}}} \quad \boldsymbol{b}^{\mathbf{1}} \in \mathbb{R}^{d_{\mathrm{hid}}}$$

$$\boldsymbol{h} \in \mathbb{R}^{d_{\mathrm{hid}}} \quad \boldsymbol{W}^{\mathbf{2}} \in \mathbb{R}^{d_{\mathrm{hid}} \times |V|} \quad \boldsymbol{b}^{\mathbf{2}} \in \mathbb{R}^{|V|}$$

Neural Language Model

- V is a finite vocabulary, including UNK for unknown words, $\langle s \rangle$ for sentence initial padding, and $\langle /s \rangle$ for end-of-sentence marking
- Training examples: sequences of $k+1$ words from a corpus, where the first k words are used as features, and the last $(k+1)$th word is used as the target label for classification

Neural Language Model

- Loss function: categorical cross-entropy loss (negative log likelihood)

$$\boldsymbol{y} = \boldsymbol{y}_{[1]}, \dots, \boldsymbol{y}_{[n]} = \text{one-hot vector} \quad n = |V|$$

$$\widehat{\boldsymbol{y}} = \widehat{\boldsymbol{y}}_{[1]}, \dots, \widehat{\boldsymbol{y}}_{[n]}$$

$$L_{\text{cross-entropy}}(\widehat{\boldsymbol{y}}, \boldsymbol{y}) = -\log(\widehat{\boldsymbol{y}}_{[t]})$$

- t is the index of the correct word

Neural Language Model

- Neural LM can be trained on raw texts, i.e., practically unlimited quantity
- Require a softmax operation, which is costly for very large vocabulary

Word Representations as Byproduct

- Word representations: $|V|$ rows of matrix $\boldsymbol{E}$ (dimension: d_w)