PA1 Report

Kevin Lin

1/30/2025

Part 1: DAN

Part 2: BPE

Part 3: Skip-Gram

Q1

3a) We are given the skip-gram model defined as:

$$P(\text{context} = y|\text{word} = x) = \frac{\exp(\mathbf{v}_x \cdot \mathbf{c}_y)}{\sum_{y'} \exp(\mathbf{v}_x \cdot \mathbf{c}_{y'})}$$

where x is the "center word", y is a "context word" being predicted, and $\mathbf{v}_x$ and $\mathbf{c}_y$ are d-dimensional vectors corresponding to words and contexts respecitvely. Each word has independent vectors for each, thus each word has two embeddings.
Given the sentences:

```
the dog
the cat
a dog
```

window size of $k = 1$, we get the training examples: $(x = \text{the}, y = \text{dog})$, $(x = \text{dog}, y = \text{the})$. Consequently, the skip-gram objective, log-likelihood is $\sum_{(x,y)} \log P(y|x)$. With word and context embeddings of dimension $d = 2$, the context embedding vectors w for *dog* and *cat* are both $(0, 1)$, and the embeddings vectors w for *a* and *the* are $(1, 0)$. Thus, the set of probabilities $P(y|\text{the})$ that maximize the log-likelihood are:

$$P(y|\text{the}) = \begin{cases} \frac{1}{2} & y = \text{dog} \\ \frac{1}{2} & y = \text{cat} \\ 0 & y = \text{a} \\ 0 & y = \text{the} \end{cases}$$

3b) We want a setting $\mathbf{v}_{\text{the}}$ where $P(\text{dog}|\text{the}) \approx 0.5$, $P(\text{cat}|\text{the}) \approx 0.5$, and $P(\text{a}|\text{the}), P(\text{the}|\text{the}) \approx 0$. We know we can calculate $P(y|\text{the})$ as:

$$P(y|\text{the}) = \frac{\exp(\mathbf{v}_{\text{the}} \cdot \mathbf{c}_y)}{\sum_{y'} \exp(\mathbf{v}_{\text{the}} \cdot \mathbf{c}_{y'})}$$

$$= \frac{\exp(\mathbf{v}_{\text{the}} \cdot \mathbf{c}_y)}{\exp(\mathbf{v}_{\text{the}} \cdot \mathbf{c}_{\text{dog}}) + \exp(\mathbf{v}_{\text{the}} \cdot \mathbf{c}_{\text{cat}}) + \exp(\mathbf{v}_{\text{the}} \cdot \mathbf{c}_{\text{a}}) + \exp(\mathbf{v}_{\text{the}} \cdot \mathbf{c}_{\text{the}})}$$

We know that $\mathbf{c}_{\text{dog}} = \mathbf{c}_{\text{cat}} = (0, 1)$ and $\mathbf{c}_{\text{a}} = \mathbf{c}_{\text{the}} = (1, 0)$. Thus, we want the dot products of cat and dog to be very large and positive, while minimizing the dot products of a and the. Therefore, we can look for settings of $\mathbf{v}_{\text{the}}$ in a $(-C, C)$ format. We search for such values using a Python script `part3.py` by iterating through values from 1 upwards and calculating the probabilities until we find that a nearly optimal vector within 0.01 of the optimum, which is $\mathbf{v}_{\text{the}} = (-2, 2)$. Increasing the values further brings the probabilities closer to the target.

Q2

3c) With a word embedding space $d = 2$, the training examples derived from the sentences:

```
the dog
the cat
a dog
a cat
```

with window size $k = 1$ are:

$$(x = \text{the}, y = \text{dog})$$
$$(x = \text{dog}, y = \text{the})$$
$$(x = \text{the}, y = \text{cat})$$
$$(x = \text{cat}, y = \text{the})$$
$$(x = \text{a}, y = \text{dog})$$
$$(x = \text{dog}, y = \text{a})$$
$$(x = \text{a}, y = \text{cat})$$
$$(x = \text{cat}, y = \text{a})$$

3c) We see that each center word appears equally often with each context word. Thus, the optimal probabilities for each context word given a

center word are:

$$P(\text{the}|\text{dog}) = P(\text{a}|\text{dog}) = 0.5$$
$$P(\text{the}|\text{cat}) = P(\text{a}|\text{cat}) = 0.5$$
$$P(\text{dog}|\text{the}) = P(\text{cat}|\text{the}) = 0.5$$
$$P(\text{dog}|\text{a}) = P(\text{cat}|\text{a}) = 0.5$$

and all other context words have probability 0. With context vectors:

$$\mathbf{c}_{\text{dog}} = \mathbf{c}_{\text{cat}} = (0, 1)$$
$$\mathbf{c}_{\text{a}} = \mathbf{c}_{\text{the}} = (1, 0)$$

we can find nearly optimal word vectors $\mathbf{v}_w$ for each word w using similar logic as in 3b. However, we also need to ensure that $\mathbf{v}_{\text{dog}}$ and $\mathbf{v}_{\text{cat}}$ this time give equal probabilities, and thus should follow $(C, -C)$ format instead. From our value in 3b, we then obtain the following vectors:

$$\mathbf{v}_{\text{the}} = (-2, 2)$$
$$\mathbf{v}_{\text{a}} = (-2, 2)$$
$$\mathbf{v}_{\text{dog}} = (2, -2)$$
$$\mathbf{v}_{\text{cat}} = (2, -2)$$

Running `part3.py` confirms that these vectors yield probabilities within 0.01 of the optimum.

LLM Usage: All work was done by myself in VSCode with GitHub Copilot integration. The integration "provides code suggestions, explanations, and automated implementations based on natural language prompts and existing code context," and also offers autonomous coding and an in-IDE chat interface that is able to interact with the current codebase. Only the Copilot provided automatic inline suggestions for both LaTex and Python in `.tex` and `.py` files respectively were taken into account / used.