Key Equations for Quiz 3

1. Decoding Algorithms

Decoding is the process of selecting a sequence of tokens from the model's probability distribution.

(a) **Greedy Decoding:** At each step, choose the single most likely token.

$$y_i = \arg\max_y P_\theta(y|y_{<i})$$

Time Complexity: $O(t \cdot |\mathcal{V}|)$, where t is the generated sequence length and $|\mathcal{V}|$ is the vocabulary size.

(b) **Beam Search Score:** Keeps track of the top K (beam size) hypotheses (sequences) by summing their log-probabilities. The goal is to find the sequence with the highest score.

$$\text{score}(y_1, y_2 ... y_t) = \sum_{i=1}^{t} \log P(y_i|y_1, ... y_{i-1})$$

Time Complexity: $O(t \cdot K \cdot |\mathcal{V}|)$, where K is the beam size.

(c) **Sampling with Temperature:** The output distribution can be sharpened (low temperature) or softened (high temperature) by a temperature parameter T. Let h be the vector of logits.

$$p_\theta(x_t = w) = \frac{\exp(h_w/T)}{\sum_{w'} \exp(h_{w'}/T)}$$

2. Speculative Decoding

Speculative decoding speeds up inference by using a small "draft" model to generate candidate tokens and a large "main" model to verify them in parallel.

(a) **Standard Autoregressive Cost:** Generating k tokens after a prompt p tokens requires k sequential forward passes. With L transformer layers, the total cost is:

$$\text{Cost} = O(pL) + O((p+1)L) + ... + O((p+k-1)L) = O(k \cdot pL + k^2 L),$$

where the term $O(k^2 L)$ is the main bottleneck.

(b) **Speculative Decoding Computational Benefit:** This method replaces the k sequential forward passes of the main model (L_{large}) with generation from a **fast** draft model (L_{small}) plus a **single verification pass** from the main model, thus avoiding the $O(k^2 L_{large})$ (assume $L_{small} \ll L_{large}$). The total cost is:

$$Cost = \underbrace{O(kpL_{small} + k^2 L_{small})}_{\text{generation cost}} + \underbrace{O((p+k)L_{large})}_{\text{verification cost}}$$

3. Pre-training & Fine-tuning

(a) **Pre-training Objective:** Models are first trained on a general language modeling task using a large, unsupervised text corpus.

$$P(w_i|w_1, w_2...w_{i-1})$$

(b) **Fine-tuning Objective:** The pre-trained model is then adapted to a specific downstream task using a smaller, supervised dataset.

$$P(y|x_1, x_2...x_{|x|})$$

4. Model-Specific Objectives

(a) **ELMo Objective:** ELMo is trained to maximize the log-likelihood of tokens in both a forward and a backward pass through the text.

$$\sum_{k=1}^{N}(\log p(t_k|t_1, ..., t_{k-1}) + \log p(t_k|t_{k+1}, ..., t_N))$$

(b) Also pay attention to other types of models' pretraining objectives in the slides.

5. Tokenization (Byte-Pair Encoding - BPE style)

Algorithm 1 Byte-Pair Encoding (BPE)

Initialize Vocabulary: Start with a vocabulary $\mathcal{V}$ containing all individual characters present in the training data as base tokens.
$\mathcal{V} \leftarrow$ All characters in the training data (as base tokens)
Iterative Merging (for k steps):
for $i = 1$ to k **do**
    Tokenize the data: Take the longest prefix of known tokens each time to break down words into current vocabulary tokens.
    Count the frequency of adjacent token pairs in the tokenized data.
    Choose the pair $\langle l, r \rangle$ that occurs most frequently.
    Merge the chosen pair and add to the vocabulary as a new token:
    $\mathcal{V} \leftarrow \mathcal{V} \cup \{lr\}$
end for
Return Final Vocabulary: After k steps, return the expanded vocabulary $\mathcal{V}$.

6. Prompt-Based Learning

Instead of fine-tuning, large language models can be guided to perform tasks using prompts. The model fills in a blank (‘[Z]‘) in a template, and the result is mapped to a final prediction.

- Let $f_{fill}(x', z)$ be the template filled with input x' and a potential answer word z from a set of possible answers Z. The model selects the answer word with the highest probability.

$$\hat{z} = \arg\max_{z \in Z} P(f_{fill}(x', z); M_\theta)$$