CSE 493 G1/ 599 G1
Deep Learning
Autumn 2024 Quiz 4

SOLUTIONS

November 22, 2024

Full Name: _____

UW Net ID: _____

Question		Score
True/False	(4 pts)	
Multiple Choice	(8 pts)	
Short Answer	(8 pts)	
Total	(20 pts)	

Welcome to the CSE 493 G1 Quiz 4!

- The exam is 30 min and is **double-sided**.
- No electronic devices are allowed.

I agree to uphold the University of Washington Student Conduct Code during this exam.

Signature: _____ Date: _____

Good luck!

Mean	15.3
Median	16.0
Stdev	3.9

This page is left blank for scratch work only. DO NOT write your answers here.

1 True / False (4 points) - Recommended 4 Minutes

Fill in the circle next to True or False, or fill in neither. Fill it in completely like this: ●. No explanations are required.

Scoring: Correct answer is worth 1 points.

1.1 Decoder only models use causal masked attention (i.e. masking out future tokens) for training and inference

◯ True

◯ False

SOLUTION:
True. Decoder-only models do indeed use causal masked attention during both training and inference to prevent looking at future tokens. This is necessary for their autoregressive nature, where each token can only attend to previous tokens.

1.2 You train model A on pre-text task A and model B on pre-text task B. Model A gets 90% accuracy on task A and Model B gets 20% on task B. You then transfer both models to task C and fine tune. This means that Model A will perform better on task C than Model B.

◯ True

◯ False

SOLUTION:
False. Pre-training performance (90% vs 20%) does not predict downstream performance after fine-tuning. Transfer success depends on task similarity and fine-tuning approach, not initial accuracy.

1.3 Chain-of-thought prompting is a type of in-context learning

◯ True

◯ False

SOLUTION:
True. Chain-of-thought prompting demonstrates reasoning patterns within the prompt that the model can apply to new examples without parameter updates, making it a form of in-context learning.

1.4 You want to classify 10 image between 4 classes using CLIP. You have unlimited memory. You want to use a single template for each class of the format "A photo of a {class name}". This would take minimum 50 runs of the image or text encoder (1 for the image and then 4 for the text classes) $\times$ 10.

◯ True

◯ False

SOLUTION:
False. Need only 14 encoder runs total: 10 image encodings (1 per image) + 4 text encodings (1 per class). Text encodings can be reused across images.

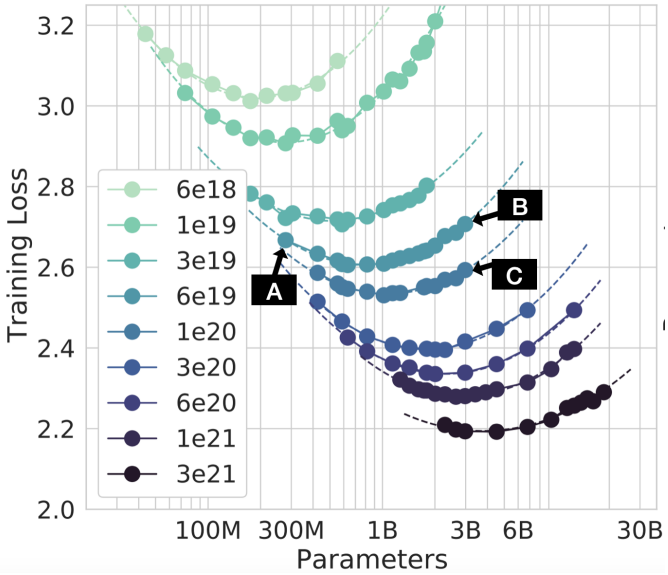
2 Multiple Choices (8 points) - Recommended 8 Minutes

Fill in the circle next to the letter(s) of your choice (like this: ●). No explanations are required. **Choose ALL options that apply.**

Each question is worth 4 points and the answer may contain one or more options. Selecting all of the correct options and none of the incorrect options will get full credits. For questions with multiple correct options, each incorrect or missing selection gets a 2-point deduction (up to 4 points).

2.1 The plot below is part of a study that explores different combinations of model size and training data while keeping the total compute budget fixed. The compute budget is the total number of operations (or FLOPs) used during training of the model. The compute budget can be affected by changing the number of parameters of the model (which changes the FLOPs per iteration) or changing the amount of training data (which changes the number of iterations).

Each individual dot on the plot represents a separate, complete training run of a neural network. That means that every point is a distinct model that was trained from scratch and shows the final training loss achieved by that specific model. All of the models that are the same color/on the same dotted line have the same compute budget, which is given in the legend.



- ◯ A: Point A and Point B correspond to models with approximately the same number of parameters.
- ◯ B: Point B and point C correspond to models with approximately the same number of parameters.
- ◯ C: If you have the compute budget to train for 6e18 FLOPs, you should select a model with slightly less than 300M parameters to minimize train loss
- ◯ D: If you have the compute budget to train for 1e21 FLOPs, you should select a model with slightly over 6B parameters to minimize train loss
- ◯ E: This graph suggests that training for longer but keeping the model size constant leads to lower training loss
- ◯ F: Given a set number of operations (FLOPs) that you can use during your training run, this graph suggests that you should pick the largest model to minimize train loss.

SOLUTION:

- ◯ A: False. Looking at points A and B on the graph, they have different x-coordinates on the Parameters axis.
- ◯ B: True. Points B and C appear to be vertically aligned on the graph, indicating they correspond to models with approximately the same number of parameters (around 3B).
- ◯ C: True. Looking at the 6e18 FLOP line, the minimum training loss occurs below 300M.
- ◯ D: False. Following the 1e21 FLOP line, the minimum training loss occurs at less than 3B parameters.
- ◯ E: True. Following any single vertical line (constant model size) shows decreasing training loss as FLOPs increase, meaning longer training leads to lower loss.
- ◯ F: False. For a fixed FLOP budget (following a single dotted line), the a middle size model often achieves lower loss than larger models that can't train as long. The optimal is a balance, not simply picking the largest model.

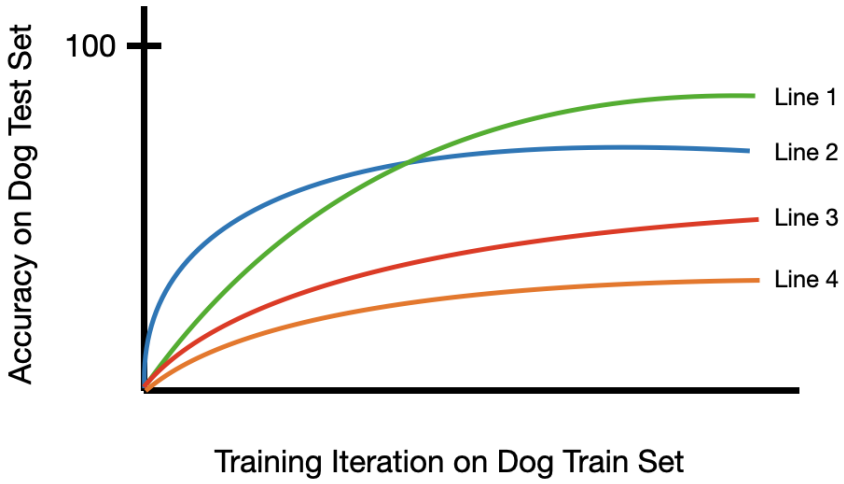
2.2 You take 4 identical copies of the same vision model and pre-train each of them with one of the following pre-text tasks/datasets:

(a) CLIP Self-Supervised objective on scraped image/text pairs – 1 billion images total

(b) Supervised classification on Cat breed train set – 1 million images total

(c) Supervised classification on MNIST (handwritten digit) train set – 1 million images total

(d) Supervised classification on Animals pre-training set (which includes different dog breeds) – 1 million images total



You then remove the final layer on all models, replace them with a new FC layer and fine-tune this last layer for all 4 models on a supervised classification task for the Dog breed train set, evaluating on each iteration on the dog breed test set. Below are the results. Map each line with its correct pre-training regime:

(a) Line 1 is pre-training regime:

☐ A ☐ B ☐ C ☐ D

(b) Line 2 is pre-training regime:

☐ A ☐ B ☐ C ☐ D

(c) Line 3 is pre-training regime:

☐ A ☐ B ☐ C ☐ D

(d) Line 4 is pre-training regime:

☐ A ☐ B ☐ C ☐ D

SOLUTION:
Line 1 is CLIP (A) due to its highest final performance despite slower initial learning, leveraging rich representations from 1B diverse images. Line 2 is Animal pre-training (D), showing strong initial gains from domain relevance but slightly lower final performance due to significantly less data. Line 3 matches Cat breeds (B) with moderate performance from a related fine-grained classification task. Line 4 corresponds to MNIST (C), showing poorest transfer due to the large domain gap between digits and dog breeds.

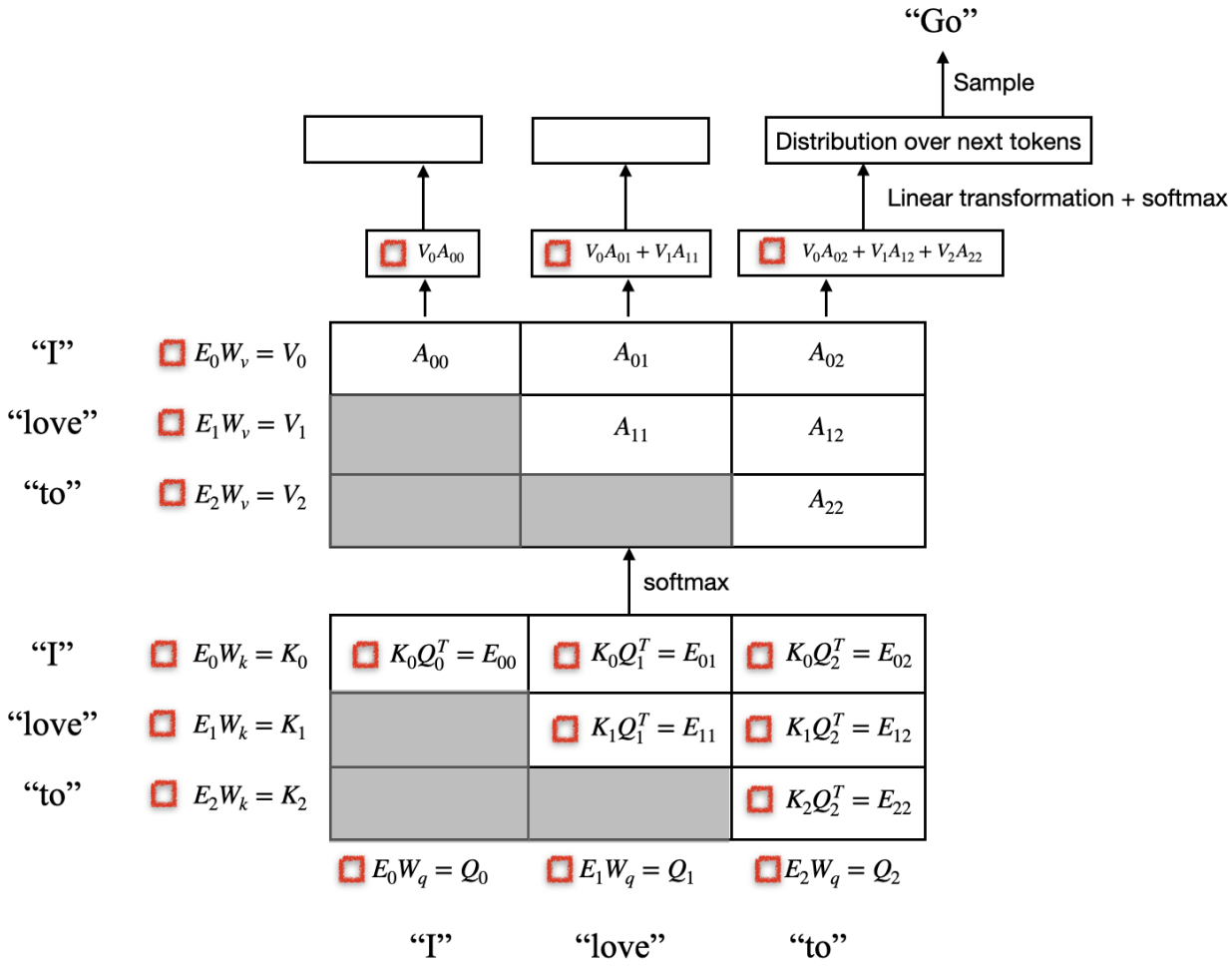
3 Short Answers (8 points) - Recommended 8 Minutes

Please make sure to write your answer only in the provided space.

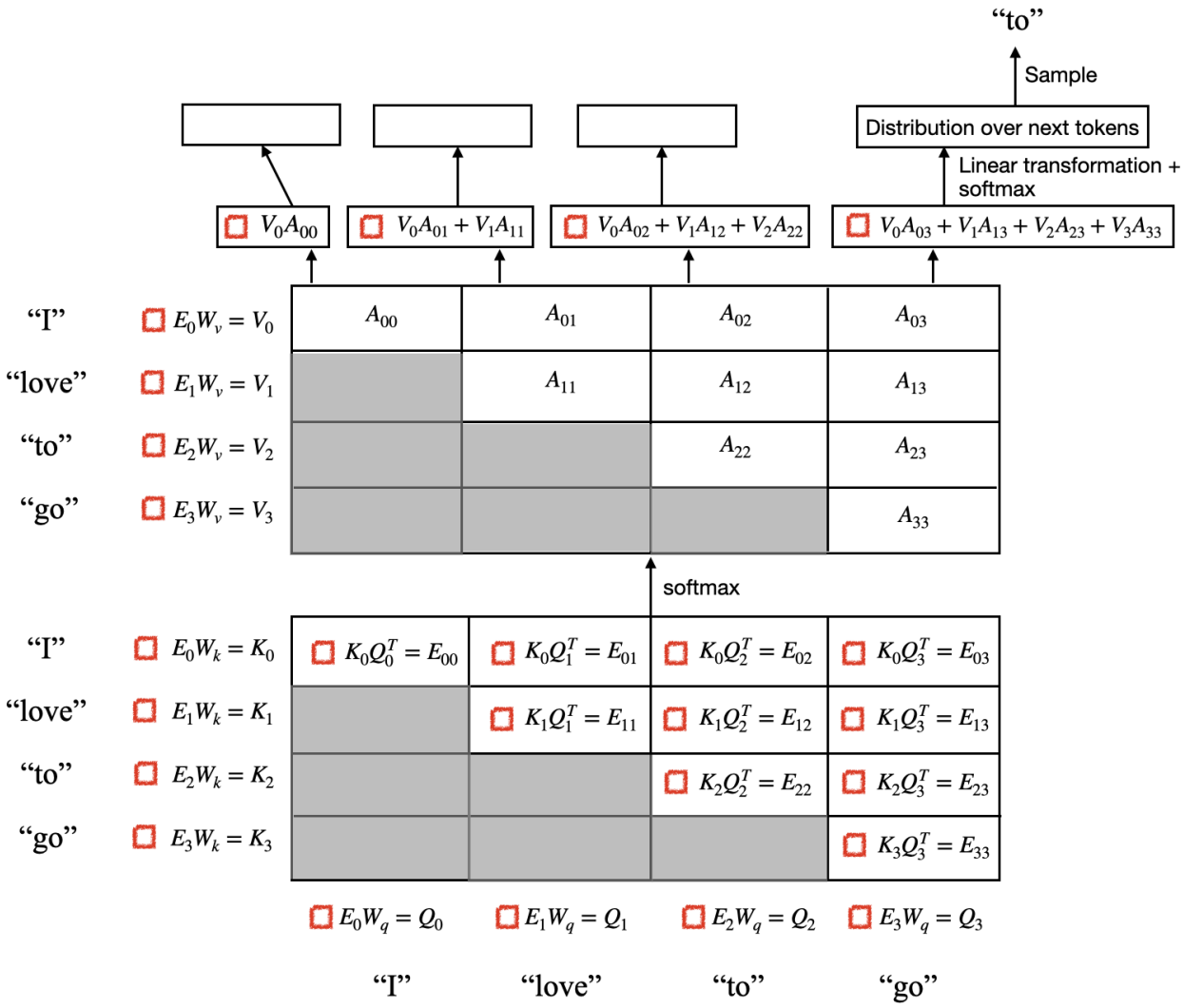
3.1 Making decoder models more efficient

You have a trained decoder-only LM and you are using it to generate some text, word by word. For this question, each token is a word.

1. (2 pnts) You are generating the next word using your transformer. Next to each operation in the transformer, there is a checkbox. Fill in each checkbox that corresponds to an operation that is necessary in order to output the next token "go" (assume you have nothing cached).



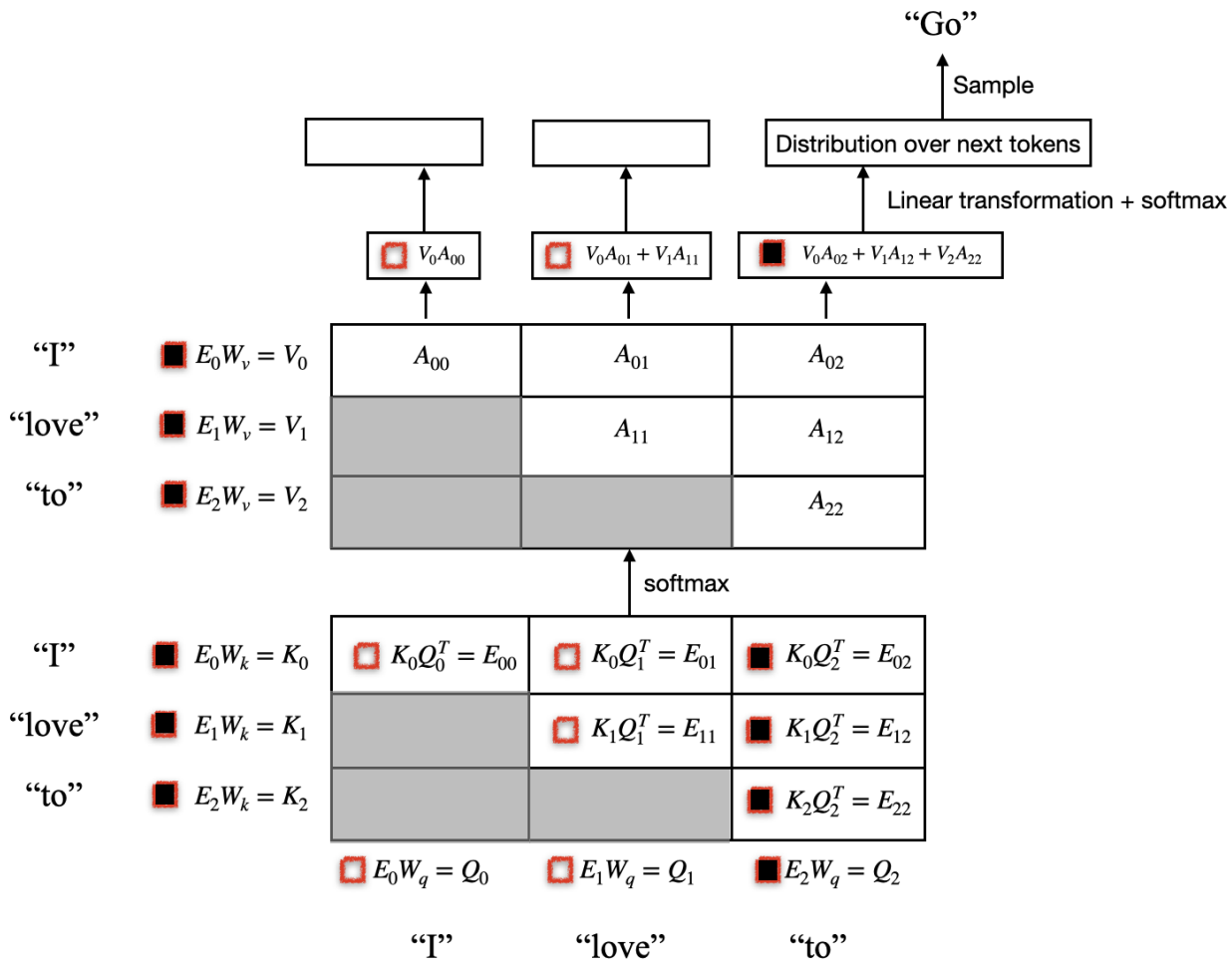
2. (2 pnts) Now you have generated a new word! You add "go" to your generated sequence (so it is now length 4) and feed this back in to generate your next token. Fill in each checkbox that corresponds to an operation that is necessary in order to output the next token "to" (assume you have nothing cached).



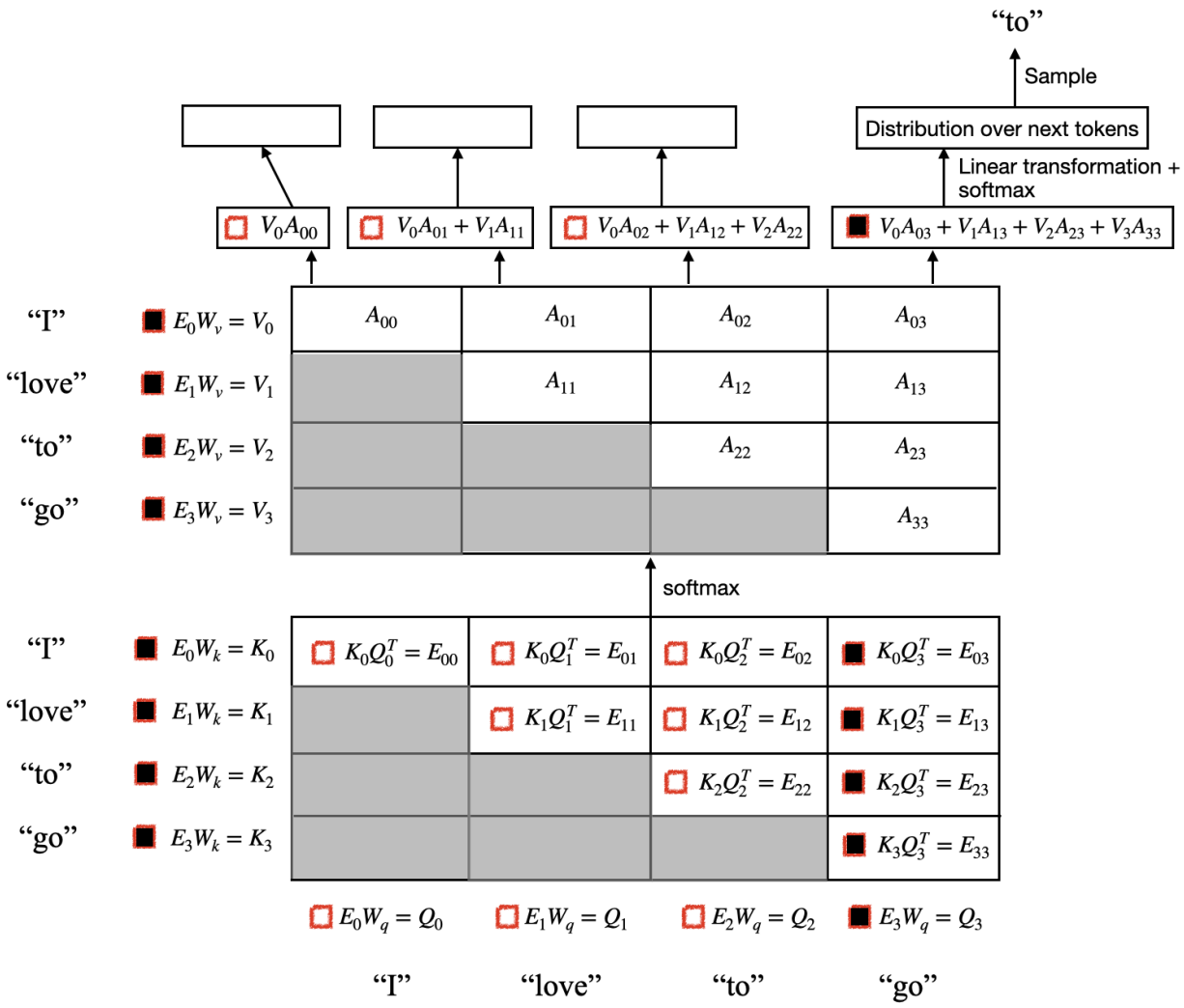
3. (4 pnts) Have you repeated any computation in generating these two tokens? If so, what? Describe a method to decrease your total computation during generation by saving things you have previously calculated to use in future steps. Feel free to answer on the back of this page.

SOLUTION:

1.


"Go"
Sample
Distribution over next tokens
Linear transformation + softmax
V_0A_{00}
$V_0A_{01} + V_1A_{11}$
$V_0A_{02} + V_1A_{12} + V_2A_{22}$
"I"
"love"
"to"
$E_0W_v = V_0$
$E_1W_v = V_1$
$E_2W_v = V_2$
A_{00}
A_{01}
A_{02}
A_{11}
A_{12}
A_{22}
softmax
"I"
"love"
"to"
$E_0W_k = K_0$
$E_1W_k = K_1$
$E_2W_k = K_2$
$K_0Q_0^T = E_{00}$
$K_0Q_1^T = E_{01}$
$K_0Q_2^T = E_{02}$
$K_1Q_1^T = E_{11}$
$K_1Q_2^T = E_{12}$
$K_2Q_2^T = E_{22}$
$E_0W_q = Q_0$
$E_1W_q = Q_1$
$E_2W_q = Q_2$
"I"
"love"
"to"

2.



3. Store previous k and v vectors