AVISO BUÉ IMPORTANTE, QUEM NÃO LER MORRE E FICA MORTO

Este teste foi feito automaticamente com AI generativo. Dei como contexto as perguntas do primeiro exame desta cadeira e disse para ele tentar replicar o nível de dificuldade. Nem sequer dei proof-read deste documento, eis a quantidade de responsabilidade que tomo pela qualidade das perguntas: 0.
Acho que até há alguns erros de formatação e tal.
Depois meto as soluções. Mas só se tiver tempo, vá, não garanto que consiga colocar as soluções antes do exame.

Deep Learning - Exam Part 2 (Generated)

Topic: Word Embeddings (T4)

1. Word2Vec Architecture and Calculation

Consider a **CBOW (Continuous Bag of Words)** model trained to predict a target word given context words.

- **Vocabulary:** {"AI", "is", "very", "cool"} encoded as indices $0, 1, 2, 3$.
- **Embedding Dimension:** $N = 2$.
- **Input Matrix** W_{in} (shape 4×2):

$$W_{in} = \begin{bmatrix} 0.5 & 0.2 \\ -0.1 & 0.8 \\ 0.0 & -0.5 \\ 0.9 & 0.1 \end{bmatrix}$$

- **Output Matrix** W_{out} (shape 2×4):

$$W_{out} = \begin{bmatrix} 1.0 & 0.0 & -0.5 & 0.5 \\ 0.0 & 1.0 & 0.5 & -1.0 \end{bmatrix}$$

Task: We want to predict the target word **"is"** (Index 1) given the context words **"AI"** (Index 0) and **"very"** (Index 2).

a) Draw the specific architecture for this training instance, showing the inputs, the hidden layer projection, and the output layer.

b) Calculate the hidden layer vector h. (Explain how the context vectors are combined).

c) Calculate the raw logits vector z for all words in the vocabulary.

d) Apply the Softmax function to calculate the probability $P(\text{"is"}|\text{context})$.

2. Static vs. Dynamic Embeddings

Explain the fundamental limitation of static embeddings (like Word2Vec and GloVe) that necessitates the use of dynamic (contextualized) embeddings. Provide a concrete example using the word "bank" to illustrate your answer.

3. FastText vs. Word2Vec

In the context of morphologically rich languages (like Finnish or Portuguese), why does FastText typically outperform standard Word2Vec? Explain how FastText handles Out-Of-Vocabulary (OOV) words during inference.

Topic: Sequence to Sequence RNNs (T5)

4. Manual Seq2Seq Calculation (The "Reversal Task")
We are training a simplified Seq2Seq model to reverse a string of tokens.

- **Vocabulary:** {<BOS>, <EOS>, A, B, C} mapped to indices $0, 1, 2, 3, 4$.
- **Embeddings:** Provided directly as size 2 vectors:
  - $\text{emb}(< BOS >) = [0, 0]$
  - $\text{emb}(< EOS >) = [0, 0]$
  - $\text{emb}(A) = [1, 0]$
  - $\text{emb}(B) = [0, 1]$
  - $\text{emb}(C) = [1, 1]$
- **Encoder Update Rule:** $h_t = h_{t-1} + x_t$ (Simple addition, $h_0 = [0, 0]$).
- **Decoder Update Rule:** $s_t = s_{t-1} + e_t$ (Simple addition, $s_0 =$ Encoder Context c).
- **Output Weights W_o:**

$$W_o = \begin{bmatrix} 0 & 0 & 1 & 0 & 0.5 \\ 0 & 0 & 0 & 1 & 0.5 \end{bmatrix}^T = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 1 \\ 0.5 & 0.5 \end{bmatrix}$$

(Note: Rows correspond to vocabulary words 0..4)

Input Sequence: "A B" (followed implicitly by EOS, but we only encode A and B). **Target Sequence:** "B A".

a) **Encoder Forward Pass:** Calculate the hidden states h_1 and h_2 for the input "A B". What is the final context vector c?

b) **Decoder Step 1:** We are at the first decoding step $t = 1$.
- The input to the decoder is the $<$ BOS $>$ token.
- Calculate the decoder hidden state s_1.
- Calculate the logits $y^{(1)} = W_o \cdot s_1$.
- Which word has the highest probability? Does it match the target "B"?

c) **Decoder Step 2 (Teacher Forcing):** We are at the second decoding step $t = 2$.
- Using **Teacher Forcing**, what is the input embedding e_2 provided to the decoder?
- Calculate the new decoder state s_2.
- Calculate the logits $y^{(2)}$.

5. Exposure Bias and Sampling

a) Explain the concept of **Exposure Bias** in Seq2Seq models. Why does it occur?

b) Describe the **Scheduled Sampling** strategy. How does the sampling probability ϵ_t change during training, and what effect does this have on the model's learning process?

6. Attention Mechanism (Bahdanau)
In the "Jointly Learning to Align and Translate" paper, the fixed-length context vector bottleneck is removed.

a) Write the mathematical formula for the dynamic context vector c_t at decoding step t. (Define all terms: $\alpha_{t,i}$ and h_i).

b) Explain intuitively what the attention weights $\alpha_{t,i}$ represent in the context of Machine Translation.

Deep Learning - Exam Part 3 (Generated)

Topic: Transformers and LLMs (T6)

1. Manual Calculation of Self-Attention

Consider a single-head Self-Attention mechanism. We have an input sequence of two tokens: $X = [x_1, x_2]$. The input embedding dimension is $d_{model} = 2$. The values of the input vectors are:

$$x_1 = [1, 0], \quad x_2 = [0, 1]$$

The weight matrices for Query (W^Q), Key (W^K), and Value (W^V) are all 2×2 matrices given by:

$$W^Q = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix}, \quad W^K = \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}, \quad W^V = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$$

Task: Calculate the self-attention output vector z_1 for the first token (x_1).

a) Calculate the Query vector q_1 for the first token.

b) Calculate the Key vectors k_1 and k_2 for both tokens.

c) Calculate the Value vectors v_1 and v_2 for both tokens.

d) Compute the raw attention scores (dot products) for x_1 against all keys: $score_{1,1} = q_1 \cdot k_1$ and $score_{1,2} = q_1 \cdot k_2$.

e) Apply the Softmax function to these scores to get the attention weights $\alpha_{1,1}$ and $\alpha_{1,2}$. *(Note: For calculation simplicity, you may leave terms in the form of e^x / sum, or approximate if values are obvious).*

f) Compute the final weighted sum output $z_1 = \alpha_{1,1} v_1 + \alpha_{1,2} v_2$.

2. Positional Encodings

Transformers use the formula $Attention(Q, K, V) = \text{softmax}(\frac{QK^T}{\sqrt{d_k}})V$.

a) Why does the Transformer architecture typically require **Positional Encodings** added to the input embeddings, whereas Recurrent Neural Networks (RNNs) do not?

b) If we removed the positional encodings, how would the self-attention mechanism treat the sentences "Alice hit Bob" and "Bob hit Alice"?

3. BERT vs. GPT Architecture

a) Explain the structural difference between the **BERT** (Encoder-only) and **GPT** (Decoder-only) architectures regarding the attention mechanism mask.

b) How do their pre-training objectives differ? Specifically, contrast **Masked Language Modeling (MLM)** with **Causal Language Modeling (CLM)**.

4. Large Language Models (LLMs) & RLHF

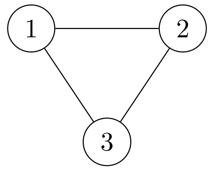
Modern LLMs like ChatGPT use a technique called **RLHF** (Reinforcement Learning from Human Feedback).

a) Briefly describe the role of the **Reward Model** in this process. What does it take as input, and what does it output?

b) Explain the concept of **In-Context Learning** (e.g., Few-Shot Prompting). Does this process update the weights of the model?

Topic: Graph Neural Networks (T7)

5. Manual Calculation of GCN Message Passing
Consider a simple graph with 3 nodes.



(It is a fully connected triangle: 1-2, 2-3, 3-1).
The initial node features $h^{(0)}$ (dimension $d = 2$) are:

$$h_1^{(0)} = [1, 0], \quad h_2^{(0)} = [0, 1], \quad h_3^{(0)} = [1, 1]$$

We use a simplified Graph Convolutional Network (GCN) update rule:

$$h_u^{(k+1)} = \sigma \left(W \cdot \sum_{v \in N(u) \cup \{u\}} \frac{1}{c_{uv}} h_v^{(k)} \right)$$

Where:

- The aggregation is a **Mean** (Average) over the node itself and its neighbors (c_{uv} is the degree of node u including self-loop).
- The weight matrix W is Identity $I = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$.
- The activation σ is ReLU.

Task: Calculate the new feature vector $h_1^{(1)}$ for **Node 1** after one layer of message passing.

a) Identify the neighbors of Node 1 (including itself).

b) Perform the aggregation step (Sum neighbors and normalize by the count).

c) Apply the linear transformation (W) and activation (σ).

6. Permutation Invariance

a) A standard Convolutional Neural Network (CNN) is designed for grid-structured data (images). Why does a standard CNN fail when applied directly to a graph represented by an adjacency matrix?

b) Explain what it means for a GNN to be **Permutation Invariant**. Why is this property critical for graph learning?

7. Spectral vs. Spatial GNNs
The slide deck mentions the GCN formulation by Kipf & Welling:

$$H^{(l+1)} = \sigma(\tilde{D}^{-1/2} \tilde{A} \tilde{D}^{-1/2} H^{(l)} W^{(l)})$$

a) What is the purpose of $\tilde{A}$ (where $\tilde{A} = A + I$)? Why do we add the Identity matrix I to the adjacency matrix A?

b) What is the role of the $\tilde{D}^{-1/2} \ldots \tilde{D}^{-1/2}$ term? (Hint: Think about node degrees).

Deep Learning - Exam Part 4 (Generated)

Topic: Reinforcement Learning (T8)

1. Markov Decision Processes (MDP) & Bellman Calculation

Consider an agent in a simple state s_t. The agent can choose between two actions: **Left** (a_L) or **Right** (a_R). The discount factor is $\gamma = 0.9$.

- **Action a_L:** Deterministic. The agent moves to state s_L and receives reward $R = +1$.
- **Action a_R:** Stochastic.
  - With probability 0.8, move to state s_{win} (Reward $R = +10$).
  - With probability 0.2, move to state s_{fail} (Reward $R = -5$).

Assume we already know the Value of the next states from previous iterations:

$$V(s_L) = 5, \quad V(s_{win}) = 20, \quad V(s_{fail}) = 0$$

Task:

a) Calculate the **Q-value** for taking action Left: $Q(s_t, a_L)$.

b) Calculate the **Q-value** for taking action Right: $Q(s_t, a_R)$.

c) Based on these Q-values, what is the value of the current state $V(s_t)$ if the agent acts greedily?

2. Manual Q-Learning Iteration

We are training an agent using the **Q-Learning** algorithm (Off-policy TD control). We are at time step t.

- **Current State:** $S_t = A$
- **Action Taken:** $A_t = \text{Run}$
- **Reward Received:** $R_{t+1} = -2$
- **Next State:** $S_{t+1} = B$
- **Learning Rate:** $\alpha = 0.1$
- **Discount Factor:** $\gamma = 0.9$

Your current Q-Table estimates are:

State	Action: Walk	Action: Run
A	4.0	**2.5**
B	**5.0**	1.0

Task: Perform one Q-Learning update step to calculate the new value $Q_{new}(A, \text{Run})$. *(Show the formula used and the substitution of values).*

3. Exploration vs. Exploitation

You are using an ϵ-greedy policy with $\epsilon = 0.2$ in a state with 4 possible actions. The Q-values for these actions are: $[10, 12, 15, 9]$.

a) What is the probability that the agent selects the action with value 15 (the greedy action)?

b) What is the probability that the agent selects the action with value 9? *(Assume the random choice is uniform over all available actions).*

4. Deep Q-Networks (DQN)

Standard Q-Learning uses a table. DQN uses a Neural Network to approximate $Q(s, a; \theta)$. However, training a standard Neural Network with RL data is unstable. DQN introduces two key mechanisms to fix this.

a) **Experience Replay:**

- Explain how Experience Replay works.
- Why is it necessary? (Hint: Discuss the correlation of sequential data).

b) **Target Networks:**

- The loss function for DQN is often written as:

$$L(\theta) = (y_i - Q(s, a; \theta))^2$$

where the target is $y_i = R + \gamma \max_{a'} Q(s', a'; \theta^-)$.
- Why do we use a separate network parameter set θ^- (the target network) to calculate y_i, rather than the current parameters θ? What problem ("Moving Target") does this solve?

5. Credit Assignment Problem

In the context of Reinforcement Learning, explain the **Credit Assignment Problem**. Why does a sparse reward signal (e.g., only getting a reward of +1 after winning a Chess game, and 0 otherwise) make learning difficult?

Deep Learning - Exam Part 5 (Generated)

Topic: SOTA and Trends (T9)

1. Vision Transformers (ViT) - Manual Calculation

In a Vision Transformer, the input image is split into fixed-size patches, which are linearly embedded. Consider the following specifications:

- **Input Image Size:** $H \times W = 48 \times 48$ pixels.
- **Channels:** $C = 3$ (RGB).
- **Patch Size:** $P \times P = 16 \times 16$ pixels.
- **Transformer Hidden Dimension (Embedding Size):** $D = 128$.

Task:

a) Calculate the total number of patches N extracted from the image.

b) Before the linear projection (embedding layer), what is the dimensionality of a single flattened patch vector?

c) The "Class Token" (CLS) is added to the sequence of patch embeddings. What is the final shape of the input tensor passed to the Transformer Encoder?

2. LoRA (Low-Rank Adaptation) - Parameter Efficiency

You want to fine-tune a pre-trained dense layer weight matrix W_0 of dimensions $d_{in} \times d_{out} = 1000 \times 1000$. Instead of standard fine-tuning (updating all weights), you use **LoRA**. LoRA approximates the weight update ΔW using two low-rank matrices A and B, such that $W = W_0 + BA$. We choose a rank $r = 8$.

- Matrix B has dimensions 1000×8.
- Matrix A has dimensions 8×1000.

Task:

a) Calculate the number of trainable parameters required for **Standard Fine-Tuning** of this layer.

b) Calculate the number of trainable parameters required for **LoRA Fine-Tuning** (parameters in A and B).

c) What is the reduction factor? (Ratio of Standard params to LoRA params).

3. Diffusion Models
Diffusion models operate using a Forward Process and a Reverse Process.

a) Describe what happens to an image x_0 during the **Forward Process** as time t goes from 0 to T. What is the final state x_T?

b) In the **Reverse Process**, a neural network (typically a U-Net) is trained. What exactly does this network predict at each step? (Does it predict the fully denoised image x_0 directly, or something else?)

4. CLIP (Contrastive Language-Image Pre-training)
CLIP is trained on pairs of (Image, Text). Suppose we have a batch of N image-text pairs. The model produces N image embeddings $I_1 \dots I_N$ and N text embeddings $T_1 \dots T_N$. We compute the similarity matrix of size $N \times N$ containing all dot products $I_i \cdot T_j$.

a) Which values in this $N \times N$ matrix does the Contrastive Loss try to **maximize**?

b) Which values does it try to **minimize**?

c) Why does this training objective allow CLIP to perform "Zero-Shot" classification on unseen datasets?

5. RAG (Retrieval Augmented Generation)
Large Language Models often suffer from "hallucinations" or outdated knowledge.

a) Explain how **RAG** solves this problem without retraining the model.

b) In a RAG pipeline, why do we need a Vector Database? What specific operation is performed there?

Deep Learning - Exam Part 6 (Theory & Concepts)

This section covers theoretical concepts, definitions, and details from the slide decks that were not covered in the calculation sections.

Topic: Data Modalities & Embeddings (T4)

1. Data Representation

- Contrast the representation of **Audio** data versus **Image** data before they are fed into a Deep Learning model.
- Why are raw waveforms often transformed into Spectrograms rather than processing the raw amplitude sequence directly?

2. The Distributional Hypothesis

- The slide deck quotes J.R. Firth (1957): *"You shall know a word by the company it keeps."*
- Explain how this specific quote forms the theoretical basis for unsupervised training of Word2Vec (Skip-gram and CBOW).

3. One-Hot Encoding Limitations

- Beyond the lack of semantic meaning (orthogonality), explain the **"Curse of Dimensionality"** or "Vocabulary Explosion" problem associated with One-Hot Encoding.
- Why is this problem particularly severe in morphologically rich languages?

4. GloVe Objective Function

- The GloVe loss function is given by:

$$J = \sum_{i,j} f(X_{ij})(w_i^T \tilde{w}_j + b_i + \tilde{b}_j - \log X_{ij})^2$$

- What is the purpose of the weighting function $f(X_{ij})$? Specifically, what does it prevent regarding rare and frequent co-occurrences?

5. Evaluation Methods

- Distinguish between **Intrinsic** and **Extrinsic** evaluation of embeddings.
- Give one concrete example of an Intrinsic task (e.g., Analogy).
- Why might a model score high on Intrinsic tasks but fail on Extrinsic ones (e.g., Sentiment Analysis)?

Topic: Seq2Seq & RNNs (T5)

6. Decoding Strategies: Beam Search

- Compare **Greedy Decoding** with **Beam Search**.
- If Beam Width $k = 1$, is Beam Search equivalent to Greedy Decoding? Explain.
- Explain the trade-off: Why does Beam Search trade off inference speed for generation quality?

7. Reversing the Source Sentence

- In the "Sequence to Sequence Learning with Neural Networks" paper (Sutskever et al.), the authors found that reversing the order of words in the *source* sentence (but not the target) improved performance.
- Explain the theoretical reason given for this improvement. (Hint: It relates to "short term dependencies" and optimization).

8. Character-Level Encoding

- What is the primary advantage of **Character-Level Encoding** (treating each character as a token) compared to Word-Level encoding?
- What is the major downside regarding the resulting sequence length and computational cost?

Topic: Transformers & LLMs (T6)

9. Multi-Head Attention Mechanism

- In Multi-Head Attention, we split the embedding dimension d_{model} into h heads.
- After computing attention for each head independently, how are the results combined back into a single vector? (Describe the Concatenation and Linear W^O operation).

10. Feed-Forward Networks in Transformers

- The Feed-Forward Network (FFN) in a Transformer layer typically expands the dimension (e.g., from $d_{model} = 512$ to $d_{ff} = 2048$) and then projects it back.
- What activation function is typically used between these two linear layers (e.g., in the original paper or BERT)?

11. Layer Normalization

- Transformers use **Layer Normalization** rather than Batch Normalization.
- Explain the difference between the two. Why is Layer Norm generally preferred for variable-length sequence data in NLP?

Topic: Graph Neural Networks (T7)

12. Types of Graph Tasks

- Provide one concrete example for each of the following learning tasks:
  - **Node Classification**
  - **Link Prediction**
  - **Graph Classification**

13. Adjacency Matrix & Self-Loops

- In the GCN update rule, we use $\tilde{A} = A + I$.
- If we strictly used A (without adding the Identity matrix I), what would happen to the node's *own* features $h_u^{(l)}$ during the update step?

14. Oversmoothing

- What is the **Oversmoothing** problem in Deep GCNs?
- What happens to the node representations across the graph if we stack too many GCN layers?

Topic: Reinforcement Learning (T8)

15. On-Policy vs. Off-Policy

- Q-Learning is described as an **Off-Policy** algorithm, while SARSA is **On-Policy**.
- What does "Off-Policy" mean?
- Contrast the update equation of Q-Learning (using $\max_{a'} Q(s', a')$) with SARSA (using $Q(s', a')$ where a' is the actual action taken).

16. The Markov Property

- Define the **Markov Property** in the context of MDPs.
- Why is it essential for the state representation S_t to effectively capture the history of the environment?

17. Reward Hypothesis

- Explain the **Reward Hypothesis**. Complete the statement: "All goals can be described by the maximization of..."

Topic: SOTA & Trends (T9)

18. Zero-Shot vs. Few-Shot Prompting

- In the context of Large Language Models:
  - What is **Zero-Shot** prompting?
  - What is **Few-Shot** prompting (In-Context Learning)?
  - Does Few-Shot prompting update the model's weights?

19. Audio Models (Whisper/Wav2Vec)

- Briefly explain the core objective of **Wav2Vec 2.0**.
- How does it learn representations from raw audio without labeled text (Self-Supervised Learning)?

20. CLIP's Training Data

- CLIP is trained on (Image, Text) pairs using a **Contrastive** loss.
- Unlike traditional supervised learning (Image, Label), why does this approach allow CLIP to recognize categories or objects it never explicitly saw as "classes" during training?