Deep Learning

Instructor: Dr. Davoodabadi Farahani
Head TA: Ali Momen

Semester: Fall 2025

Homework 3:

Recurrent Neural Networks Language Modeling

Designed By:

Abtin Badiee	TA2	TA3
@Abtin_003	@Haghighat_Mohammad	@Mobinbarfi

Deadline: 28 Aban

Solution Release & Presentation Day: 6-7 Azar

Preface

In this homework, we aim to develop a fundamental understanding of Recurrent Neural Networks (RNNs) and their advanced variant, Long Short-Term Memory (LSTM) networks. Unlike feedforward networks, RNNs are designed to capture sequential dependencies in data, making them ideal for tasks like language modeling and time series prediction. Through theoretical study and practical implementation, we explore how LSTMs overcome issues like vanishing gradients to retain long-term information, enabling more effective learning from sequential inputs.

Notes and Honor Code

This homework is part of the Deep Learning course at IUST offered in Fall 2025 by Dr.Davoodabadi. Please read all instructions carefully before starting.

- Collaboration is encouraged, but each student must submit their own work.
- Submitting other students' work or copying solutions from another student would result in a 0 score on the assignments
- Typesetting in LaTeX is strongly recommended.
- Clearly mention collaborators, if any.

If you have any further questions regarding the submission policies,course policies e.t.c. feel free to contact the Head Teaching Assistant of the course, Ali Momen, via Telegram.

Submission

The deadline for this homework is "DEADLINE DATE".
Please submit your work by following the instructions below:

* Place your solution alongside the Jupyter notebook(s). Your written solution must be a single PDF file named **HW3_Solution.pdf**.

* Zip all the files together with the following naming format:
  DL_HW3_[StudentNumber]_[FullName].zip.
  Replace `[FullName]` and `[StudentNumber]` with your full name and student number, respectively. Your `[FullName]` must be in CamelCase with no spaces.

* Submit the zip file through Quera in the appropriate section.

1 Theoretical Problems

1.1 Recurrent Neural Networks (RNNs) & LSTMs

1.1.1 Vanishing and Exploding Gradients

1. Provide a clear and concise explanation of the **vanishing gradient** and **exploding gradient** problems as they manifest in Recurrent Neural Networks. In your explanation, detail why these issues are significantly more severe in RNNs compared to standard feed-forward neural networks.

2. Explain how the core architectural principle of a vanilla RNN—the recursive application of a shared weight matrix across time steps—is the direct mathematical cause of the vanishing and exploding gradient problems. Using the chain rule for derivatives, provide a mathematical derivation for the gradient of the loss function $L^{(T)}$ at the final time step T with respect to the hidden state $h^{(t)}$ at a much earlier time step $t \ll T$. Show that this gradient can be expressed as a product of Jacobian matrices: Using this formulation, explain how the magnitude of the recurrent weight matrix W_{hh} within the Jacobian $\frac{\partial h^{(k)}}{\partial h^{(k-1)}}$ leads to either exponential decay or growth of the backpropagated error signal.

3. Identify and briefly describe two distinct techniques used to mitigate gradient instability in RNNs. For each technique, specify whether it primarily addresses the vanishing or the exploding gradient problem and explain its mechanism of action.

1.1.2 Long Short-Term Memory (LSTM) Architecture

1. Draw a detailed diagram of a single Long Short-Term Memory (LSTM) cell. Your diagram must clearly label the following components at time step t:

   - Input vector ($x^{(t)}$)
   - Previous hidden state ($h^{(t-1)}$)
   - Previous cell state ($c^{(t-1)}$)
   - Forget gate ($f^{(t)}$)
   - Input gate ($i^{(t)}$)
   - Output gate ($o^{(t)}$)
   - Candidate cell state ($\tilde{c}^{(t)}$)
   - Current cell state ($c^{(t)}$)
   - Current hidden state ($h^{(t)}$)

2. Provide the complete set of mathematical equations that define the forward pass of an LSTM cell. Your answer should include the equations for the forget gate, input gate, output gate, the candidate cell state, the final cell state update, and the final hidden state. Following the equations, explain the specific roles of the sigmoid (σ) and hyperbolic tangent (tanh) activation functions within the LSTM architecture. Why are gates controlled by sigmoid functions, while the cell state update and hidden state output are modulated by tanh functions?

3. Explain precisely how the architectural design of an LSTM, particularly its use of a separate cell state with an additive update mechanism, helps to mitigate the vanishing gradient problem. Contrast the gradient flow through the LSTM's cell state with the gradient flow through the hidden state of a vanilla RNN.

1.1.3 Gated Recurrent Unit (GRU) vs. LSTM

1. Describe the architecture of a Gated Recurrent Unit (GRU). Identify its primary gating mechanisms and explain how it combines the roles of the cell state and hidden state found in an LSTM into a single state vector.

2. Provide a comprehensive comparison between the LSTM and GRU architectures. Your analysis should be structured to address the following dimensions:

   - **Architectural Differences**: Detail the differences in the number and function of their respective gates and state vectors.
   - **Computational Complexity**: Explain which architecture is generally more computationally efficient and justify your answer based on the number of parameters.
   - **Performance Trade-offs**: Discuss the typical performance differences observed in practice. When might one architecture be preferred over the other?

1.1.4 Backpropagation Through Time (BPTT)

1. Explain the Backpropagation Through Time (BPTT) algorithm. Your explanation should describe the two conceptual phases of the algorithm and how it enables the use of gradient-based optimization for training RNNs.

2. What is **Truncated Backpropagation Through Time (T-BPTT)**? Explain the primary motivation for using this variant of BPTT and describe its mechanism. How does T-BPTT balance computational feasibility with the need to learn from sequential data?

1.2 Word Embeddings & Language Modeling

1.2.1 Word Embeddings

1. Define the term "word embedding." Explain why dense, low-dimensional vector representations are generally superior to sparse, high-dimensional one-hot encodings for representing words in natural language processing tasks.

2. State the **distributional hypothesis** and explain its fundamental role in motivating and shaping the development of word embedding models like Word2Vec and GloVe.

3. Compare and contrast the Continuous Bag-of-Words (CBOW) and Skip-gram architectures of the Word2Vec model. Your comparison must address:

   - Their respective objective functions (i.e., what each model is trained to predict).
   - Their relative computational efficiency.
   - Their typical performance characteristics, particularly concerning frequent versus rare words.

1.2.2 Language Modeling

1. Provide a formal definition of a statistical language model. What is the primary task that a language model is trained to perform?

2. Describe the architecture of an RNN-based language model. Specify the nature of the inputs to the model at each time step, the role of the hidden state, and the structure of the output layer used to predict the next word in a sequence.

3. Define the **perplexity** metric. Provide its mathematical formula and explain its interpretation in the context of evaluating language models. What does a lower perplexity score signify about a model's performance?

1.2.3 Contextual vs. Non-Contextual Embeddings

1. Explain the primary limitation of non-contextual word embedding models such as Word2Vec or GloVe. Illustrate this limitation with a clear example using a polysemous word (e.g., "bank," "plant," "play"), showing how a single static vector fails to capture its different meanings in varying contexts.

2. Provide a high-level explanation of how models like ELMo or BERT generate **contextualized word embeddings**. How does this process differ from the static vector lookup used in non-contextual models? Explain why this dynamic, context-dependent approach represents a significant improvement for downstream NLP tasks.

1.2.4 Softmax and Output Layer

1. What is the role and function of the softmax layer in an RNN-based language model? Explain how it transforms the model's raw output scores (logits) into a valid probability distribution.

2. Explain why the full softmax calculation becomes a major computational bottleneck during the training of language models with large vocabularies. Your answer should identify the specific part of the softmax equation that is computationally expensive. Describe in detail one of the following methods for addressing this bottleneck: **Negative Sampling** or **Hierarchical Softmax**. Explain how your chosen method approximates the full softmax and reduces the computational complexity of the training step.

2 Coding challenges

2.1 Divan Hafez (30 points)

Note: You must complete all the `TODO` sections in the notebook to receive the full score.

2.2 Apple Stock Market (30 points)

Note: You must complete all the `TODO` sections in the notebook to receive the full score.