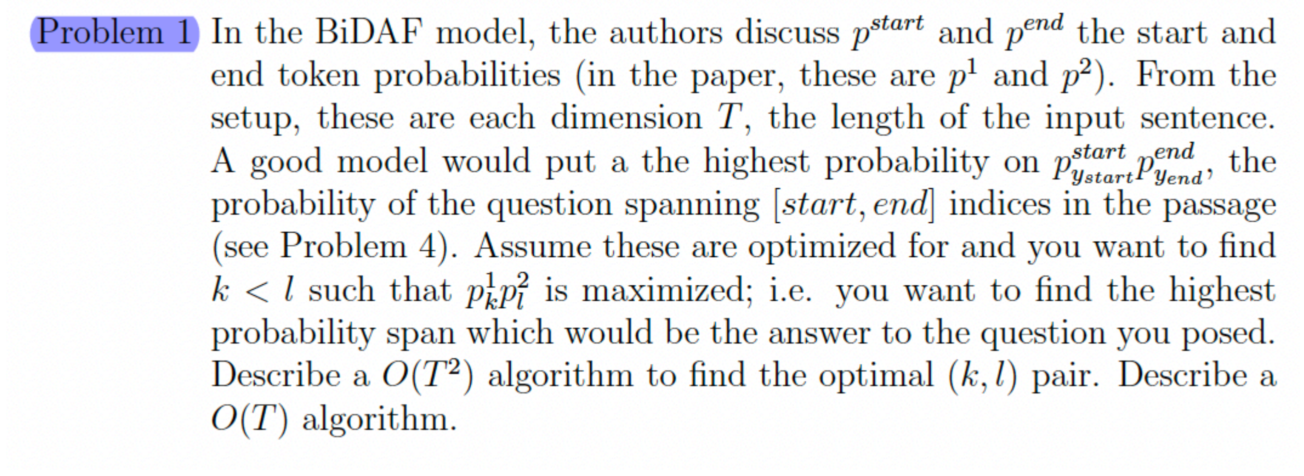


Problem 1 In the BiDAF model, the authors discuss p^{start} and p^{end} the start and end token probabilities (in the paper, these are p^1 and p^2). From the setup, these are each dimension T, the length of the input sentence. A good model would put a the highest probability on $p^{start}_{y_{start}} p^{end}_{y_{end}}$, the probability of the question spanning $[start, end]$ indices in the passage (see Problem 4). Assume these are optimized for and you want to find $k < l$ such that $p^1_k p^2_l$ is maximized; i.e. you want to find the highest probability span which would be the answer to the question you posed. Describe a $O(T^2)$ algorithm to find the optimal (k, l) pair. Describe a $O(T)$ algorithm.

O(T^2) Algorithm:

1. Initialize max_prob to 0 and optimal_pair to (0,0)
2. Iterate through all possible start positions k from 0 to T-1
   A. Calculate the probability of the span by multiplying p1_k (p_start) by p2_l (p_end) when k<l
   B. If the probability is greater than max_probability, update max_prob with probability and update optimal_pair to (k,l)
3. Return the optimal_pair

O(T) Algorithm:

1. Initialize max_prob to 0 and optimal_pair to (0,0)
2. Iterate k from 0 to T-1
   A. Keep track of the maximum p1_k (p_start) encountered so far, call it max_pstart
   B. Calculate the probability of the span by multiplying max_pstart by p2_k
   C. If the probability is greater than max_proby, update max_prob and update optimal_pair to (max_pstart_index, k).
   D. If p1_k is greater than max_pstart, update max_pstart with p1_k and max_pstart_index with k
3. Return the optimal pair.

Problem 5 Here are some questions on ULM-Fit.

- Describe the 3 steps of ULM-Fit at a high level.
- What do the authors argue should be the representation fed to each classifier? I.e. What is the input to the new classifier layer added in Step 3?
- What is catastrophic forgetting? What is discriminative fine tuning in ULM-Fit?
- What is gradual unfreezing in ULM-Fit?

5.1 Three steps of ULM-Fit

Step 1. General Domain LM Pre-Training: The LM is trained on a general-domain corpus to capture general features of the language in different layers. This step is the most expensive, only needs to be performed once and improved performance and convergence of downstream models.

Step 2. Target task LM fine-tuning: This step applies a discriminative fine-tuning to converge faster as it only needs to adapt to the features of the target data which allow to train a robust LM.

Step 3. Target task classifier fine-tuning: In this final step, a classifier is added to the fine-tuned LM, and both are fine-tuned together using discriminative fine-tuning and gradual unfreezing of layers. This step involves training a classifier that can predict labels for new data based on what was learned in steps 1 and 2.

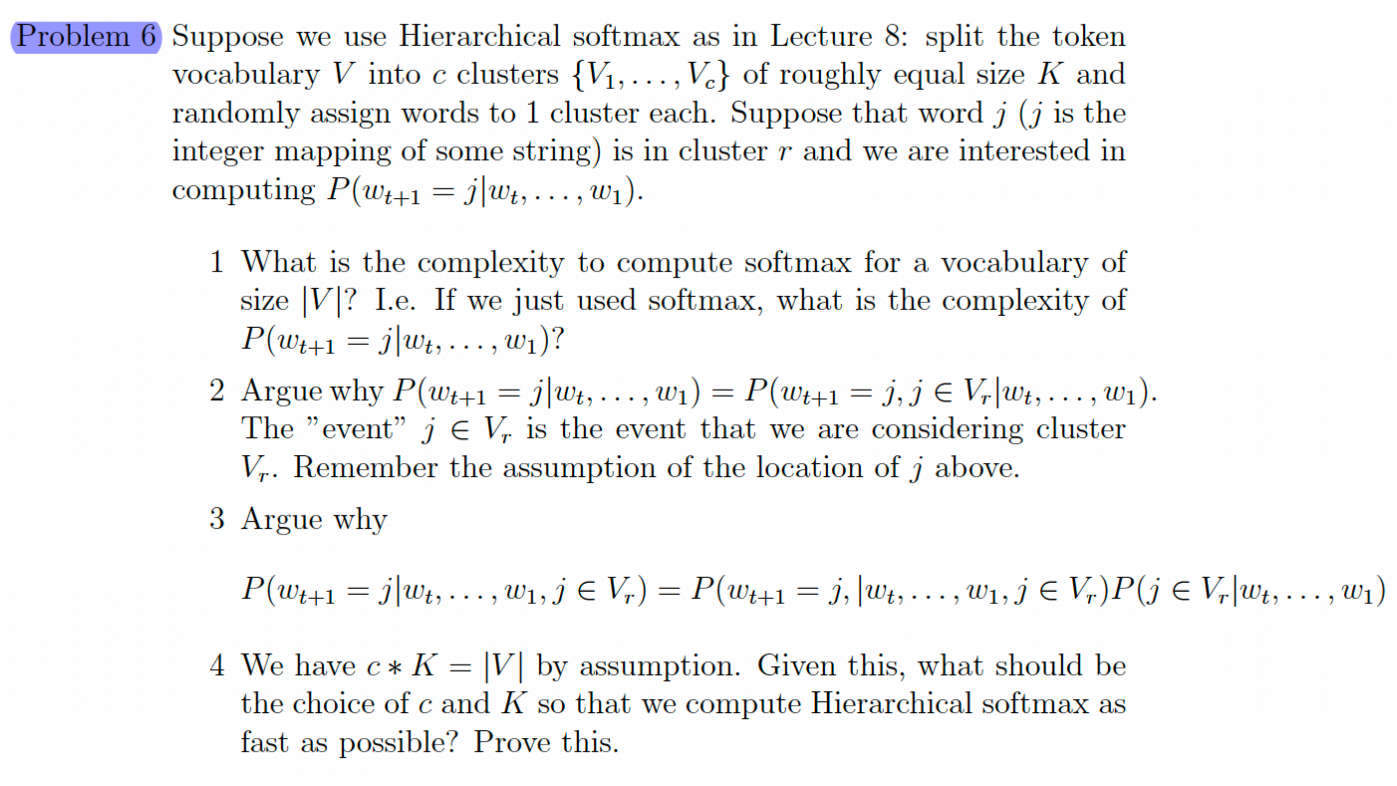
5.2 The input for the classifier are concat pooling, for hidden states H = {h_1,...,h_T}, the input is

h_c = [h_T, maxpool(H), meanpool(H)]

5.3 Catastrophic forgetting is eliminating the benefit of the information captured through language modeling caused by the aggressive fine tuning.

Discriminative fine-tuning is a technique used in transfer learning where each layer of a pre-trained neural network model is fine-tuned with a different learning rate. Instead of using the same learning rate for all layers of the model, discriminative fine-tuning allows us to tune each layer with different learning rates.

5.4 Rather than fine-tuning all layers at once, which risks catastrophic forgetting, we propose to gradually unfreeze the model starting from the last layer as this contains the least general knowledge.



Problem 6 Suppose we use Hierarchical softmax as in Lecture 8: split the token vocabulary V into c clusters $\{V_1, \dots, V_c\}$ of roughly equal size K and randomly assign words to 1 cluster each. Suppose that word j (j is the integer mapping of some string) is in cluster r and we are interested in computing $P(w_{t+1} = j | w_t, \dots, w_1)$.

1 What is the complexity to compute softmax for a vocabulary of size $|V|$? I.e. If we just used softmax, what is the complexity of $P(w_{t+1} = j | w_t, \dots, w_1)$?

2 Argue why $P(w_{t+1} = j | w_t, \dots, w_1) = P(w_{t+1} = j, j \in V_r | w_t, \dots, w_1)$. The "event" $j \in V_r$ is the event that we are considering cluster V_r. Remember the assumption of the location of j above.

3 Argue why

$$P(w_{t+1} = j | w_t, \dots, w_1, j \in V_r) = P(w_{t+1} = j, | w_t, \dots, w_1, j \in V_r) P(j \in V_r | w_t, \dots, w_1)$$

4 We have $c * K = |V|$ by assumption. Given this, what should be the choice of c and K so that we compute Hierarchical softmax as fast as possible? Prove this.

Problem 6.1

The complexity is O(V), The complexity of choosing the correct cluster is O(C), given cluster r, the complexity of choosing the word j in the cluster r is O(K).

Total complexity O(C+K) = O(2*sqrt(V)) = O(V), by applying binary balanced tree, the time complexity can be reduced to O(log V)

Problem 6.2

In the hierarchical softmax, we compute the P(w_j | context) by considering the probabilities of selecting the correct cluster and then selecting the correct word in the cluster. We first compute the probability of choosing the right cluster, and then compute the probability of choosing the right word j in the cluster. The event j in V_r means the word j in cluster r, since the word j only assigned to one cluster, if we consider word j we have to consider the word belong to cluster V_r.

Therefore, P(W_j | context) = P(W_j, j in V_r | context)

Problem 6.3

By the chain rule of joint probability,

P(W_j | context, j in V_r) * P(j in V_r | context) = P(W_j, j in V_r | context) = P(W_j | context)

Therefore, P(W_j | context) = P(W_j | context, j in V_r) * P(j in V_r | context)

Problem 6.4

A hierarchical decomposition represented by a balanced binary tree should provide a exponential speed-up, on the order of |V|/log|V|.

Each word v must be represented by a bit vector (b_1(v),...,b_m(v)) This can be achieved by building a binary hierarchical clustering of words, and a method for doing so is presented in the next section.

$$P(v|w_{t-1},\ldots,w_{t-n+1}) = \prod_{j=1}^{m} P(b_j(v)|b_1(v),\ldots,b_{j-1}(v),w_{t-1},\ldots,w_{t-n+1})$$

$$P(b=1|node,w_{t-1},\ldots,w_{t-n+1}) = \text{sigmoid}(\alpha_{node}+\beta'.\tanh(c+Wx+UN_{node}))$$

Source from: https://www.iro.umontreal.ca/~lisa/pointeurs/hierarchical-nnlm-aistats05.pdf