TTIC 31230 Fundamentals of Deep Learning, winter 2020

Quiz 5

Problem 1. The Spatial Transformer. 25 Points A self-attention layer in the transformer takes a sequence of vectors $h_{\text{in}}[T, J]$ and computes a sequence of vectors $h_{\text{out}}[T, J]$ using the following equations where k ranges over "heads". Heads are intended to allow for different relationship between words such as "coreference" for a pronoun or "subject of" for a verb. But the actual meaning emerges during training and is typically difficult or impossible to interpret. In the following equations we typically hve $U < J$ and we require $I = J/K$ so that the concatenation of K vectors of dimension I is a vector of dimension J.

$$
\begin{aligned}
\text{Query}[k, t, U] &= W^Q[k, U, J] h_{\text{in}}[t, J] \\
\text{Key}[k, t, U] &= W^K[k, U, J] h_{\text{in}}[t, J] \\
\alpha[k, t_1, t_2] &= \underset{t_2}{\text{softmax}}\ \text{Query}[k, t_1, U]\text{Key}[k, t_2, U] \\
\text{Value}[k, t, I] &= W^V[k, I, J] h_{\text{in}}[t, J] \\
\text{Out}[k, t, I] &= \sum_{t'} \alpha[k, t, t'] \text{Value}[k, t', I] \\
h_{\text{out}}[t, J] &= \text{Out}[1, t, I]; \cdots ; \text{Out}[K, t, I]
\end{aligned}
$$

Just as CNNs can be done in two dimensions for vision and in one dimension for language, the transformer can be done in two dimensions for vision — the so-called spatial transformer.

Rewrite the equations from problem 1 so that the time index t is replaced by spatial dimensions x and y.

Solution:

$$
\begin{aligned}
\text{Query}[k,x,y,U] &= W^Q[k,U,J]h_{\text{in}}[x,y,J] \\
\text{Key}[k,x,y,U] &= W^K[k,U,J]h_{\text{in}}[x,y,J] \\
\alpha[k,x_1,y_1,x_2,y_2] &= \underset{x_2,y_2}{\text{softmax}}\ \text{Query}[k,x_1,y_1,U]\text{Key}[k,x_2,y_2,U] \\
\text{Value}[k,x,y,I] &= W^V[k,I,J]h_{\text{in}}[x,y,J] \\
\text{Out}[k,x,y,I] &= \sum_{x',y'} \alpha[k,x,y,x',y']\text{Value}[k,x',y',I] \\
h_{\text{out}}[x,y,J] &= \text{Out}[1,x,y,I];\cdots;\text{Out}[K,x,y,I]
\end{aligned}
$$

Problem 2. REINFORCE for BLEU Translation Score. 25 points. Consider training machine translation on a corpus of translation pairs (x, y) where x is, say, an English sentence $x_1, \ldots, \text{EOS}$ and y is a French sentence $y_1, \ldots, \text{EOS}$ where EOS is the "end of sentence" tag.

Suppose that we have a parameterized autoregressive model defining $P_\Phi(y_t|x, y_1, \ldots, y_{t-1})$ so that $P_\Phi(y_1, \ldots, y_T|x) = \prod_{t=1}^{T'} P_\Phi(y_t|x, y_1, \ldots, y_{t-1})$ where y_T is EOS.

For a sample $\hat{y}$ from $P_\Phi(y|x)$ we have a non-differentiable BLEU score $\text{BLEU}(\hat{y}, y) \geq 0$ that is not computed until the entire output y is complete and which we would like to maximize.

(a) Give an SGD update equation for the parameters Φ for the REINFORCE algorithm for maximizing $E_{\hat{y}\sim P_\Phi(y|x)}$ for this problem.

Solution: For $\langle x, y\rangle$ samples form the training corpus of translation pairs, and for $\hat{y}_1, \ldots, \hat{y}_T$ sampled from $P_\Phi(\hat{y}|x)$ we update Φ by

$$
\Phi \texttt{ += } \eta \text{BLEU}(\hat{y}, y) \sum_{t=1}^{T} \nabla_\Phi \ln P_\Phi(\hat{y}_t|x, \hat{y}_1, \ldots, \hat{y}_{t-1})
$$

Samples with higher BLEU scores have their probabilities increased.

(b) Suppose that somehow we reach a parameter setting Φ where $P_\Phi(y|x)$ assigns probability very close to 1 for a particular translation $\hat{y}$ so that in practice we will always sample the same $\hat{y}$. Suppose that this translation $\hat{y}$ has less than optimal BLEU score. Can the REINFORCE algorithm recover from this situation and consider other translations? Explain your answer.

Solution: No. The REINFORCE algorithm will not recover. The update will only increase the probability of the single translation which it always selects. A deterministic policy has zero gradient and is stuck.

(c) Modify the REINFORCE update equations to use a value function approximation $V_\Phi(x)$ to reduce the variance in the gradient samples and where V_Φ is trained by Bellman Error. Your equations should include updates to train $V_\Phi(x)$ to predict $E_{\hat{y}\sim P(y|x)}$ BLEU$(\hat{y}, y)$. (Replace the reward by the "advantage" of the particular translation).

Solution: For $\langle x, y\rangle$ sampled form the training corpus of translation pairs, and for $\hat{y}_1, \ldots, \hat{y}_T$ sampled from $P_\Phi(\hat{y}|x)$ we udate Φ by

$$\Phi \quad \texttt{+=} \quad \eta(\text{BLEU}(\hat{y}, y) - V_\Phi(x)) \sum_{t=1}^{T} \nabla_\Phi \ \ln P_\Phi(\hat{y}_t | x, \hat{y}_1, \ldots, \hat{y}_{t-1})$$

$$\Phi \quad \texttt{-=} \quad \eta \nabla_\Phi (V_\Phi(x) - \text{BLEU}(\hat{y}, y))^2 \ = \ 2\eta(V_\Phi(x) - \text{BLEU}(\hat{y}, y)) \nabla_\Phi V_\Phi(x)$$

Problem 3. AlphaZero for BLEU Translation Score. 25 Points.

A version of AlphaZero can be defined as follows.

To select a move in the game, first construct a search tree over possible moves to evaluate options.

The tree is grown by running "simulations". Each simulation descends into the tree from the root selecting a move from each position until the selected move has not been explored before. When the simulation reaches an unexplored move it expands the tree by adding a node for that move. Each simulation returns the value $V_\Phi(s)$ for the newly added node s.

Each node in the search tree represents a board position s and stores the following information which can be initialized by running the value and policy networks on position s.

- $V_\Phi(s)$ — the value network value for the position s.
- For each legal move a from s, the policy network probability $\pi_\Phi(s, a)$.
- For each legal move a from s, the number $N(s, a)$ of simulations that have taken move a from s. This is initially zero.
- For each legal move a from s with $N(s, a) > 0$, the average $\hat{\mu}(s, a)$ of the values of the simulations that have taken move a from position s.

In descending into the tree, simulations select the move $\text{argmax}_a U(s,a)$ where we have

$$U(s,a) = \begin{cases} \lambda_u \pi_\Phi(s,a) & \text{if } N(s,a) = 0 \\ \hat{\mu}(s,a) + \lambda_u \pi_\Phi(s,a)/N(s,a) & \text{otherwise} \end{cases} \quad (1)$$

When the search is completed, we must select a move from the root position. For this we use a post-search stochastic policy

$$\pi_{s_{\text{root}}}(a) \propto N(s_{\text{root}}, a)^\beta \quad (2)$$

where β is a temperature hyperparameter.

For training we construct

$$\text{a replay buffer of triples } (s_{\text{root}}, \pi_{s_{\text{root}}}, z) \quad (3)$$

accumulated from self play where s_{root} is a root position from a search during a game, $\pi_{s_{\text{root}}}$ is the post-search policy constructed for s_{root}, and z is the outcome of that game.

Training is then done by SGD on the following objective function.

$$\Phi^* = \underset{\Phi}{\text{argmin}}\ E_{(s,\pi,z)\sim\text{Replay},\ a\sim\pi} \begin{pmatrix} (V_\Phi(s) - z)^2 \\ -\lambda_1 \log \pi_\Phi(a|s) \\ +\lambda_2 ||\Phi||^2 \end{pmatrix} \quad (4)$$

Reformulate this algorithm to optimizing BLEU score in machine translation. More specifically,

(a) Define the optimization of the BLEU score as a tree search problem. What are the nodes and what are the moves?

Solution: The tree is defined by sequentially choosing words. Each node correspond to a sequences of moves — a node at depth d in the tree corresponds to a prefix of length d (the first d words) of a possible translation. The moves from a given node correspond to the choice of the next word.

(b) AlphaZero has three levels — complete games resulting in a final outcome z — move selection at each position in a complete game based on UTC algorithm — and simulations within the UTC algorithm. Describe how each of these can be interpreted in an algorithm for optimizing BLEU score in machine translation.

Solution: A "complete game" consists of selecting a move (word) at each position in the game ending one specific translation.

Move selection corresponds to selecting the next word using athe UCT tree search algorithm.

We can use the same simulation algorithm as in AlphaZero. It might also be possible to let each simulation generate a full translation, although there is a danger of the simulation getting very deep.

(c) What do we take z to be in the replay buffer and in equation (4)?

Solution: z should be taken to be the BLEU score of the final translation generated in the "game".

Problem 4. Rapid Mixing for Asymptotic Avergage Reward. 25 Points.

We consider a case where we are interested in asymptotic average reward.

$$R(\pi) \;=\; \lim T \to \infty \, \frac{1}{T} \sum_{t=1}^{T} r_t$$

For a given policy π we have a Markov process over states — a well defined state transition probability $P_\pi(s_{t+1}|s_t)$ defined by

$$P_\pi(s_{t+1}|s_t) \;=\; \sum_a \pi(a|s_1) P_\pi(s_2|s_1, a)$$

Under very mild conditions a Markov process has a well define stationary distribution on states which we will write $P_\pi(s)$. This distribution is "stationary" in the sense that

$$\sum_{s_1} P_\pi(s_1) P_\pi(s_2|s_1) \;=\; P_\pi(s_2)$$

(a) Write the asymptotic average reward $R(\pi)$ in terms of the stationary distribution P_π, the policy $\pi(a|s)$ and the reward function $R(s, a)$

Solution:

$$R(\pi) = E_{s \sim P_\pi(s),\; a \sim \pi(a|s)} \; R(s, a)$$

(b) Now for $\Delta t > 1$ we define $P_\pi(s_{t+\Delta t}|s_t)$ recursively as by

$$P_\pi(s_{t+\Delta t}|s_t) = \sum_{s_{t+\Delta t-1}} P_\pi(s_{t+\Delta t-1}|s_t) P_\pi(s_{t+\Delta t}|s_{t+\Delta t-1})$$

We now assume a "mixing parameter" $0 < \gamma < 1$ for π defined by the property

$$\sum_{s_{t+\Delta t}} |P_\pi(s_{t+\Delta t}|s_t) - P_\pi(s_{t+\Delta t})| \leq \gamma^{\Delta t}$$

We now define an advantage function on state-action pairs to be the "extra" reward we get by taking action a (rather than drawing from $\pi(a|s)$) summed over all time.

$$A(s,a) = E \sum_{t=0}^{\infty} (r_t - R(\pi)) \;\mid\; s_0 = s,\; a_0 = a$$

Assuming the reward is bounded by $r_{\max}$ and that we have the above mixing parameter γ, give a (finite) upper bound on the infinite sum $A(s,a)$.

Solution:

$$\begin{aligned}
& E\, r_t - R(\pi) \;\; | s_0 = s,\; a_0 = a, t > 0 \\
= \;& \left(\sum_{s_t} P_\pi(s_t|s_0) E_{a\sim\pi(a|s_t)}\; R(s_t,a) \right) - R(\pi) \\
= \;& \left(\sum_{s_t} (P_\pi(s_t) + P_\pi(s_t|s_0) - P_\pi(s_t)) E_{a\sim\pi(a|s_t)}\; R(s_t,a) \right) - R(\pi) \\
= \;& R(\pi) + \left(\sum_{s_t} (P_\pi(s_t|s_0) - P_\pi(s_t)) E_{a\sim\pi(a|s_t)}\; R(s_t,a) \right) - R(\pi) \\
= \;& \sum_{s_t} (P_\pi(s_t|s_0) - P_\pi(s_t)) r_{\max} \\
\leq \;& r_{\max} \gamma^t \\
A(s,a) \;\leq \;& r_{\max} \sum_{t=0}^{\infty} \gamma^t = \frac{r_{\max}}{1-\gamma}
\end{aligned}$$

It can be shown that

$$\nabla_\Phi R(\pi_\Phi) = E_{s\sim P_\pi(s),\; a\sim\pi(a|s)}\; \nabla_\Phi \ln \pi_\Phi(a|s)\; A(s,a)$$

You do not have to prove this.