Giới thiệu chung về mục tiêu on-policy prediction trong Reinforcement Learning

Trong Reinforcement Learning (RL), **on-policy prediction** tập trung vào việc ước lượng hàm giá trị (value function) của một chính sách (policy) hiện tại khi agent tương tác trực tiếp với môi trường theo chính sách đó. Việc ước lượng chuẩn xác hàm giá trị giúp đánh giá hiệu năng của chính sách và làm cơ sở cho các bước cải tiến (policy improvement). Các slide "The Objective for On-policy Prediction" trình bày ba nội dung chính: (1) mục tiêu mean-squared value error cho policy evaluation, (2) phương pháp gradient descent và cách áp dụng trong RL, và (3) kỹ thuật state aggregation để xấp xỉ giá trị trong không gian trạng thái lớn. Phần dưới đây sẽ giải thích và phân tích chi tiết từng khái niệm, thuật toán và ví dụ minh họa, kèm theo dẫn chứng từ các nguồn tham khảo.

Mean-Squared Value Error Objective cho Policy Evaluation

Mean-Squared Value Error (MSVE) là một objective thường dùng trong policy evaluation để đo lường độ lệch giữa giá trị thật $v_\pi(s)$ của chính sách π và giá trị xấp xỉ $\hat{v}(s;w)$ do hàm tham số (parameterized function) với tham số w cung cấp. Cụ thể, nếu ký hiệu $d(s)$ là phân phối trạng thái theo bước thời gian trong quá trình tương tác on-policy (steady-state distribution dưới π), thì objective có thể viết:

$$J(w) = \sum_{s \in \mathcal{S}} d(s)[v_\pi(s) - \hat{v}(s;w)]^2$$

Mục tiêu là tìm tham số w sao cho $J(w)$ nhỏ nhất, tức giảm thiểu lỗi bình phương trung bình giữa giá trị thật và giá trị xấp xỉ trên phân phối trạng thái quan tâm legacydirs.umiacs.umd.edu stats.stackexchange.com . Thực tế, $v_\pi(s)$ thường không biết trước, nhưng có thể ước lượng qua Monte Carlo (MC) returns hoặc bootstrap (TD target). Khi dùng MC, target cho mỗi trạng thái s_t tại thời điểm t là $G_t = \sum_{k=0}^{\infty} \gamma^k r_{t+k+1}$; khi dùng TD(0), target là $r_{t+1} + \gamma\hat{v}(s_{t+1};w)$. Cả hai đều hướng đến giảm thiểu sai số bình phương đối với target tương ứng. Trong slide, khái niệm "Value Error Objective" và việc "Adapting the weights to minimize the mean squares value error objective" nhấn mạnh việc dùng gradient descent để điều chỉnh tham số sao cho độ lệch giữa $\hat{v}$ và target càng nhỏ càng tốt legacydirs.umiacs.umd.edu users.ece.cmu.edu .

- **Vai trò phân phối trạng thái $d(s)$:** Trong on-policy prediction, phân phối $d(s)$ xác định tầm quan trọng của từng trạng thái trong objective. Nếu agent hiếm khi gặp một trạng thái, lỗi ở trạng thái đó ít ảnh hưởng đến $J(w)$. Phân phối này có thể được ước tính từ tần suất xuất hiện trạng thái khi chạy policy.
- **Khó khăn thực tế:** Giá trị thật $v_\pi(s)$ không biết trước, nên không thể trực tiếp tính gradient đầy đủ của $J(w)$. Thay vào đó, sử dụng các ước lượng target (MC hoặc TD) và áp dụng **semi-gradient** (đối với TD) hoặc gradient của MSE (đối với MC) để cập nhật tham số. Sự khác biệt giữa full gradient và semi-gradient liên quan đến việc target phụ thuộc tham số hay không; TD target phụ thuộc $\hat{v}(s';w)$ nên gradient thực tế bỏ qua đạo hàm của target, dẫn đến semi-gradient update. Điều này vẫn hội tụ dưới điều kiện linear approximation on-policy legacydirs.umiacs.umd.edu users.ece.cmu.edu .

Giới thiệu về Gradient Descent và áp dụng trong Reinforcement Learning

Gradient Descent là thuật toán tối ưu bậc một (first-order optimization), điều chỉnh tham số theo hướng làm giảm giá trị hàm mục tiêu. Trong RL, gradient descent được áp dụng ở nhiều nơi: policy gradient methods, value function approximation, và cập nhật trong actor-critic. Các slide nhắc về "Introducing Gradient Descent" và minh họa khái niệm local minimum, local maximum, global minimum, giúp người học nắm khái niệm cơ bản về bề mặt hàm mất mát (loss landscape) và cách gradient descent tìm hướng giảm.

- **Khái niệm cơ bản:** Với hàm mục tiêu $J(w)$, mỗi bước cập nhật $w \leftarrow w - \alpha \nabla J(w)$ (với α là learning rate) giúp di chuyển tham số theo hướng gradient âm, giảm giá trị hàm mục tiêu. Trong RL, khi dùng MC, $J(w) = \mathrm{E}[(G_t - \hat{v}(s_t; w))^2]$, gradient descent thuần túy có thể áp dụng. Khi dùng TD, dùng semi-gradient nhưng cơ bản vẫn là cập nhật theo hướng dự kiến làm giảm sai số giữa ước lượng và target. Gradient descent trong RL có thể bị nhiễu do target không ổn định và dữ liệu không độc lập (non-i.i.d) do tương tác liên tục với môi trường users.ece.cmu.edu proceedings.neurips.cc .
- **Gradient descent cho policy gradient:** Trong policy gradient methods, mục tiêu là tối đa hóa expected return $J(\theta) = \mathrm{E}_\pi[G_t]$. Dựa vào policy gradient theorem, gradient $\nabla_\theta J(\theta)$ được ước lượng từ trải nghiệm, và cập nhật theo $\theta \leftarrow \theta + \alpha \nabla_\theta J(\theta)$ (gradient ascent) datacamp.com davidstarsilver.wordpress.com . Slides trình bày "Policy Gradient Methods" nhấn mạnh agent học trực tiếp tham số policy $\pi_\theta(a|s)$ bằng gradient descent/ascent, tăng xác suất hành động mang lại reward cao hơn.
- **Value Function Approximation:** Khi xấp xỉ hàm giá trị (state-value $V(s)$ hoặc action-value $Q(s,a)$) bằng hàm tham số $\hat{v}(s; w)$ hoặc $\hat{q}(s, a; w)$, mục tiêu là giảm thiểu MSE hoặc TD error. Gradient descent (thực chất semi-gradient cho TD) được áp dụng: $w \leftarrow w + \alpha \delta_t \nabla_w \hat{v}(s_t; w)$ với $\delta_t = r_{t+1} + \gamma \hat{v}(s_{t+1}; w) - \hat{v}(s_t; w)$. Trong slide "Gradient Descent in RL" phần Value Function Approximation trình bày rõ cách áp dụng gradient descent để cập nhật tham số hàm xấp xỉ giá trị geeksforgeeks.org users.ece.cmu.edu .
- **Off-policy methods và actor-critic:** Off-policy methods (như Q-learning với function approximation) cũng dùng gradient descent để tối thiểu hóa TD error, dù thu thập dữ liệu từ policy khác. Actor-critic kết hợp policy gradient (actor) và value-based update (critic), cả hai đều dùng gradient descent để cập nhật tham số tương ứng datacamp.com davidstarsilver.wordpress.com .

Phân tích Gradient cho Policy Evaluation

Slide "Gradient for Policy Evaluation" nhắc rằng gradient của hàm xấp xỉ giá trị chỉ ra cách thay đổi trọng số để tăng hoặc giảm ước lượng cho một trạng thái: nếu sai số $\hat{v}(s)$ thấp hơn giá trị thật (difference positive), thì tăng estimate; nếu sai số ngược lại, giảm estimate. Cụ thể, với update semi-gradient TD:

$$\delta_t = r_{t+1} + \gamma \hat{v}(s_{t+1}; w) - \hat{v}(s_t; w), \qquad w \leftarrow w + \alpha\, \delta_t\, \nabla_w \hat{v}(s_t; w).$$

Với linear approximation, $\nabla_w \hat{v}(s_t; w) = \phi(s_t)$, nên update rất đơn giản: $w \leftarrow w + \alpha\delta_t\phi(s_t)$. Điều này thể hiện rõ: nếu $\delta_t > 0$ (ước lượng thấp hơn target), ta cộng $\phi(s_t)$ vào w, làm tăng $\hat{v}(s_t)$; nếu $\delta_t < 0$, làm giảm. Cách giải thích "If the difference is positive (the true value is higher than our estimate) → we should change the weights in the direction that increases our estimate" và ngược lại xuất phát từ công thức này geeksforgeeks.org users.ece.cmu.edu .

- **Ý nghĩa thực tiễn:** Cập nhật gradient giúp hàm xấp xỉ dần tiệm cận giá trị thật dưới phân phối trải nghiệm on-policy. Tuy nhiên, do bootstrap nên có thể không hội tụ về global minimum của MSVE mà chỉ hội tụ đến TD fixed point trong linear on-policy case, với sai số bounded theo discount factor và cấu trúc feature legacydirs.umiacs.umd.edu users.ece.cmu.edu .

State Aggregation cho xấp xỉ giá trị

State aggregation là một kỹ thuật đơn giản nhưng hiệu quả để giảm số lượng giá trị phải lưu và cập nhật khi không gian trạng thái lớn. Ý tưởng: chia nhóm các trạng thái tương tự vào cùng một aggregate (cluster) và gán chung một giá trị xấp xỉ cho tất cả trạng thái trong nhóm đó.

- **Định nghĩa toán học:** Giả sử tập trạng thái S được chia thành K nhóm $G_1, G_2, \ldots, G_K$. Định nghĩa feature vector $\phi(s)$ kích thước K sao cho $\phi_i(s) = 1$ nếu $s \in G_i$, và 0 nếu không. Với tham số $w \in \mathrm{R}^K$, xấp xỉ giá trị là $\hat{v}(s; w) = w^T\phi(s) = w_i$ khi $s \in G_i$. Khi cập nhật trạng thái $s \in G_i$, chỉ cập nhật thành phần w_i, và ảnh hưởng đến tất cả trạng thái trong nhóm ai.stackexchange.com ai.stackexchange.com .
- **Ví dụ minh họa:** Slide đưa ví dụ bảng 8 trạng thái được gộp thành 2 nhóm mỗi nhóm 4 trạng thái. Thay vì lưu 8 mục giá trị, chỉ cần 2 tham số. Khi nhận được trải nghiệm từ một trạng thái trong nhóm, update đồng thời cải thiện xấp xỉ cho các trạng thái khác trong cùng nhóm. Điều này hỗ trợ generalization nhưng đồng thời có thể gây bias nếu trạng thái trong cùng nhóm có giá trị thật khác biệt.
- **State aggregation là linear function approximation:** Như đã trình bày, feature one-hot theo nhóm, hàm xấp xỉ tuyến tính trên feature. Đây là trường hợp đặc biệt đơn giản của linear approximation ai.stackexchange.com ai.stackexchange.com .
- **Ưu và nhược điểm:**
  - Ưu điểm: Giảm chiều tham số, đơn giản, tiết kiệm bộ nhớ và tính toán, hỗ trợ generalization khi trạng thái trong nhóm có giá trị tương tự.
  - Nhược điểm: Nếu grouping không hợp lý, trạng thái khác biệt bị gộp chung sẽ dẫn đến bias lớn trong ước lượng. Thiết kế grouping (feature) cần dựa trên chuyên môn hoặc tự động qua các kỹ thuật clustering/adaptive aggregation. Các nghiên cứu về adaptive state aggregation (như Adaptive Soft Aggregation) nhằm xác định grouping tối ưu để giảm Bellman error proceedings.neurips.cc ai.stackexchange.com .

Mối liên hệ giữa các phần và ứng dụng thực tế

- **Từ MSVE đến cập nhật gradient:** Mục tiêu on-policy prediction là giảm MSVE. Trong thực tế, gradient descent đầy đủ khó áp dụng do v_π không biết, nhưng ta dùng MC hoặc TD semi-gradient để xấp xỉ gradient descent, điều chỉnh tham số hàm xấp xỉ một cách incrementally.
- **Vai trò của feature và function approximation:** Khi không gian trạng thái lớn, phải dùng hàm tham số. State aggregation là ví dụ cơ bản, còn có thể dùng tile coding, coarse coding, RBF, neural networks để biểu diễn feature phức tạp hơn. Việc chọn feature quyết định khả năng generalization và sai số xấp xỉ.
- **Gradient descent trong policy improvement và actor-critic:** Sau khi ước lượng giá trị (critic), actor dùng gradient của objective dựa trên giá trị để cập nhật policy; ngược lại, policy gradient cần ước lượng advantage hay value function. Điều này kết nối chặt chẽ giữa phần on-policy prediction và policy gradient.
- **Thách thức:** Dữ liệu không i.i.d, target phụ thuộc tham số, non-stationarity khi policy thay đổi, khiến gradient descent trong RL phức tạp hơn supervised learning thông thường. Cần điều chỉnh learning rate, regularization, replay buffer (trong off-policy), entropy bonus, v.v. để ổn định.

Kết luận và khuyến nghị

- **Hiểu rõ objective:** Luôn nắm MSVE là mục tiêu cơ bản cho on-policy prediction, nhưng trong thực tế dùng semi-gradient TD nhằm xấp xỉ. Nhận biết sai khác giữa full gradient và semi-gradient.
- **Áp dụng gradient descent đúng cách:** Cần chọn learning rate phù hợp, xem xét adaptive methods (Adam, RMSProp) khi dùng neural networks, nhưng vẫn cẩn trọng với bias và variance của gradient. Với linear approximation, update đơn giản, hội tụ chắc chắn trong on-policy.
- **Thiết kế feature hợp lý:** Nếu dùng state aggregation, cần grouping khôn ngoan để giảm bias; nếu dùng feature phức tạp hơn, cân nhắc trade-off giữa tính biểu diễn và tính ổn định. Có thể thử adaptive aggregation hoặc representation learning (deep RL) để tự học feature.
- **Liên kết với policy improvement:** Kết quả on-policy prediction trực tiếp hỗ trợ actor-critic và policy gradient; đảm bảo critic đủ chính xác để cung cấp gradient tốt cho actor.