Actuarial Computation and Simulation

Week 6: Risk-Sensitive RL Foundations (Distributional RL)

Aprida Siska Lestia

September 22, 2025

Motivation: Why Distributional RL?

- RL klasik mempelajari **rataan** nilai aksi: $Q^{\pi}(s,a) = \mathbb{E}[Z^{\pi}(s,a)]$.
- Banyak domain (finance/insurance) memiliki *tail risk* yang penting.
- **Distributional RL** mempelajari **distribusi return** $Z^{\pi}(s,a)$, bukan hanya mean.
- Manfaat: melihat *spread*, *asymmetry*, dan informasi kuantil untuk pengambilan keputusan.

From Mean to Full Distribution

Risk-neutral (mean-based):

$$Q^{\pi}(s,a) = \mathbb{E}\left[\sum_{t\geq 0} \gamma^t R_{t+1}\right]$$

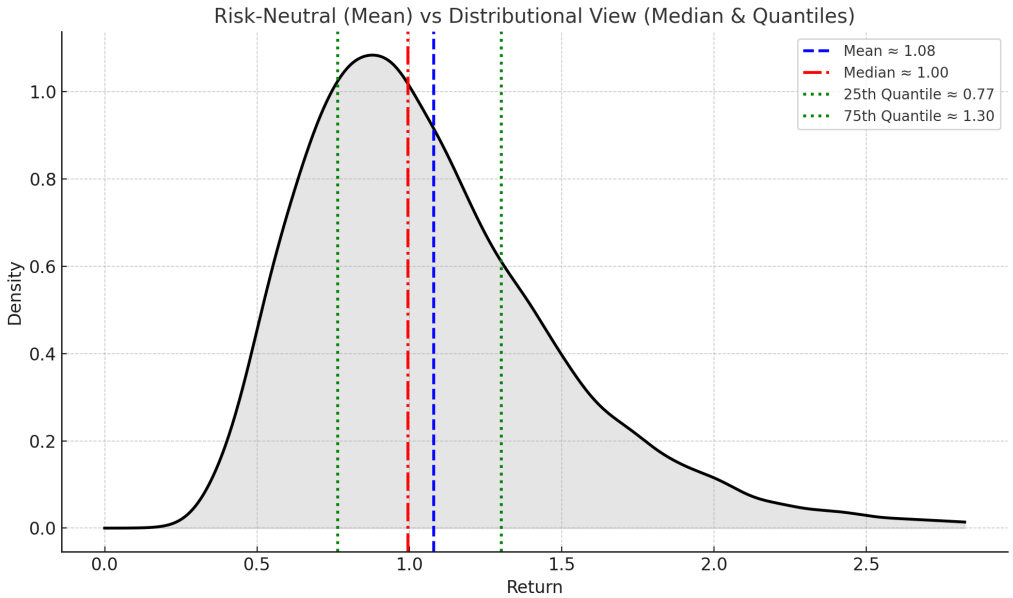
Distributional view:

$$Z^{\pi}(s,a) \in \mathcal{P}(\mathbb{R}),$$

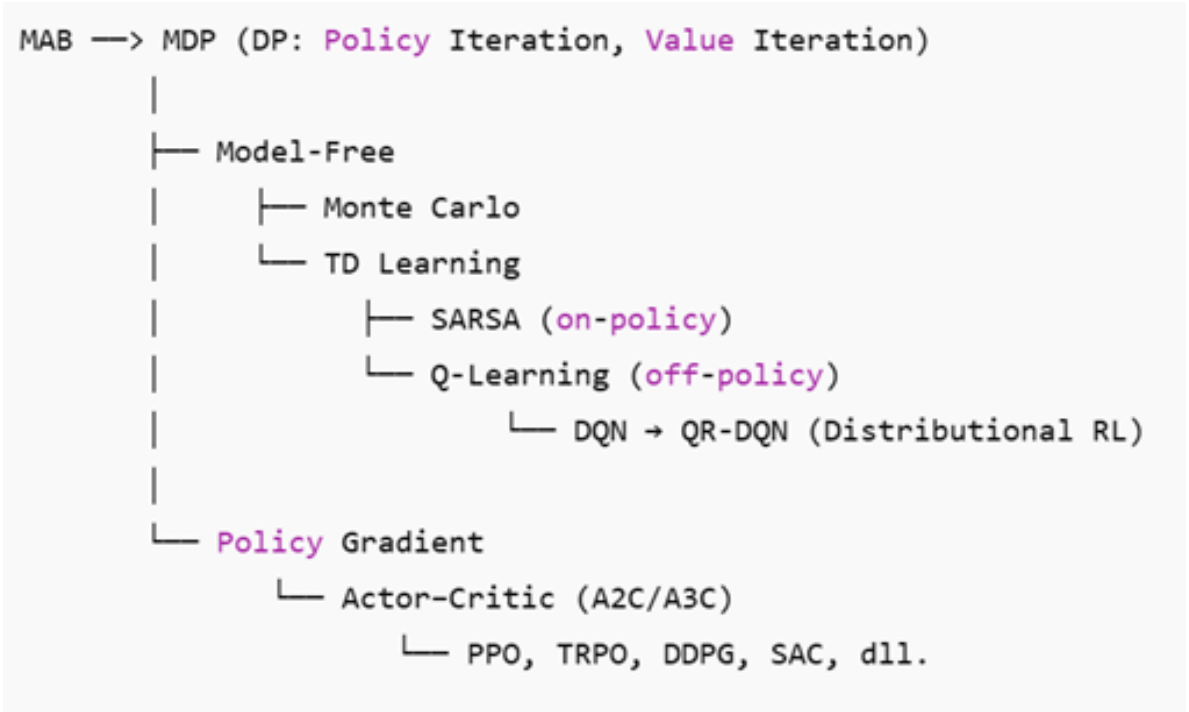
dimana $\mathcal{P}(\mathbb{R})$ adalah himpunan distribusi di $\mathbb{R}$.

$$Q^{\pi}(s,a) = \mathbb{E}[Z^{\pi}(s,a)]$$

Dengan Z^{π} bisa menentukan *mean*, *median*, maupun *quantiles*.



Evolusi RL → DRL



Peta Deep RL (DRL)

Jalur	**Algoritma**	**Representasi dengan NN**	**Karakteristik**
Value-Based DRL	DQN, Double DQN, QR-DQN	Q-Network (NN aproksimasi Q)	*Action space* diskrit, fokus *value*
Policy-Based DRL	PG, REINFORCE, PPO, TRPO	*Policy Network* (NN aproksimasi distribusi aksi)	Cocok untuk aksi kontinu, optimasi *policy* secara langsung
Hybrid (Actor–Critic)	A2C, A3C, DDPG, TD3, SAC	*Actor* NN (policy) + *Critic* NN (value)	Lebih stabil, bisa untuk diskrit & kontinu

Review Singkat: Q-learning → Deep Q-network (DQN)

Motivasi DQN

- Q-learning (tabel): $Q(s,a) \leftarrow Q(s,a) + \alpha\big[r + \gamma \max_{a'} Q(s',a') - Q(s,a)\big]$
- Tidak skala untuk state space besar/continuum.
- **DQN**: aproksimasi $Q_\theta(s,a)$ dengan neural network, memakai *replay buffer* dan *target network*.

Target & Loss DQN

$$y = r + \gamma \max_{a'} Q_{\theta^-}(s',a'),$$

$$\mathcal{L}(\theta) = \big(y - Q_\theta(s,a)\big)^2$$

Ringkasan implementasi

- Input: vektor fitur state s.
- Output: $[Q(s,a_1),\ldots,Q(s,a_K)]$ (satu skalar per aksi).
- Pemilihan aksi: ϵ-greedy terhadap $\arg\max_a Q(s,a)$.
- Stabilitas: target network (parameter θ^-), replay buffer, gradient clipping.

Dari DQN (Mean) ke Quantile Regression Deep Q-Network atau QR-DQN (Distribusi)

DQN (risk-neutral)

- Belajar **nilai harapan** return: $Q^{\pi}(s,a) = \mathbb{E}[Z^{\pi}(s,a)]$.
- Output per-aksi: **skalar** $Q(s,a)$.
- Keputusan: maksimalkan mean value.

QR-DQN (distributional)

- Belajar **distribusi** return $Z^{\pi}(s,a)$ via *quantile regression*.
- Output per-aksi: **vektor kuantil** $\hat{z}_a = \{\hat{q}_{a,\tau_i}\}_{i=1}^{N}$ untuk $0 < \tau_1 < \cdots < \tau_N < 1$.
- Keputusan: bisa berdasarkan mean/median/%kuantil (mis. 50th/75th).

Inti Perbedaan

DQN meminimalkan MSE terhadap *mean*; QR-DQN meminimalkan *quantile (pinball) loss* terhadap barisan kuantil $\Rightarrow$ informasi tail lebih kaya.

QR-DQN: Target Kuantil & Loss vs DQN

Pemilihan aksi untuk target (gaya Double-DQN):

$$a^* = \arg\max_{a'} \frac{1}{N} \sum_{i=1}^{N} \hat{q}^{\text{online}}_{a',\tau_i}(s'), \qquad y_j = r + \gamma\, \hat{q}^{\text{tgt}}_{a^*,\tau_j}(s')$$

Pinball / Quantile Loss (umum, versi non-Huber):

$$\rho_\tau(u) = u\big(\tau - \mathbf{1}\{u < 0\}\big), \qquad \mathcal{L} = \frac{1}{N} \sum_{i=1}^{N} \sum_{j=1}^{N} \rho_{\tau_i}\big(y_j - \hat{q}_{a,\tau_i}(s)\big)$$

Intuisi:

- DQN $\Rightarrow$ cocok bila mean cukup merepresentasikan risiko.
- QR-DQN $\Rightarrow$ belajar bentuk distribusi (spread, skew, tail).
- Dapat membuat kebijakan *risk-aware* (contoh: pilih aksi memaksimalkan median/75th).

Catatan praktis: target network, replay buffer, gradient clipping; N kuantil (mis. 32–101).

QR-DQN: Core Idea (Main Content)

- Representasi distribusi return dengan N **quantiles** tetap: $0 < \tau_1 < \cdots < \tau_N < 1$. Umum dipakai grid tengah $\tau_i = \frac{i-0.5}{N},\ i = 1, \ldots, N$.
- Untuk setiap aksi a, jaringan mengeluarkan **vektor kuantil** $\hat{Z}_a = \{\hat{q}_{a,\tau_i}\}_{i=1}^{N}$ (approximated quantile function).
- **Tujuan**: mendekati kuantil sejati q_{a,τ_i} dari $Z^\pi(s,a)$ sehingga bentuk distribusi (spread, skew, tail) ikut terpelajari.
- *Catatan praktis*:
  - Monotonicity (hindari *quantile crossing*) diupayakan dengan arsitektur/regularisasi yang baik; secara empiris crossing kecil tidak fatal.
  - Trade-off N: lebih besar $\Rightarrow$ distribusi lebih halus, biaya komputasi meningkat. Praktik: $N \in [32, 101]$.

QR Loss (Pinball / Quantile Regression Loss)

Untuk residu $u = y - \hat{y}$, **pinball loss**:

$$\rho_\tau(u) = u \cdot \big(\tau - \mathbf{1}\{u < 0\}\big).$$

Dalam praktik sering dipakai **Huberized quantile loss** (lebih stabil untuk outlier kecil); ide dasarnya mengganti bagian linear di sekitar $u = 0$ dengan kuadratik halus (parameter ambang κ).

Training target (Bellman) untuk kuantil aksi terpilih (gaya Double-DQN):

$$a^* = \arg\max_{a'} \frac{1}{N} \sum_{i=1}^{N} \hat{q}^{\text{online}}_{a',\tau_i}(s_{t+1}), \qquad y_j = r_{t+1} + \gamma\, \hat{q}^{\text{tgt}}_{a^*,\tau_j}(s_{t+1}).$$

Loss total (per transisi):

$$\mathcal{L} = \frac{1}{N} \sum_{i=1}^{N} \sum_{j=1}^{N} \rho_{\tau_i}\big(y_j - \hat{q}_{a_t,\tau_i}(s_t)\big) \quad \text{(dapat dihemat dengan subsampling indeks } j\text{)}.$$

Catatan: mekanisme seperti DQN (replay & target network), tetapi optimisasi dilakukan pada **prediksi kuantil** alih-alih satu nilai mean.

QR-DQN: Arsitektur dan Hiperparameter

Arsitektur:

- Backbone (MLP/CNN sesuai state), **output shape**: $|\mathcal{A}| \times N$ (setiap aksi memiliki N kuantil).
- Contoh: $N = 51$, hidden 2–3 layer, aktivasi ReLU.

Hiperparameter contoh:

- Optimizer Adam, learning rate 10^{-3}; $\gamma = 0.99$; batch 64–128.
- Target update: hard tiap 1–5k langkah *atau* soft update ($\tau_{\text{polyak}} \approx 0.005$).
- Replay buffer $\approx 10^5$; eksplorasi ϵ-greedy seperti DQN; gradient clipping.

Output keputusan (risk-neutral vs risk-aware):

- *Mean-based*: $\frac{1}{N} \sum_{i=1}^{N} \hat{q}_{a,\tau_i}$ (setara memaksimalkan ekspektasi).
- *Quantile-based*: pilih aksi yang memaksimalkan kuantil tertentu (mis. median/75th) untuk kebijakan **lebih konservatif** (relevan di insurance/finance).

QR-DQN: Training Loop (Pseudocode)

1. Inisialisasi *online* & *target* networks (output $|\mathcal{A}| \times N$ kuantil).
2. Untuk setiap episode / langkah:
   - 2.1 Pilih aksi a_t dengan ϵ-greedy berdasar **mean** kuantil *atau* kuantil terpilih (mis. median/75th).
   - 2.2 Dapatkan $(s_t, a_t, r_{t+1}, s_{t+1}, \texttt{done})$; simpan ke replay.
   - 2.3 Ambil minibatch; untuk tiap transisi hitung **target kuantil**:

$$a^* = \arg\max_{a'} \frac{1}{N} \sum_i \hat{q}^{\text{online}}_{a',\tau_i}(s_{t+1}),$$

$$y_j = \begin{cases} r_{t+1}, & \text{jika done} = 1, \\ r_{t+1} + \gamma\, \hat{q}^{\text{tgt}}_{a^*,\tau_j}(s_{t+1}), & \text{lainnya.} \end{cases}$$

   - 2.4 Hitung **Huberized quantile loss** antar pasangan (τ_i, y_j); untuk efisiensi boleh *subsample* indeks j.
   - 2.5 **Backprop** ke online net; sinkronisasi target: *hard* tiap K langkah *atau soft* $(\theta^- \leftarrow (1-\tau)\theta^- + \tau\theta)$.
   - 2.6 **Terminal handling**: jika done$= 1$, gunakan $y_j = r_{t+1}$ (tanpa bootstrapping)
3. Evaluasi: bandingkan kebijakan **mean** vs **kuantil** (median/75th).

Practical Notes & Pitfalls

- **Stabilitas**: gradient clipping, target network, replay buffer memadai.
- **Eksplorasi**: jadwal ϵ (mis. $1.0 \to 0.05$ secara linier selama 10^5 langkah).
- **Target update**: *hard* setiap 1–5k langkah *atau soft* (Polyak $\tau \approx 0.005$).
- **Biaya** N: N besar $\Rightarrow$ distribusi lebih halus, komputasi naik; praktik $N \in [32, 101]$. Pertimbangkan *subsample* indeks j pada loss.
- **Finansial/claim**: normalisasi/standardisasi reward; *reward clipping* jika perlu; definisikan episode & reward dengan jelas.
- **Terminal handling**: untuk `done`$= 1$, set $y_j = r_{t+1}$ (tanpa bootstrapping).
- **Debugging**: mulai dari baseline DQN (mean), lalu ganti head menjadi kuantil.
- **Logging** (disarankan): pinball loss; mean vs median return; spread kuantil (IQR); performa kebijakan risk-neutral vs risk-aware.

Lab / Practice (QR-DQN on Claims or Stocks)

Dataset (pilih salah satu):

- *Simulated insurance claims*: Pareto / Lognormal (heavytail).
- *Stock data*: **sintetis meanreverting** (default di notebook). (Opsional: ganti dengan OHLC Yahoo Finance.)

Langkah:

1. Bentuk MDP sederhana: state, aksi (mis. retensi /alokasi diskret), reward (episode return).
2. Implementasikan **QR-DQN** dengan $N \in [32, 101]$ kuantil.
3. Visualisasikan distribusi prediksi per-aksi (quantile curves / fan chart).
4. Bandingkan aksi terpilih saat memakai **mean** vs **median/75th**.

Class Discussion (Week 6) Mean vs Quantile

Tujuan: memahami dampak target pembelajaran pada keputusan.

- Bandingkan strategi/aksi yang dipilih berdasarkan:
  1. **Expected return (mean)** dari kuantil yang dipelajari.
  2. **Selected quantile** (median dan/atau 75th percentile).
- Tunjukkan contoh state yang menghasilkan keputusan **berbeda**.
- Sajikan grafik: **distribusi/kuantil vs aksi terpilih**.

Catatan notebook: sel visualisasi sudah tersedia untuk *predicted quantile functions per action* serta *perbandingan kebijakan mean vs median/75th*. (Opsional: tambah fan chart jika ingin.)

Arahan Diskusi & Integrasi ke Manuskrip

Pertanyaan untuk didiskusikan:

- Kapan kebijakan **mean** dan **median/75th** memilih aksi yang berbeda? Apa ciri distribusi (skew/tail) pada state tersebut?
- Bagaimana **risiko ekor** tercermin pada kuantil tinggi (75th/90th)? Implikasi untuk konteks *insurance/finance*?
- Sensitivitas keputusan terhadap **jumlah kuantil** N dan **jadwal** ϵ?
- Apakah **spread kuantil** (mis. IQR) berkorelasi dengan performa/kerugian?

Catatan untuk manuskrip (ringkas hasil diskusi):

- *Pendahuluan*: motivasi heavy-tail & perlunya distributional RL.
- *Preliminary/Results*: contoh state nyata saat mean $\neq$ quantile, plus grafik.
- *Metodologi*: rasional memilih tujuan **median/kuantil** (risk-aware) alih-alih mean.

Distributional RL (Mention Only)

C51 (Bellemare et al., 2017)
$\rightarrow$ *Categorical distribution* dengan 51 atom tetap; project + KL loss.

IQN (Dabney et al., 2018)
$\rightarrow$ *Implicit Quantile Network*: sampling kuantil kontinu; menggeneralisasi QR-DQN.

FQF (Yang et al., 2019)
$\rightarrow$ *Fully Parameterized Quantile Function*: *learned* quantile fractions (adaptif).

References & Datasets

- Bellemare, Dabney, Munos (2017). *A Distributional Perspective on Reinforcement Learning*.
- Dabney et al. (2018). *Distributional RL with Quantile Regression*.
- Yang, Zhang, Lu (2019). *FQF: Fully Parameterized Quantile Function for DRL*.

- Simulated insurance claims: Pareto / Lognormal generators.
- Stock data: Yahoo Finance API.