防衛装備庁主催

第4回空戦AIチャレンジ チュートリアルセミナー

2025年8月13日

SIGNATE
Empowering Your Potential

戦闘場面とルールのおさらい

- 場面の概要
  - ✓ 4機の戦闘機編隊が1機の護衛対象機が撃墜されないように守りつつ相手の護衛対象機の撃墜を目指す。
- 戦闘のルール
  - ✓ いずれかの護衛対象機が撃墜されるか、制限時間が経過した場合に終了する。相手の護衛対象機を先に撃墜した陣営が勝者となり、時間切れとなった場合は引き分けとする。
  - ✓ 戦闘機の初期配置はランダムに決まり、東側(Blue)と西側(Red)はお互いに点対称となる。
    - ➢ ユース部門は高度10,000[m]で固定(2次元平面上を動く)。
  - ✓ 護衛対象機は、戦域中心から東に100[km]離れた直線上で南北方向に±60[km]の2点を結んだ線分の周囲を反時計回りに周回し続ける。
    - ➢ 周回時の速度は250[m/s]、高度は10,000[m]とする。また、周回軌道上の初期位相はランダムとする。
    - ➢ ただし、護衛対象機のMWSが相手の誘導弾を検出した場合に限り周回軌道を逸脱し、検出方向に背を向けて回避機動をとる。回避完了後は、速やかに元の周回軌道に戻る。
    - ➢ 東側(Blue)と西側(Red)はお互いに点対称で動く。
  - ✓ オープン部門のみ対戦ごとに各機の機体性能がランダム化される。

考えられるアプローチ

手法	メリット	デメリット
ルールベース	• 行動判断ロジックが明確にできる • 実装してから対戦シミュレーションする試行錯誤のスピードが速い	• 網羅性が低くなる • 弱点が見つけられやすい • 想定外への対応が困難
強化学習	• 行動判断ロジックを詳細化する必要がない • 考えつかないロジックを自動的に学習できる	• まともに戦闘できるようになるまで時間がかかる • 試行錯誤のスピードが遅い • 行動判断ロジックを考えなくてもよい代わりにモデル構築や報酬の設計などが難しい

- 対戦における戦闘場面などをよく理解して仮説を立て戦略を立てる
  - ✓ 目的は相手護衛対象機の撃墜。制限時間も考慮してどのような行動をとるべきかを考える。
  - ✓ ルールベースの中に相手の動きや誘導弾の位置を予測するなどの機械学習モデルを併用するなど。
  - ✓ ルールベース＋機械学習(強化学習含む)のハイブリッドな方法を考案してみるのもよい。
  - ✓ 初期行動判断モデルを参考にしたり、ブラッシュアップしてみる。
- ログ保存機能を活用して命令に対する動き方などを分析
  - ✓ 動画と合わせて確認できるとよりよい。
  - ✓ 初期行動判断モデルの動きを観察する。

強化学習の観点でのAgentの設計

操作方法の種別	概要	行動空間の広さ	メモリ消費量
分散型	1Agentにつき1機を操作すること	広くない	多い
中央集権型	1Agentで陣営全体を操作すること	広い	少ない

- 中央集権型の場合、行動空間の広さを緩和する対策がないと学習が進みにくい(サンプルはactor部分の重み共有により対策済)。
- 中央集権型の場合、既に撃墜された機体への行動も出力され続けるが、これは明らかに環境に影響を及ぼさないため、lossの計算で適切に処理する必要がある(サンプルはNN内でdetachにより対応済)。
- 分散型の場合、Observationの共通部分が重複するためメモリ消費量が増える。
- 分散型の場合、探索時やlossの計算時に味方の行動をどう扱うかが悩みどころとなる。

強化学習における報酬設計の基本的な考え方の一例

- 勝利時の最小値 ＞ 敗北時の最大値であることが望ましい。
- 勝敗に関わらず明確に避けてほしい動き（自滅など）に対しては大きな減点を与えることも有効。
- 護衛対象機を撃墜したときに大きく報酬を与える、逆に撃墜できなかった場合は大きく減点。
- 墜落や場外のペナルティはなるべく機体ごと（Agentごと）に与えた方が学習しやすい（分散型の場合）。
- 誘導弾は、高高度、高速度で発射するほど（お互いに）射程が延びるが、低高度ほど弾速が低下しやすく逃げやすいので一長一短。
  - ✓ 高度や速度に紐づいた報酬を入れる選択肢もあり。

サンプルルールベースモデル※の攻略

- 護衛対象機を優先して探索や攻撃を行うロジックが組み込まれていない。
  - ✓ 勝利条件である「護衛対象機の早期撃墜」にフォーカスする。
    - ➢ 複数機で一斉に敵護衛対象機へ集中攻撃。
  - ✓ 「防衛」には最低限だけリソースを割く。
    - ➢ 護衛対象機の近くに1～2機を配置+残りは機動的に攻撃へ回す、必要なら全機で攻撃など。
  - ✓ 必要最低限の回避行動のみで護衛対象機損失リスクを排除。
  - ✓ 護衛対象機が回避運動を取るタイミングを予測し、誘導弾が到達するよう追撃を重ねる。
- 戦域（可視化した際の長方形）の外に出ないようにするロジックが組み込まれている。
  - ✓ 今回は場外ペナルティが無く戦域はあくまで目安なので、外側から回り込む戦術もあり得る。
- 誘導弾を回避する際必ず降下する。
  - ✓ 下から撃つ方が効果的になる場面も多い。
- 通常時は一定高度を維持する。
  - ✓ 高度差があっても合わせてこない。

※サンプルとして提供されているアルゴリズムで、現在ベンチマークとして参戦中。投稿物を提出時にも検証処理において対戦を行い、それをもとに初期スコアが決定する

強化学習のアイデアに関するポイント

- 観測空間の改善
  - ✓ サンプルAgentの観測空間は主要な要素の例示という役割を持っているため、不要な情報も多い。
  - ✓ 例えば、残燃料の情報は今回のルールでは不要。ユース部門の場合は機体性能の情報も定数なので不要。
- 行動空間の改善
  - ✓ 例えばサンプルAgentの行動空間は両部門共通だが、ユース部門の場合は本来高度方向の行動が不要。
  - ✓ 場外防止や同時射撃数の制限を外すのも一案。
  - ✓ 大まかな離散行動(「前進」「右旋回」「左旋回」「ミサイル発射」等)か、連続制御(角速度・スロットル・仰角など)かを設計。
  - ✓ 「誰を攻撃するか」「いつミサイル発射するか」「防御/回避行動への切替」を複数エージェントで意思決定する設計。
- 対戦相手の追加
  - ✓ 前頁の弱点を改善したルールベースモデルの追加。
  - ✓ 特定の動きに特化したルールベースモデルの追加。
- 報酬関数の改善
  - ✓ 例えば、サンプルの勝敗報酬は戦闘時間によらず一定なので、設定値次第では護衛対象機を落とすよりも他の細々とした報酬を時間ギリギリまで稼いだ方が高くなってしまう。
- その他
  - ✓ いきなり4vs4+護衛目標で学習させると学習が進みにくい(Sparse Reward問題)ため、段階的に難易度を上昇させる。
  - ✓ 経験再生(Experience Replay)やPrioritized Experience Replayの利用。

最近の主な更新情報

- [2025/08/13] R7ContestSampleモジュールの一部機能(GenericTorchModelUtil.py)を更新
- [2025/08/08] 1回あたりの対戦ログのサイズ上限を0.08[GB]から0.01[GB]へ変更
- [2025/08/06] 対戦実行時にRCSスケールのランダム化が有効になっていなかったのを有効化した
- [2025/08/06] 垂直上昇時の挙動を修正した
- [2025/08/06] HandyRLサンプルの学習時におけるメモリリークの改善を行った
- [2025/08/06] DockerfileでlibPython3.10のインストールを追加で行うように修正した

ご清聴ありがとうございました

説明に使用したコードや資料は後日コンペティションサイトで公開します

SIGNATE
Empowering Your Potential

APPENDIX

サンプルAgentに関する参考資料

SIGNATE
Empowering Your Potential

サンプルAgentで使用可能な観測空間・行動空間

観測空間

・画像情報（$N_{ch} \times N_{lon} \times N_{lat}$）
　以下から選択してN_{ch}方向に重ねる
　・操作対象機の軌跡（1ch）
　・他の味方機の軌跡（1ch）
　・彼機の軌跡（1ch）
　・味方誘導弾の軌跡（1ch）
　・操作対象機のレーダ覆域（1ch）
　・他の味方機のレーダ覆域（1ch）
　・被我防衛ライン（1ch）
　・場外ライン（1ch）

・共通情報（D_c）
　以下から選択してD_c方向に重ねる
　・残り時間（1）

・操作対象機の情報（$N_p \times D_p$）
　各操作対象について以下から選択してD_p方向に重ねたものをN_p機分並べる
　・前回の行動
　　・左右旋回（1）
　　・上昇・下降（1）
　　・加減速（1）
　　・射撃目標（$1 + N_e$）
　・位置（3）
　・速度（4）
　・初期弾数（1）
　・残弾数（1）
　・姿勢（3）
　・角速度（3）
　・余剰燃料（1）
　・RCSスケール（1）
　・レーダ探知距離（1）
　・レーダ覆域（1）
　・最大速度倍率（1）
　・誘導弾推力倍率（1）

・他の味方機の情報（$N_f \times D_f$）
　各機について「操作対象機の情報」から「前回の行動」を除いたものをN_f機分並べる

・彼機の情報（$N_e \times D_e$）
　以下から選択してD_e方向に重ねたものをN_e機分並べる
　・位置（3）
　・速度（4）

・味方誘導弾の情報（$N_{fm} \times D_{fm}$）
　以下から選択してD_{fm}方向に重ねたものをN_{fm}発分並べる
　・位置（3）
　・速度（4）
　・飛翔時間（1）
　・目標との距離（1）
　・目標への誘導状態（3）
　・目標の位置（3）
　・目標の速度（4）
　・誘導弾推力倍率（1）

・彼側誘導弾の情報（$N_{em} \times D_{em}$）
　以下から選択してD_{em}方向に重ねたものをN_{em}発分並べる
　・観測点(我機)の位置（3）
　・検出方向（3）
　・方向変化率（3）

・彼我ペアの情報（$(N_p + N_f) \times N_e \times D_{fe}$）
　以下から選択してD_{fe}方向に重ねたものを$(N_p + N_f) \times N_e$組分並べる
　・我側から彼側への射程（3）
　・彼側から我側への射程（3）

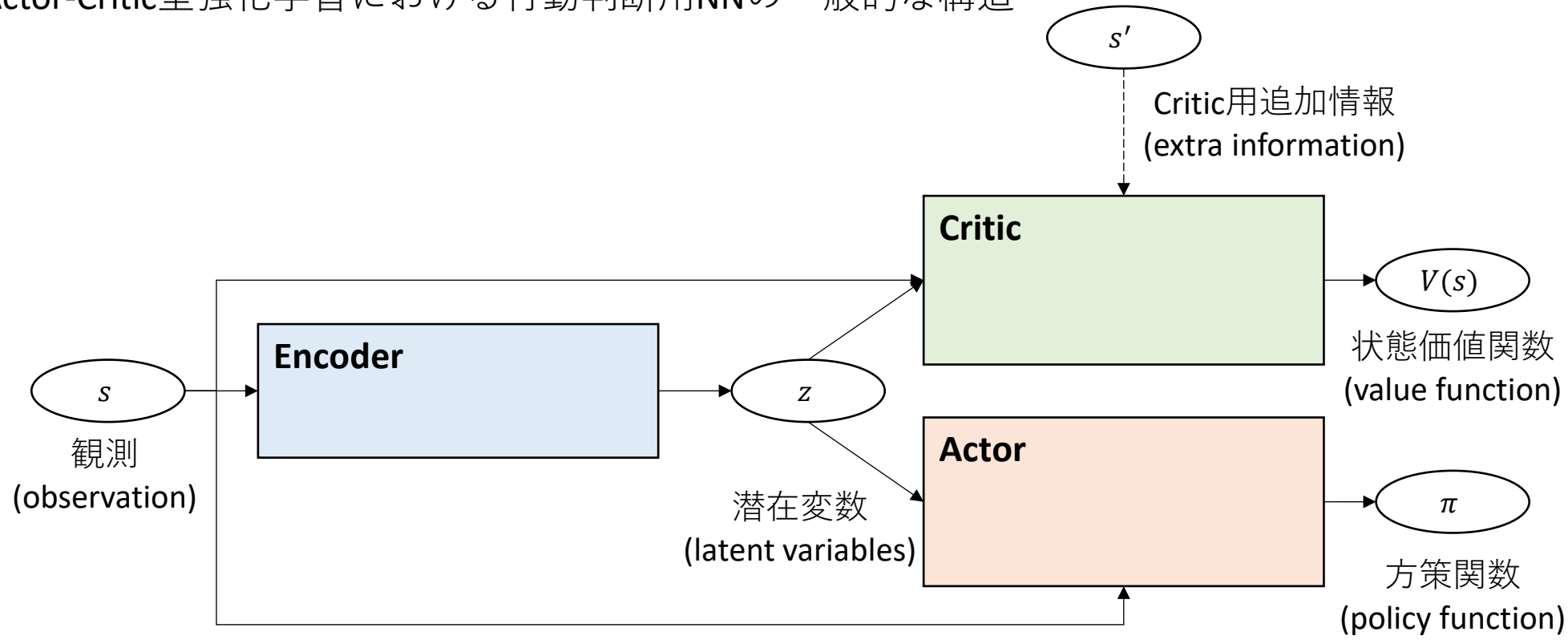
・要素の有無に関するマスク情報
　要素が有る部分は1、無い部分は0
　・操作対象機（N_p）
　・他の味方機（N_f）
　・彼機（N_e）
　・味方誘導弾（N_{fm}）
　・彼側誘導弾（N_{em}）
　・彼我ペア（$(N_p + N_f) \times N_e$）

・有効な行動に関するマスク情報
　有効な選択肢は1、無効な選択肢は0
　・左右旋回（D_{turn}）
　・上昇・下降（D_{pitch}）
　・加減速（D_{accel}）
　・射撃間隔（$D_{shot.int.}$）
　・射程条件（$D_{shot.thr.}$）
　・射撃目標（$1 + N_e$）

行動空間

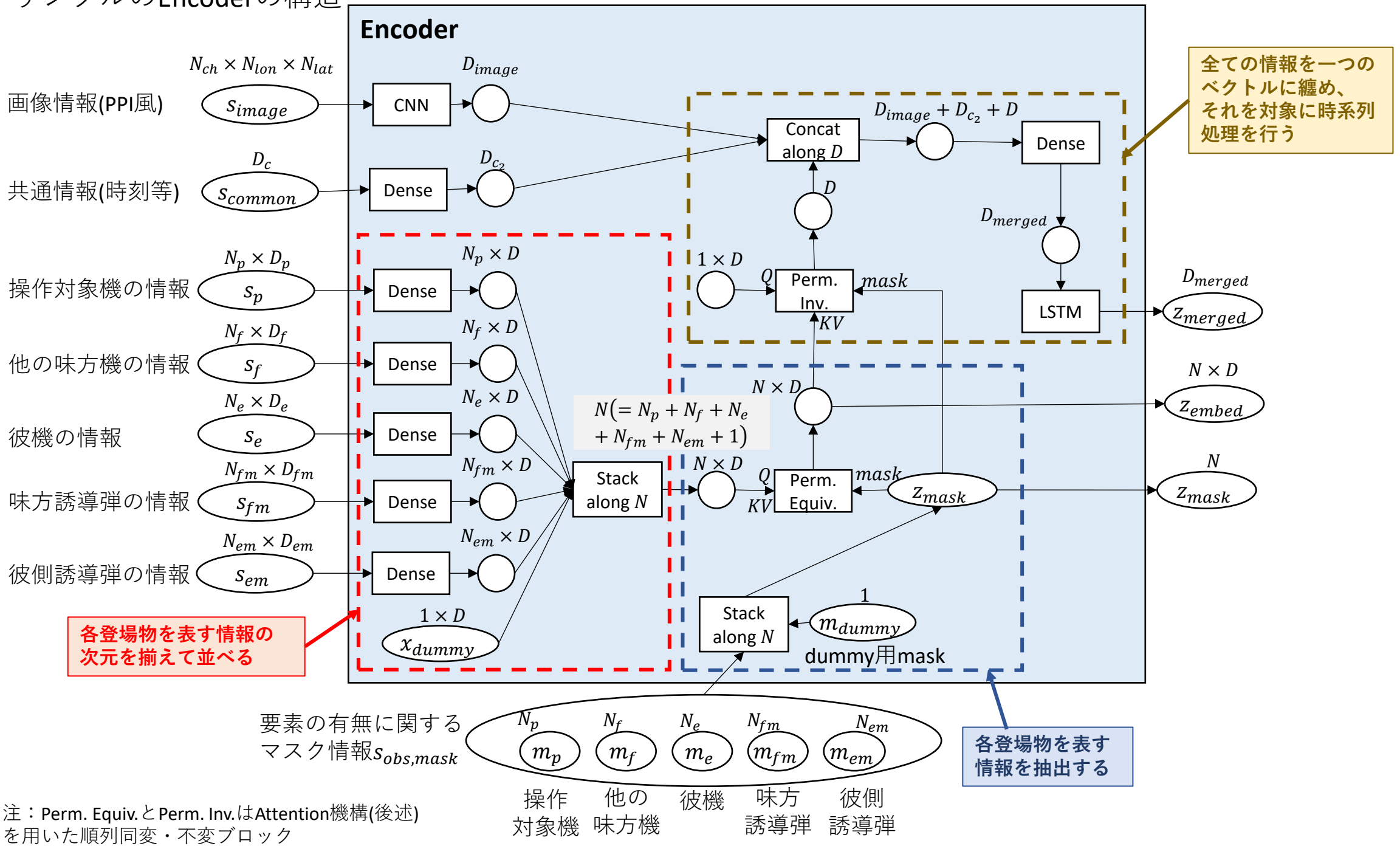
以下のDictをN_p機分並べたTuple(List)
{
　“turn”: 左右旋回（Discrete(D_{turn})），
　“pitch”: 上昇・下降（Discrete(D_{pitch})），
　“accel”: 加減速（Discrete(D_{accel})），
　“shotInterval”: 射撃間隔（Discrete($D_{shot.int.}$)），
　“shotThreshold”: 射程条件（Discrete($D_{shot.thr.}$)），
　“target”: 射撃目標（Discrete($1 + N_e$)）
}

Actor-Critic型強化学習における行動判断用NNの一般的な構造



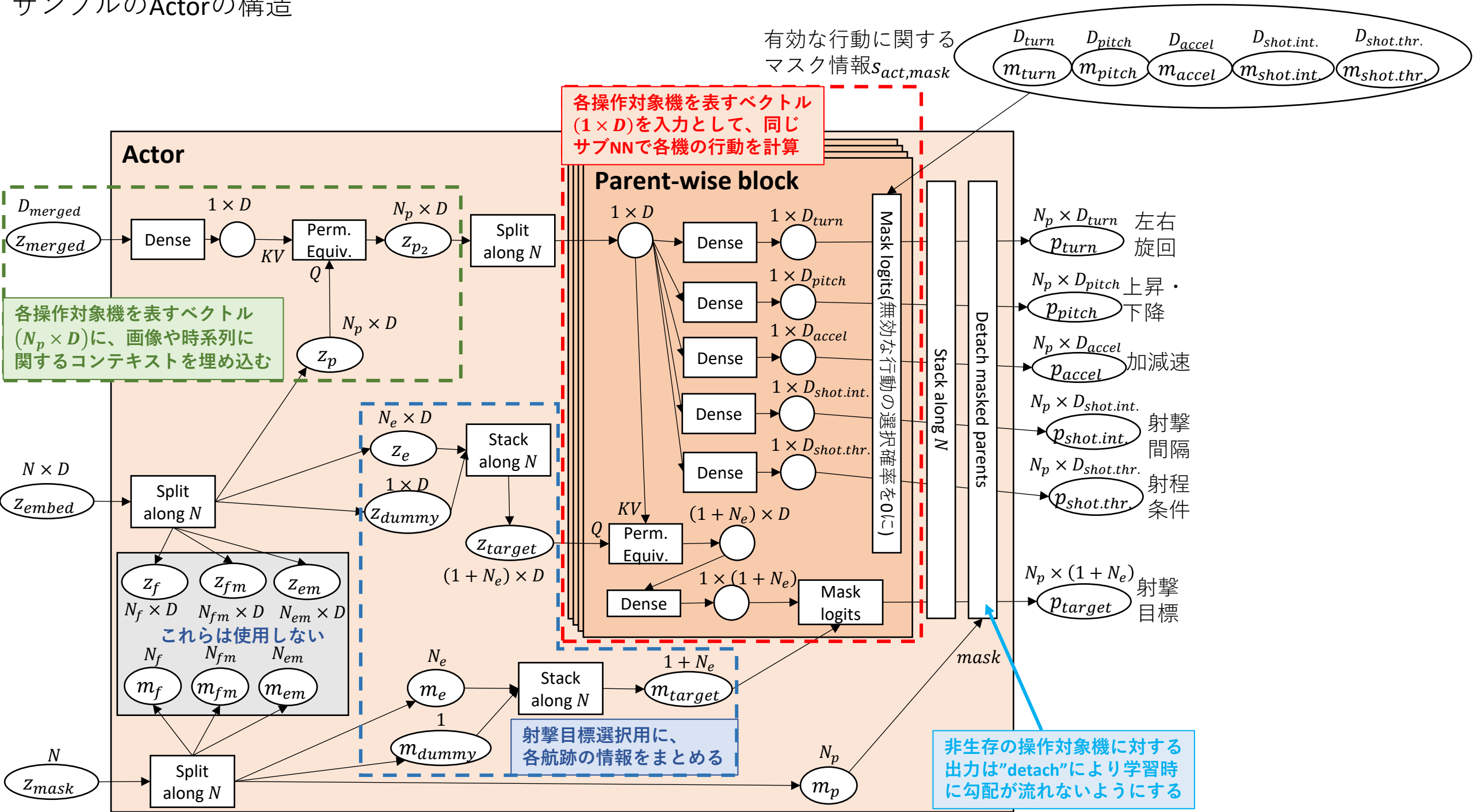
- Criticは学習時のみ使用されるので、本来は観測できない情報を追加で利用してもよい(上図のs'に相当。サンプルとしては提供されていないが、Callbackを使用したり、SimulationManagerのメンバを直接参照したりしてobservationを書き換える形になる)
- Actorの出力は行動そのものではなく、行動を選ぶための確率分布を表すパラメータ(例えばlogitや平均・標準偏差)

サンプルのEncoderの構造

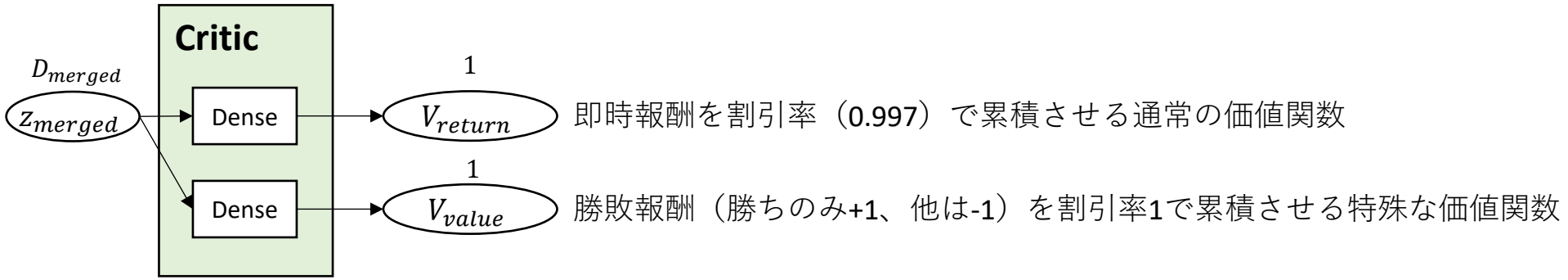


注：Perm. Equiv.とPerm. Inv.はAttention機構(後述)を用いた順列同変・不変ブロック

サンプルのActorの構造

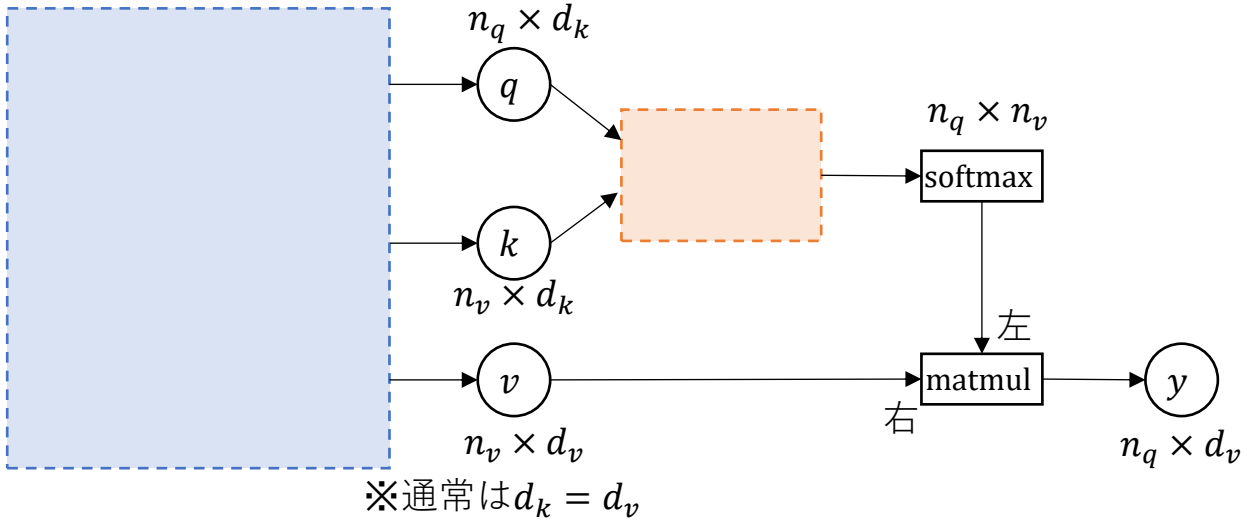


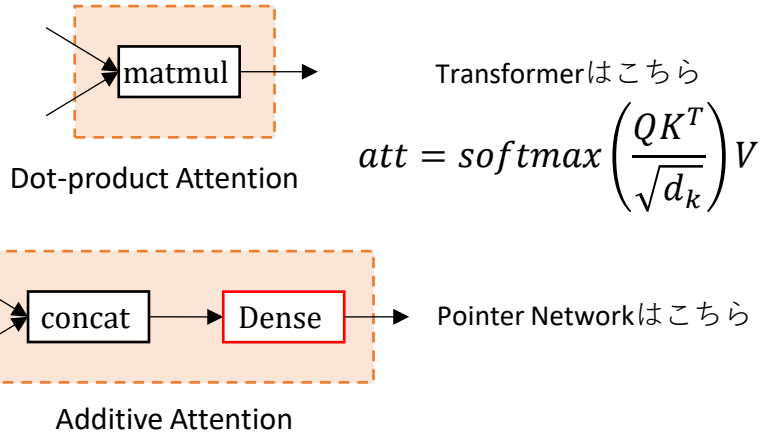
サンプルのCriticの構造



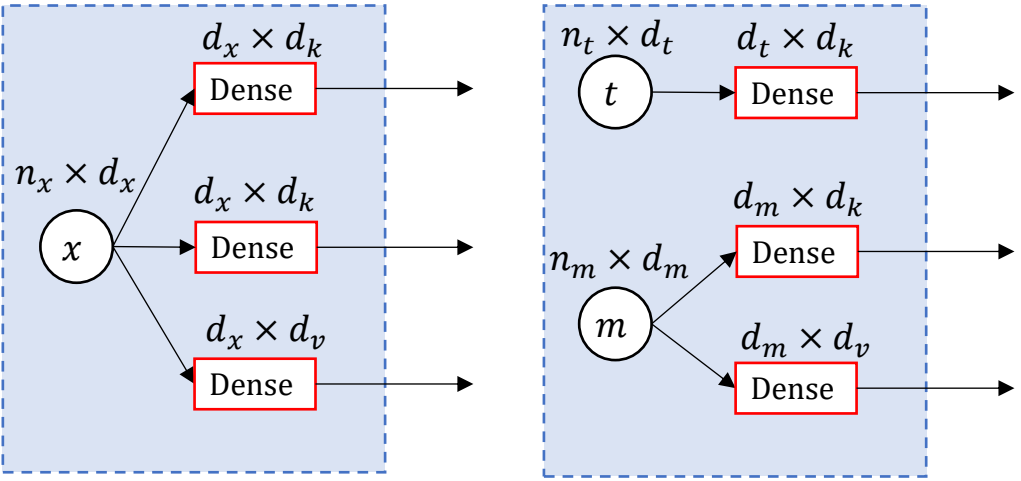
- Critic用追加情報を与える一例として、彼側目線でのobservationを与えて、それをCritic内部に設けた二つ目の **Encoder** で処理してz_{merged}に合流させるという方法がある。

（参考）Attentionの仕組み（基本構造）





※赤枠部分が学習対象パラメータ

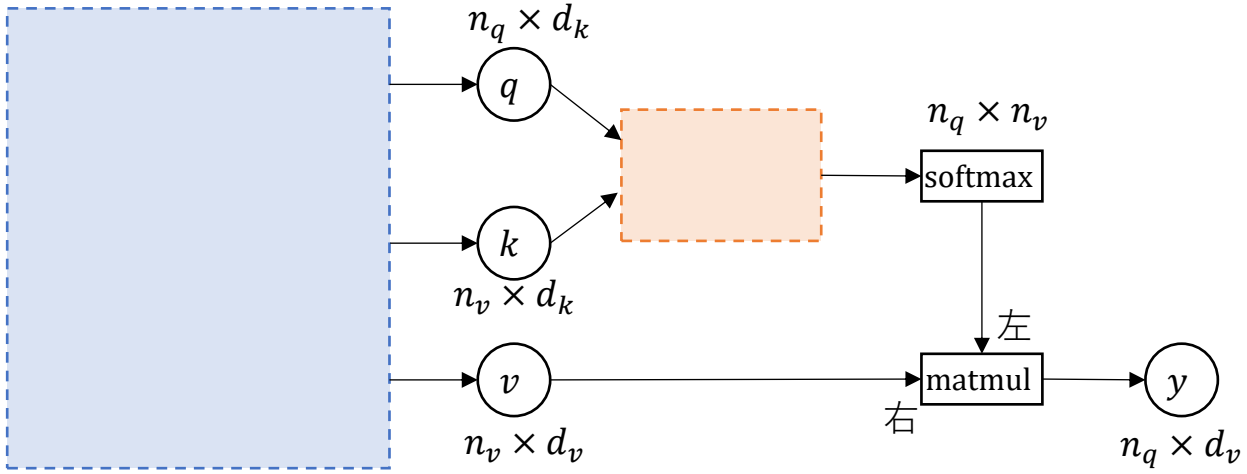


<Self Attention>
n_x個のd_x次元ベクトルについて、互いの関係性を抽出して変換するもの。出力もn_x個のd_x次元ベクトル

<Source Target Attention>
n_t個のd_t次元ベクトル(“target”)について、n_m個のd_m次元ベクトル(“memory”)から情報を抽出して変換するもの。
出力はn_t個のd_v次元ベクトル

<順列不変性と順列同変性>
Attentionの出力は、K,Vとして与えたシーケンスに対して順列不変性(permutation invariance)、Qとして与えたシーケンスに対して順列同変性(permutation equivariance)を持っている。

（参考）Attentionの仕組み（マスキング）



- の結果$(n_q \times n_v)$は各Qが各KVから情報を抽出する重みを表す。
- 情報を抽出したくない要素がある場合には、 の結果に $-\infty$を加えるマスク処理を行う。 Softmaxによってその部分の重みが0となるため、該当するKV要素を無視した出力が得られる。

1. 要素が存在しない空の情報を無視するためのマスク
   - 右記のように、KV側の要素が存在しない列に$-\infty$を加えればよい。
   - Q側の要素が存在しない場合については、Attentionとしては何もせず、後段の処理で不要な行を捨てることで対応することになる。

$$n_q \begin{pmatrix} 0 & 0 & -\infty & 0 \\ 0 & 0 & -\infty & 0 \\ 0 & 0 & -\infty & 0 \\ 0 & 0 & -\infty & 0 \end{pmatrix} \quad (n_v)$$

2. 未来の情報を参照させないためのマスク
   - 時系列データを与える場合には、未来の情報を用いてしまうと推論時の条件と不整合が生じることがある。そのような場合には、右記のように右上部分を$-\infty$にすればよい。

$$n_q \begin{pmatrix} 0 & -\infty & -\infty & -\infty \\ 0 & 0 & -\infty & -\infty \\ 0 & 0 & 0 & -\infty \\ 0 & 0 & 0 & 0 \end{pmatrix} \quad (n_v)$$