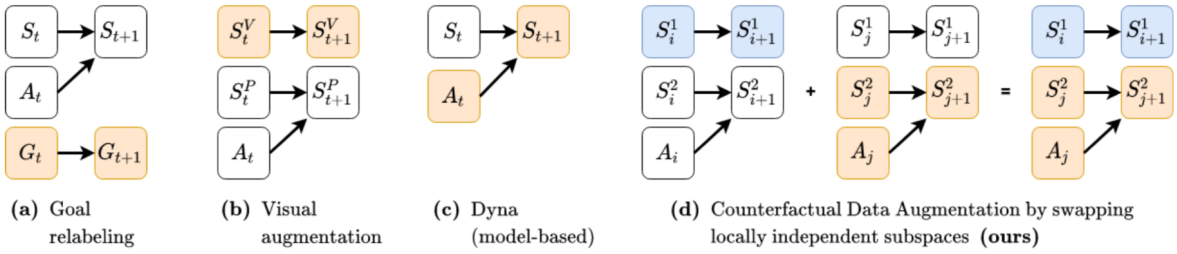
0.1 Counterfactual Data Augmentation (CoDA)

- dynamic processes involve a set of interacting subprocesses
- interactions between subprocesses are sparse
- introduces **local causal models** included from a global causal model by conditioning on a subset of state space
- local structures + experience replay to generate counterfactual experiences
- locally (during time between interaction) factor dynamics and model subprocesses independently
- underlying processes are difficult to model precisely
- factored subspaces of observed trajectory pairs are swapped
- CoDA is a data-augmentation strategy
- use attention to discover local causal structure
- improves sample-efficiency in batch-constrained and goal-constrained RL
- model time slice $(t, t+1)$ using a structural causal model (SCM)
- $\mathcal{M}_t = \langle V_t, U_t, \mathcal{F} \rangle$ with a DAG $\mathcal{G}$
- V_t is the set of state, action and next states, U_t is a set of noise variables and $\mathcal{F}$ is the set of structure equations that maps noise $\times$ current state to next state
- assume structural minimality allows us to think of edges in $\mathcal{G}$ as representing global causal dependence
- $P(S^i_{t+1}|S_t, A_t) = P(S^i_{t+1}|Pa(S^i_{t+1}))$ and S is independent of all nodes that isn't its parent
- often exists large subspace such that the next state is independent
- e.g. two-armed robot, there are many states such that the two arms are too far apart to influence each other
- consider a *local causal model* whose DAG is strictly sparser than the global DAG

Figure 1: Different instances of CoDA



(a) Goal relabeling **(b)** Visual augmentation **(c)** Dyna (model-based) **(d)** Counterfactual Data Augmentation by swapping locally independent subspaces **(ours)**

- Dyna augments real states with new actions and resamples the next state using a learned dynamics model

- can generate an exponential amount of data with CoDA (n independent samples from subspace $\mathcal{L}$ whose graph has m connected components $\implies n^m$ samples)
- CoDA can be used to mix and match data across timesteps

Figure 2: CoDA algorithm

Algorithm 1 Mask-based Counterfactual Data Augmentation (CoDA)

function CODA(transition `t1`, transition `t2`):
`s1, a1, s1' ← t1`
`s2, a2, s2' ← t2`
`m1, m2 ←` MASK`(s1, a1),` MASK`(s2, a2)`
`D1 ←` COMPONENTS`(m1)`
`D2 ←` COMPONENTS`(m2)`
`d ←` random sample from `(D1 ∩ D2)`
`s̃, ã, s̃' ←` copy`(s1, a1, s1')`
`s̃[d], ã[d], s̃'[d] ← s2[d], a2[d], s2'[d]`
`D̃ ←` COMPONENTS`(`MASK`(s̃, ã))`
return `(s̃, ã, s̃')` **if** `d ∈ D̃` **else** $\emptyset$

function MASK(state `s`, action `a`):
Returns $(n+m)\times(n)$ matrix indicating if the n next state components (columns) locally depend on the n state and m action components (rows).

function COMPONENTS(mask `m`):
Using the mask as the adjacency matrix for $\mathcal{G}^{\mathcal{L}}$ (with dummy columns for next action), finds the set of connected components $C = \{C_j\}$, and returns the set of independent components $D = \{\mathcal{G}_i = \bigcup_k \mathcal{C}_k^i \,|\, \mathcal{C}^i \subset \text{powerset}(C)\}$.

- specify a ratio between observed to counterfactual data to control for selection bias
- inferring local fatorization, how to approximate causal model
- use a global network mask (MADE) for autoregressive distribution modeling and GraN-DAG for causal discovery
- locally conditioned network mask by taking matrix product of locally conditioned layer masks
- MLP and single-head set transformer
- Inferring local factorization
  - SANDy (Sparse Attention Neural Dynamics)
  - it learns a function or mask that represents the adjacency matrix of the local causal graph
  - SANDy transformer performs between than MLP in AUC score
  - SANDy mixture is only sufficient for the simple synthetic MP environments, does not work for Spriteworld
  - the transformer has stronger inductve bias and more reliably infer local interaction patterns
- Some experiment details and notes:
  - working with the original TD3 codebase
  - mask function trained using 42k samples from random policy
  - increase agent batch size from 256 to 1000, more batch size allows agent to see more of its on-policy data in the face of many off-policy CoDA samples
  - batch-RL experiment in the Pong environment

  - trained a transformer to learn a mask function
  - goal-conditioned RL experiments on OpenAI gym FetchPush environment
  - these experiments use a hand-coded heuristic designed with domain knowledge
  - e.g. action is entangled with gripper and gripper + objects are disentangled when they are more than 10cm apart
- Other experimental insights:
  - tried augmenting the buffer by sampling from a dynamics model similar to model-based RL
  - good at next-state prediction, but fail to capture collisions and long-term dependencies

0.2 Differentiable Causal Discovery from Interventional Data (DCDI)

- learning a causal directed acyclic graph from data
- reformulates as continuous constrained optimization problem, solved using the augmented Lagrangian method
- this work proposes a NN model that can leverage interventional data
- using only observational data is challenging because the *faithfulness assumption* states that the true DAG is only identifiable up to a *Markov equivalence class*
- this can be improved by considering interventional data
- when observing enough interventions, the DAG is exactly identifiable
- there are score-based and constraint-based optimization methods, but they are computationally expensive and rely on parametric assumption
- perfect interventions remove the dependencies of a node on its parents, e.g. gene knockout in biology
- two classes of methods in causal structure learning
  - constraint-based methods
    - PC algorithm works with observational data
    - rely on conditional independence
    - COmbINE and HEJ support interventional data
    - JCI supports latent cofounders and can deal with interventions with unknown targets
  - score-based methods

    - formulate problem of estimating DAG G^* by optimizing a score function $\mathcal{S}$ over the space of DAGs

$$\mathcal{G} \in \underset{\mathcal{G} \in \text{DAG}}{\text{argmax}} \mathcal{S}(\mathcal{G})$$

    - common choice in the purely observational setting is the regularized ML

$$\mathcal{S}(\mathcal{G}) := \max_{\theta} \mathbb{E}_{X \sim P_X}[log f_\theta(X) - \lambda |\mathcal{G}|]$$

    - the space of DAGs is super-exponential in the number of nodes
    - these methods rely on greedy combinatorial search algorithms
    - e.g. GIES (adaptation of GES), assumes a *linear* gaussian model
    - CAM uses greedy search, nonlinear, additive noise model
  - hybrid-methods
    - combines both score-based and constraint, e.g. IGSP
- *weighted adjacency matrix* and acyclicity constraint: $\text{Tr} e^{A_\theta} - d = 0$
- shown that graph is acyclic iff the above constraint is satisfied
- problem is solved approximately using an augmented Lagrangian procedure
- performance is assessed by two metrics comparing estimated graph to the ground-truth graph
- *structural Hamming Distance* (SHD) = number of edges that are different between the two DAGs
- *structural interventional distance* (SID) how the two DAGs differ wrt to their causal inference statements
- DCDI relies on stochastic gradient descent and thus is scalable to graphs with hundreds of nodes even though the augmented Lagrangian procedure requires computing marix exponential which is $O(d^3)$
-