Due Date: April 15th, 2022, 11 p.m. EST

Instructions

- *For all questions, show your work!*
- *Please use a document preparation system such as LaTeX.*
- *Unless noted that questions are related, assume that notation and definitions for each question are self-contained and independent.*
- *Submit your answers electronically via Gradescope.*
- *TAs for this assignment are* ***Mohammad-Javad Bayazi*** *and* ***Naga Karthik****.*

This assignment covers mathematical and algorithmic techniques in the families of deep generative models and some of the recent self-supervised learning methods. Thus, we will explore Variational autoencoders (VAEs, Question 1), Autoregressive models (Question 2), Normalizing flows (Question 3), Generative adversarial networks (GANs, Question 4), and Self-supervised models (Question 5).

Question 1 (5-5-6)**.** Consider a latent variable model $p_\theta(\boldsymbol{x}) = \int p_\theta(\boldsymbol{x}|\boldsymbol{z})p(\boldsymbol{z})dz$, where $p(\boldsymbol{z}) = \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}_K)$ and $\boldsymbol{z} \in \mathbb{R}^K$.The encoder network (aka "recognition model") of variational autoencoder, $q_\phi(\boldsymbol{z}|\boldsymbol{x})$, is used to produce an approximate (variational) posterior distribution over latent variables $\boldsymbol{z}$ for any input datapoint $\boldsymbol{x}$.[1] This distribution is trained to match the true posterior by maximizing the evidence lower bound (ELBO):

$$\mathcal{L}(\theta, \phi; \boldsymbol{x}) = \mathbb{E}_{q_\phi}[\log p_\theta(\boldsymbol{x} \mid \boldsymbol{z})] - D_{\mathrm{KL}}(q_\phi(\boldsymbol{z} \mid \boldsymbol{x})||p(\boldsymbol{z}))$$

Let $\mathcal{Q}$ be the family of variational distributions with a feasible set of parameters $\mathcal{P}$; i.e. $\mathcal{Q} = \{q(\boldsymbol{z}; \pi) : \pi \in \mathcal{P}\}$; for example π can be mean and standard deviation of a normal distribution. We assume q_ϕ is parameterized by a neural network (with parameters ϕ) that outputs the parameters, $\pi_\phi(\boldsymbol{x})$, of the distribution $q \in \mathcal{Q}$, i.e. $q_\phi(\boldsymbol{z}|\boldsymbol{x}) := q(\boldsymbol{z}; \pi_\phi(\boldsymbol{x}))$.

1.1 Show that maximizing the expected complete data log likelihood (ECLL)

$$\mathbb{E}_{q(\boldsymbol{z}|\boldsymbol{x})}[\log p_\theta(\boldsymbol{x}|\boldsymbol{z})p(\boldsymbol{z})]$$

for a fixed $q(\boldsymbol{z}|\boldsymbol{x})$, wrt the model parameter θ, is equivalent to maximizing

$$\log p_\theta(\boldsymbol{x}) - D_{\mathrm{KL}}(q(\boldsymbol{z}|\boldsymbol{x})||p_\theta(\boldsymbol{z}|\boldsymbol{x}))$$

This means the maximizer of the ECLL coincides with that of the marginal likelihood only if $q(\boldsymbol{z}|\boldsymbol{x})$ perfectly matches $p(\boldsymbol{z}|\boldsymbol{x})$.

answer:

[1] Using a recognition model in this way is known as "amortized inference" ; this can be contrasted with traditional variational inference approaches (see, e.g., Chapter 10 of Bishop's *Pattern Recognition an Machine Learning*), which fit a variational posterior independently for each new datapoint.

As we know $p(x|z) = \frac{p(z|x)p(x)}{p(z)}$, we can write:

$$\begin{aligned}
\mathbb{E}_{q(\boldsymbol{z}|\boldsymbol{x})}[\log p_\theta(\boldsymbol{x}|\boldsymbol{z})p(\boldsymbol{z})] &= \mathbb{E}_{q(\boldsymbol{z}|\boldsymbol{x})}[\log p_\theta(\boldsymbol{z}|\boldsymbol{x})p(\boldsymbol{x})] \\
&= \mathbb{E}_{q(\boldsymbol{z}|\boldsymbol{x})}[\log[\frac{p_\theta(\boldsymbol{z}|\boldsymbol{x})}{q_\phi(\boldsymbol{z}|\boldsymbol{x})}p(\boldsymbol{x})q_\phi(\boldsymbol{z}|\boldsymbol{x})]] \\
&= \mathbb{E}_{q(\boldsymbol{z}|\boldsymbol{x})}[\log p(\boldsymbol{x})q_\phi(\boldsymbol{z}|\boldsymbol{x})] - \mathbb{E}_{q(\boldsymbol{z}|\boldsymbol{x})}[\log \frac{q_\phi(\boldsymbol{z}|\boldsymbol{x})}{p_\theta(\boldsymbol{z}|\boldsymbol{x})}] \\
&= \mathbb{E}_{q(\boldsymbol{z}|\boldsymbol{x})}[\log p_\theta(\boldsymbol{x})] + \mathbb{E}_{q(\boldsymbol{z}|\boldsymbol{x})}[\log q_\phi(\boldsymbol{z}|\boldsymbol{x})] - \mathbb{E}_{q(\boldsymbol{z}|\boldsymbol{x})}[\log \frac{q_\phi(\boldsymbol{z}|\boldsymbol{x})}{p_\theta(\boldsymbol{z}|\boldsymbol{x})}] \\
&= \log p_\theta(\boldsymbol{x}) - D_{KL}(q_\phi(\boldsymbol{z}|\boldsymbol{x})||p_\theta(\boldsymbol{z}|\boldsymbol{x})) + Constant
\end{aligned}$$

Note 1: since $p_(\boldsymbol{x})$ is not dependent on z we can take it out from expectation.
Note 2: derivative of $\mathbb{E}_{q(\boldsymbol{z}|\boldsymbol{x})}$ respect to θ is zero. So we can eliminate it for computing the maximizing expectation respect to θ.

1.2 Consider a finite training set $\{\boldsymbol{x}_i : i \in \{1, ..., n\}\}$, n being the size the training data. Let ϕ^* be the maximizer $\arg\max_\phi \sum_{i=1}^n \mathcal{L}(\theta, \phi; \boldsymbol{x}_i)$ with θ fixed. In addition, for each $\boldsymbol{x}_i$ let $q_i \in \mathcal{Q}$ be an "instance-dependent" variational distribution, and denote by q_i^* the maximizer of the corresponding ELBO. Compare $D_{\text{KL}}(q_{\phi^*}(\boldsymbol{z}|\boldsymbol{x}_i)||p_\theta(\boldsymbol{z}|\boldsymbol{x}_i))$ and $D_{\text{KL}}(q_i^*(\boldsymbol{z})||p_\theta(\boldsymbol{z}|\boldsymbol{x}_i))$. Which one is bigger ?

answer:

We can consider the inference gap as:

$$\log p_\theta(x_i) - \mathcal{L}[q_{\phi^*}(z \mid x_i) = \log p_\theta(x_i) - \mathcal{L}[q_i^*(z)] + \mathcal{L}[q_i^*(z)] - \mathcal{L}[q_{\phi^*}(z \mid x_i)] \ (1)$$

Also we know that :

$$D_{KL}(q(z \mid x) \| p_\theta(z \mid x)) = \log p_\theta(x) - \mathcal{L}[q(z \mid x)]$$

So we can rewrite the equation (1) based on the previous equation and we will have:

$$D_{KL}(q_{\phi^*}(z \mid x_i) \| p_\theta(z \mid x_i)) = D_{KL}(q_i^*(z) \| p_\theta(z \mid x_i)) + (\mathcal{L}[q_i^*(z)] - \mathcal{L}[q_{\phi^*}(z \mid x_i)])$$

$\mathcal{L}[q_i^*(z)] \geqslant \mathcal{L}[q_{\phi^*}(z \mid x_i)]$ since $q_i^\star$ is the maximizer of the corresponding ELBO so we will have:

$$D_{KL}(q_{\phi^*}(z \mid x_i) \| p_\theta(z \mid x_i)) \geqslant D_{KL}(q_i^*(z) \| p_\theta(z \mid x_i))$$

1.3 Following the previous question, compare the two approaches in the second subquestion

(a) in terms of bias of estimating the marginal likelihood via the ELBO, in the best case scenario (i.e. when both approaches are optimal within the respective families)

answer: In estimating the marginal likelihod via the ELBO the bias comes from the KL-divergence so based on the relation in the Question 1.2, we will have:

$$D_{KL}\left(q_{\phi^*}\left(z \mid x_i\right) \| p_\theta\left(z \mid x_i\right)\right) \geqslant D_{KL}\left(q_i^*(z) \| p_\theta\left(z \mid x_i\right)\right)$$

Thus the bias in $q_\phi^\star(z|x_i)$ is greater than $q_i^\star(z)$

(b) from the computational point of view (efficiency)

answer: In the view of efficiency $q_\phi^\star$ is more efficient compare to $q_i^\star$. Because for each of training data we should compute $q_i^\star$. So it takes n times more computation because we have n samples.

(c) in terms of memory (storage of parameters)

answer:

$q_i^\star$ needs $O(\boldsymbol{n})$ memory(n times more memory that $q_\phi^\star$) because it will store the variational parameter for each training data.

Question 2 (5-5-5-5)**.** One way to enforce autoregressive conditioning is via masking the weight parameters.[2] Consider a two-hidden-layer convolutional neural network without kernel flipping, with kernel size 3 × 3 and padding size 1 on each border (so that an input feature map of size 5 × 5 is convolved into a 5 × 5 output). Define mask of type A and mask of type B as

$$(\boldsymbol{M}^A)_{::ij} := \begin{cases} 1 & \text{if } i < 2 \\ 1 & \text{if } i = 2 \text{ and } j < 2 \\ 0 & \text{elsewhere} \end{cases} \qquad (\boldsymbol{M}^B)_{::ij} := \begin{cases} 1 & \text{if } i < 2 \\ 1 & \text{if } i = 2 \text{ and } j \leq 2 \\ 0 & \text{elsewhere} \end{cases}$$

where the index starts from 1. Masking is achieved by multiplying the kernel with the binary mask (elementwise). Specify the receptive field of the output pixel that corresponds to the third row and the third column (index 33 of Figure 1 (Left)) in each of the following 4 cases:

11	12	13	14	15
21	22	23	24	25
31	32	33	34	35
41	42	43	44	45
51	52	53	54	55

11	**12**	13	14	15
21	22	23	24	25
31	32	33	34	35
41	42	43	44	45
51	52	53	54	55

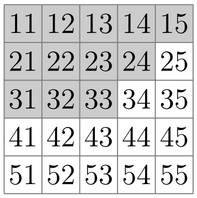
FIGURE 1 – (Left) 5 × 5 convolutional feature map. (Right) Template answer.

1. If we use $\boldsymbol{M}^A$ for the first layer and $\boldsymbol{M}^A$ for the second layer.

11	**12**	**13**	**14**	**15**
21	**22**	**23**	**24**	25
31	**32**	33	34	35
41	42	43	44	45
51	52	53	54	55

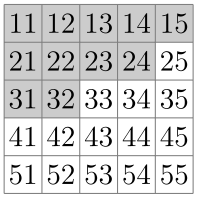
[2] An example of this is the use of masking in the Transformer architecture.

2. If we use $\boldsymbol{M}^A$ for the first layer and $\boldsymbol{M}^B$ for the second layer.



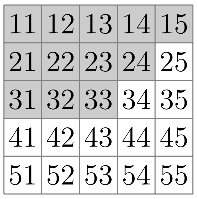
11	12	13	14	15
21	22	23	24	25
31	32	33	34	35
41	42	43	44	45
51	52	53	54	55

3. If we use $\boldsymbol{M}^B$ for the first layer and $\boldsymbol{M}^A$ for the second layer.



11	12	13	14	15
21	22	23	24	25
31	32	33	34	35
41	42	43	44	45
51	52	53	54	55

4. If we use $\boldsymbol{M}^B$ for the first layer and $\boldsymbol{M}^B$ for the second layer.



11	12	13	14	15
21	22	23	24	25
31	32	33	34	35
41	42	43	44	45
51	52	53	54	55

Your answer should look like Figure 1 (Right).

Question 3 (6-4)**.** In this question, we study some properties of normalizing flows. Let $X \sim P_X$ and $U \sim P_U$ be, respectively, the distribution of the data and a base distribution (e.g. an isotropic gaussian). We define a normalizing flow as $F : \mathcal{U} \to \mathcal{X}$ parametrized by $\boldsymbol{\theta}$. Starting with P_U and then applying F will induce a new distribution $P_{F(U)}$ (used to match P_X). Since normalizing flows are invertible, we can also consider the distribution $P_{F^{-1}(X)}$.

However, some flows, like planar flows, are not easily invertible in practice. If we use P_U as the base distribution, we can only sample from the flow but not evaluate the likelihood. Alternatively, if we use P_X as the base distribution, we can evaluate the likelihood, but we will not be able to sample.

3.1 Show that $D_{KL}[P_X||P_{F(U)}] = D_{KL}[P_{F^{-1}(X)}||P_U]$. In other words, the forward KL divergence between the data distribution and its approximation can be expressed as the reverse KL divergence between the base distribution and its approximation.

answer:

$$\begin{aligned}
D_{kL}\left[P_x \| P_{F(u)}\right] &= E_{P_x(x)}\left[\log P_x(x) - \log P_{F(u)}(x)\right] \\
&= E_{p_x(x)}\left[\log p_x(x) - \log p_u\left(f^{-1}(x)\right) - \log|\det J_{F^{-1}}(x)|\right] \\
&= E_{P_{F^{-1}(x)}(u)}\left[\log P_x(F(u)) - \log P_u(u) - \log|\det J_F(u)|\right] \\
&= E_{P_{F^{-1}(x)}(u)}\left[\log P_{F^{-1}(x)}(u) - \log P_u(u)\right] \\
&= D_{kL}\left[P_{F^{-1}(x)}(u) \| P_u\right]
\end{aligned}$$

3.2 Suppose two scenario: 1) you don't have samples from $p_X(\boldsymbol{x})$, but you can evaluate $p_X(\boldsymbol{x})$, 2) you have samples from $p_X(\boldsymbol{x})$, but you cannot evaluate $p_X(\boldsymbol{x})$. For each scenario, specify if you would use the forward KL divergence $D_{KL}[P_X||P_{F(U)}]$ or the reverse KL divergence $D_{KL}[P_{F(U)}||P_X]$ as the objective to optimize. Justify your answer.

answer:

1)

for the first scenario, because I can not sample from $p_x(x)$, I should use:

$$D_{KL}[P_{F(U)}||P_x] = \mathbb{E}_{P_U(u)}[\log P_U(u) - \log P_{F_{-1}(\boldsymbol{x})}(u)]$$

based on the question since Normalizing flows are invertible, we can consider the distribution $P_{F_{-1}(\boldsymbol{x})}$ and also we can compute P_U either.

2) for the second scenario, because we can sample from $P_x(x)$ we should use:

$$D_{KL}[P_x||P_{F(U)}] = -\mathbb{E}_{P_x(\boldsymbol{x})}[\log P_{F(U)}(x)] + constant$$

So in the above equation if we can sample from $P_x(\boldsymbol{x})$ we can approximate $\mathbb{E}_{P_x(\boldsymbol{x})}[\log P_{F(U)}(x)]$ and the constant$(=E_{P_x(\boldsymbol{x})}[logP_x(\boldsymbol{x})])$ has no impact in optimization task for finding the F function. Because it's derivative becomes zero.

Question 4 (4-3-6). In this question, we are concerned with analyzing the training dynamics of GANs. Consider the following value function

$$V(d, g) = dg \tag{2}$$

with $g \in \mathbb{R}$ and $d \in \mathbb{R}$. We will use this simple example to study the training dynamics of GANs.

1. Consider gradient descent/ascent with learning rate α as the optimization procedure to iteratively minimize $V(d, g)$ w.r.t. g and maximize $V(d, g)$ w.r.t. d. We will apply the gradient descent/ascent to update g and d simultaneously. What is the update rule of g and d? Write your answer in the following form

$$[d_{k+1}, g_{k+1}]^\top = A[d_k, g_k]^\top$$

where A is a 2×2 matrix ; i.e. specify the value of A.

answer:

$$d_{k+1} = d_k - \alpha\frac{\partial V(d,g)}{\partial d} = d_k - \alpha g$$
$$g_{k+1} = g_k - \alpha\frac{\partial V(d,g)}{\partial g} = g_k - \alpha d$$

So we can infer that:

$$A = \begin{bmatrix} 1 & \alpha \\ -\alpha & 1 \end{bmatrix}$$

.

2. The optimization procedure you found in 4.1 characterizes a map which has a stationary point [3], what are the coordinates of the stationary points ?

   answer:

   For the stationary point we should put all the partial derivates of V(d,g) equal to zero so: $\frac{\partial V(d,g)}{\partial g} = 0$ and $\frac{\partial V(d,g)}{\partial d} = 0$
   Thus the stationary points are: g=0, d=0.

3. Analyze the eigenvalues of A and predict what will happen to d and g as you update them jointly. In other word, predict the behaviour of d_k and g_k as $k \to \infty$.

   answer:

   For calculating the eigenvalue of A:

$$\det(\boldsymbol{A} - \lambda I) = 0$$

$$\det\left(\begin{bmatrix} 1-\lambda & \alpha \\ -\alpha & 1-\lambda \end{bmatrix}\right) = 0$$
$$(1-\lambda)^2 + \alpha^2 = 0$$
$$\lambda = 1 \pm i\alpha$$

   by computing determinant matrix A we will have:

$$det(A) = \lambda_1 * \lambda_2 = (1+i\alpha)(1-i\alpha) = 1+\alpha^2$$

   The determinant of a matrix is the factor by which areas are scaled by this matrix. So here the determinant of Matrix A is bigger than one, thus it d_k and g_k will diverge when $k \to \infty$
   Moreover in different view we can say that: If the eigenvalues of the system are complex with the positive real part the solution(d_{k+1}, g_{k+1}) will grow very large when $k \to \infty$. So in the complex eigenvalue since the real part is positive the trajectories will spiral out from the origin, so it will never converge to optimum as $K \to \infty$.

Question 5 (4-2-8-4-2)**.** In this question, we will see why stop-gradient is critical for non-contrastive SSL methods like SimSiam and BYOL. We will show that removing stop-gradient results in collapsed representations, using the dynamics of SimSiam as our running example.

Consider a two-layer linear SimSiam model with the time-varying weight matrices given by $W(t) \in \mathbb{R}^{n_2 \times n_1}$ and $W_p(t) \in \mathbb{R}^{n_2 \times n_2}$. Note that $W(t)$ corresponds to the weights of the online **and** the target network, while $W_p(t)$ denotes the weights of the predictor. Let $\boldsymbol{x} \in \mathbb{R}^{n_1}$ be an input datapoint and $\boldsymbol{x}_1, \boldsymbol{x}_2$ be the two augmented versions of the input $\boldsymbol{x}$. Also note that in some instances, the dependence on time (t) is omitted for notational simplicity, and the weight matrices are referred to as W and W_p.

[3] A stationary point is a point on the surface of the graph (of the function) where all its partial derivatives are zero (equivalently, the gradient is zero). Source: https://en.wikipedia.org/wiki/Stationary_point

Let $\boldsymbol{f}_1 = W\boldsymbol{x}_1$ be the online representation of $\boldsymbol{x}_1$ and $\boldsymbol{f}_2 = W\boldsymbol{x}_2$ be the target representation of $\boldsymbol{x}_2$. The learning dynamics of W and W_p can be obtained by minimizing SimSiam's objective function as shown below:

$$J(W, W_p) = \frac{1}{2}\mathbb{E}_{x_1,x_2}\left[\|W_p\boldsymbol{f}_1 - \text{Stop-Grad}(\boldsymbol{f}_2)\|_2^2\right]. \tag{3}$$

5.1 Show (with proof) that the above objective can be simplified to:

$$J(W, W_p) = \frac{1}{2}\left[\text{tr}(W_p^{\text{T}}W_pF_1) - \text{tr}(W_pF_{12}) - \text{tr}(F_{12}W_p) + \text{tr}(F_2))\right], \tag{4}$$

where $F_1 = \mathbb{E}\left[\boldsymbol{f}_1\boldsymbol{f}_1^{\text{T}}\right] = W(X+X')W^{\text{T}}$, $F_2 = \mathbb{E}\left[\boldsymbol{f}_2\boldsymbol{f}_2^{\text{T}}\right] = W(X+X')W^{\text{T}}$, and $F_{12} = F_{21} = \mathbb{E}\left[\boldsymbol{f}_1\boldsymbol{f}_2^{\text{T}}\right] = WXW^{\text{T}}$. Here, X is the average augmented view of a datapoint $\boldsymbol{x}$ and X' is the covariance matrix of augmented views $\boldsymbol{x}'$ conditioned on $\boldsymbol{x}$ and then averaged over the data $\boldsymbol{x}$, and tr is the Trace operation [4].

answer:

For solving the question 5, I used the help of this paper.

we know that a real number can be thought of as 1x1 matrix and it's trace is itself and also trace has the property of linear mapping thus:

$$\begin{aligned}(W_pf_1 - f_2)^T(W_pf_1 - f_2) &= \frac{1}{2}[tr(E_{x_1,x_2}[f_1^TW_P^TW_pf_1 - f_2^TW_pf_1 - f_1^TW_p^Tf_2 + f_2^Tf_2])] \\ &= \frac{1}{2}[tr(\mathbb{E}[f_1^TW_p^TW_pf_1]) - tr(E[f_2^TW_pf_1] - tr(\mathbb{E}[f_1^TW_p^Tf_2]) + tr(\mathbb{E}(f_2^Tf_2))]\end{aligned}$$

Now by using the **linearity** of the trace operator and **cyclic property** of the trace we can write:

$$\begin{aligned}J(W, W_p) &= \frac{1}{2}[\mathbb{E}[tr(f_1^TW_p^TW_pf_1)] - E[tr(f_2^TW_pf_1)] - \mathbb{E}[tr(f_1^TW_p^Tf_2)] + \mathbb{E}(tr(f_2^Tf_2)] \\ &= \frac{1}{2}[\mathbb{E}[tr(W_p^TW_pf_1f_1^T)] - E[tr(W_pF_1f_2^T)] - \mathbb{E}[tr(W_p^Tf_2f_1^T)] + \mathbb{E}(tr(f_2^Tf_2)] \\ &= \frac{1}{2}[\mathbb{E}[tr(W_p^TW_pf_1f_1^T)] - E[tr(W_pf_1f_2^T)] - \mathbb{E}[tr(f_2f_1^TW_p^T)] + \mathbb{E}(tr(f_2^Tf_2)] \\ &= \frac{1}{2}[tr(W_p^TW_pF_1) - tr(W_pF_{12}) - tr(F_{12}W_p^T) + tr(F_2)]\end{aligned}$$

5.2 Based on the above expression for $J(W, W_p)$, find the gradient update for W_p (the predictor network), denoting it as $\dot{W}_p$. In other words, obtain an expression for $\dot{W}_p = -\frac{\partial J}{\partial W_p}$ (the derivative of the objective function w.r.t the parameters W_p).

answer:

$$\dot{W}_p = -[\frac{1}{2}2W_pF_1 - \frac{1}{2}F_{12} - \frac{1}{2}F_{12}] = -W_pF_1 + F_{12}^T$$

[4] https://en.wikipedia.org/wiki/Trace_(linear_algebra).

5.3 Consider the case when the Stop-Grad is removed. The gradient of the objective function $J(W, W_p)$ w.r.t the parameters W i.e. $\dot{W}(t) = -\frac{\partial J}{\partial W(t)}$, is given by:

$$\dot{W}(t) = \frac{d}{dt}\text{vec}(W(t)) = -H(t)\text{vec}(W(t)),$$

where $H(t)$ is a time-varying positive semi-definite matrix defined as

$$H(t) = X' \otimes \left(W_p(t)^{\mathrm{T}} W_p(t) + I_{n_2}\right) + X \otimes \left(\tilde{W}_p(t)^{\mathrm{T}} \tilde{W}_p(t)\right).$$

Here, $\otimes$ is the Kronecker product [5] , $\tilde{W}_p(t) = (W_p(t) - I_{n_2})$, and "vec(W)" refers to the *vectorization* of a matrix W [6]. For simplicity, we are not taking weight decay into account here [7].

If the minimal eigenvalue $\lambda_{min}(H(t))$ is bounded away from zero, i.e. $\inf_{t\geq 0} \lambda_{min}(H(t)) \geq \lambda_0 > 0$, then **prove that** $W(t) \to 0$.

Note: In order to prove the above question, the following property must be used:

For a time-varying positive definite matrix $H(t)$ whose minimal eigenvalues are bounded away from 0, the dynamics shown below:

$$\frac{d}{dt}\boldsymbol{w}(t) = -H(t)\boldsymbol{w}(t),$$

satisfies the constraint $\|\boldsymbol{w}(t)\|_2 = e^{-\lambda_0 t}\|\boldsymbol{w}(0)\|_2$, implying that $\boldsymbol{w}(t) \to 0$.

answer:

for the first step we will compute the $\dot{W}$:

$$\begin{aligned}
\dot{W} &= -\frac{1}{2}[W_p^T W_p^T \frac{\partial F_1}{\partial W} - W_p \frac{\partial F_{12}}{\partial W} - W_p \frac{\partial F_{12}}{\partial W} + \frac{\partial F_2}{\partial W}] \\
&= -W_p^T W_p W(X + X') + (W_p^T + W_p)WX - W(X + X') \\
&= -W_p^T W_p WX + W_p^T W_p WX' + Wp^T WX + W_p WX - WX - WX' \\
&= -(W_p^T W_p + I)WX' - (W_p^T W_p - W_p^T - W_p + I)WX \\
&= -(W_p^T W_p + I)WX' - (W_p - I)^T (W_p - I)WX \\
&= -(W_p^T W_p + I)WX' - \tilde{W_p}^T \tilde{W}_p WX
\end{aligned}$$

$$\dot{W} = -(W_p^T W_p + I)WX' - \tilde{W_p}^T \tilde{W}_p WX \tag{5}$$

Based on the Kronecker product we have: $vec(AXB) = (B^T \otimes A)vec(X)$ so we can assume that $(W_p^T W_p + I) = A$ and $W = X$ and $X' = B$

also for the second part we can assume that: $\tilde{W_p}^T \tilde{W}_p = A$ and $W = X$ and $X = B$

So we can convert the first part of equation (5) as:

[5] For more information, see https://en.wikipedia.org/wiki/Kronecker_product#Matrix_equations

[6] Also known as the "vec trick", it is obtained by stacking all the columns of a matrix A into a single vector.

[7] Although omitted here, it must be noted that having weight decay is important. It has also been shown that, in practice, weight decay leads to stable learning.

$$\begin{aligned}\dot{W} &= -(X' \otimes (W_p^T W_p + I)vec(W)) - (X \otimes (\tilde{W_p}^T \tilde{W_p})vec(W)) \\ &= -[X' \otimes (W_p^T W_p + I) + X \otimes (\tilde{W_p}^T \tilde{W_p})]vec(W)\end{aligned}$$

So we have:

$$\dot{W} = -[X' \otimes (W_p^T W_p + I) + X \otimes (\tilde{W_p}^T \tilde{W_p})]vec(W) \tag{6}$$

Now by comparing the equation (6) with the given one we prove that :

$$\dot{W} = -H(t)vec(W(t))$$

As we have the minimal eigenvalue $\lambda_{min}(H(t))$ is bounded away from zero and H is psotive semi-definite matrix, by applying the "Note" we can assume that:

$$||vec(W(t))||_2 = e^{-\lambda_0 t}||vec(W(0))||_2 \rightarrow 0$$

5.4 Consider the case when both the Stop-Grad **and** the predictor are removed. Show that the representations collapse i.e. $W(t) \rightarrow 0$. You may assume that X' is a positive definite matrix.

answer:

For removing the predictor we put: $W_p = I$ and by computing the $\dot{W}$ we will have:

$$\dot{W} = -[X' \otimes I + X \otimes [0]]vec(W) = -X'W(t)$$

X' is positive definite matrix so all it's eigenvalue is positive and we can consider that the minimal eigenvalue of X' is bounded away from zero. So based on the previous question we can prove that W(t) → 0.

5.5 Speculate (in 1-2 sentences) as to why the stop-gradient and the predictor are necessary for avoiding representational collapse.

answer:

The stop-gradient and the predictor are critical in this framework since as we proved without them the W will no longer learn any features. The main reason is because by removing the stop-gradient it makes our system unstable because when we want to compute the objective function $J(W, W_p)$ the f_1 and f_2 will be not independent as they both rely on θ.