Data Analytics and Machine Learning Group
Department of Informatics
Technical University of Munich

Esolution
Place student sticker here

Note:

- During the attendance check a sticker containing a unique code will be put on this exam.
- This code contains a unique number that associates this exam with your registration number.
- This number is printed both next to the code and to the signature field in the attendance check list.

Machine Learning for Graphs and Sequential Data

Exam:	IN2323 / Retake	**Date:**	Wednesday 7th October, 2020
Examiner:	Prof. Dr. Stephan Günnemann	**Time:**	14:15 – 15:30

Working instructions

- This exam consists of **0 pages** with a total of **10 problems**.
  Please make sure now that you received a complete copy of the exam.
- The total amount of achievable credits in this exam is 38 credits.
- Detaching pages from the exam is prohibited.
- Allowed resources:
  - all materials that you will use on your own (lecture slides, calculator etc.)
  - **not allowed are any forms of collaboration between examinees and plagiarism**
- You have to sign the code of conduct.
- Make sure that the **QR codes are visible** on every uploaded page. Otherwise, we cannot grade your exam.
- Only write on the provided sheets, **submitting your own additional sheets is not possible**.
- Last two pages can be used as scratch paper.
- All sheets (including scratch paper) have to be submitted to the upload queue. Missing pages will be considered empty.
- **Only use a black or blue color (no red or green)!**
- Write your answers only in the provided solution boxes or the scratch paper.
- **For problems that say "Justify your answer" you only get points if you provide a valid explanation.**
- **For problems that say "Prove" you only get points if you provide a valid mathematical proof.**
- If a problem does not say "Justify your answer" or "Prove" it's sufficient to only provide the correct answer.
- Instructor announcements and clarifications will be posted **on Piazza** with email notifications
- Exam duration - 75 minutes.

Left room from _____ to _____ / Early submission at _____

Problem 1 **Normalizing Flows (5 credits)**

0 1

a) Let $k \in \mathbb{N}$. We consider the transformation $f(\boldsymbol{z}) = (z - 10)^{2k}$ from $\mathbb{R}$ to $\mathbb{R}$. Is this transformation invertible ? Justify your answer. If yes, compute the determinant of its Jacobian and its inverse.

No, $10 + z$ and $10 - z$ would lead to the same output.

0 1 2

b) We consider the transformation $f(\boldsymbol{z}) = \begin{bmatrix} z_n \\ z_1 \\ : \\ : \\ z_{n-1} \end{bmatrix}$ from $\mathbb{R}^n$ to $\mathbb{R}^n$. Is this transformation invertible ? Justify your answer. If yes, compute the determinant of its Jacobian and its inverse. *Hint: The determinant of one elementary permutation (i.e. a permutation which interchanges any two rows) is* -1

Yes, it can be seen as $n - 1$ stacked permutation. The determinant of the Jacobian is $(-1)^{n-1}$. The inverse is $f(\boldsymbol{z}) = \begin{bmatrix} x_2 \\ x_3 \\ : \\ : \\ x_1 \end{bmatrix}$

0 1 2

c) We consider the transformation $f(\boldsymbol{z}) = \begin{bmatrix} z_1 + 2z_2 \\ 3z_1 + 4z_2 \\ z_3 - 2z_4 \\ 3z_3 - 4z_4 \end{bmatrix}$ from $\mathbb{R}^4$ to $\mathbb{R}^4$. Is this transformation invertible ? Justify your answer. If yes, compute the determinant of its Jacobian and its inverse.

Yes. we can compute the determinant of the Jacobian per block of size 2 and compute the inverse per block as well. The Jacobian is $J = \begin{bmatrix} 1 & 2 & 0 & 0 \\ 3 & 4 & 0 & 0 \\ 0 & 0 & 1 & -2 \\ 0 & 0 & 3 & -4 \end{bmatrix}$. Its determinant is $(1 \times 4 - 2 \times 3)(1 \times (-4) - (-2) \times 3) = -4$. The inverse is $f(\boldsymbol{z}) = \begin{bmatrix} -2x_1 + x_2 \\ \frac{3}{2}x_2 - \frac{1}{2}x_2 \\ -2x_3 + x_4 \\ -\frac{3}{2}x_3 + \frac{1}{2}x_4 \end{bmatrix}$

Problem 2 Variational Inference (3 credits)

0
1
2
3

We are performing variational inference in some latent variable model $p_\theta(x, z)$ using the following family of variational distributions $\mathcal{Q}_1 = \{\mathcal{N}(z|\phi, 1) : \phi \in \mathbb{R}\}$.
Assume that

$$p(z) = \mathcal{N}(z|0, 1) \propto \exp\left(-\frac{1}{2}z^2\right)$$
$$p(x|z) = \text{Bernoulli}(x \mid \sigma(z)) \propto \exp\left(xz - \log(1 + \exp(z))\right)$$

and x is known and fixed and $\sigma(z) = \frac{\exp(z)}{1+\exp(z)}$ is the sigmoid function. Does there exist a $q^\star \in \mathcal{Q}_1$, such that $\mathbb{KL}(q^\star(z) \,\|\, p(z|x)) = 0$? Justify your answer.

KL divergence between two probability densities is equal to zero if and only if these densities are equal. In other words, $\mathbb{KL}(q^\star(z) \,\|\, p(z|x)) = 0$ only if $p(z|x) = q^*(z)$ for some $q^* \in \mathcal{Q}_1$. This means, we need to check whether the posterior distribution $p(z|x)$ is a normal distribution with some mean ϕ and unit variance.
By Bayes' rule,

$$\begin{aligned} p(z|x) &= \frac{p(x|z) \cdot p(z)}{p(x)} \\ &\propto p(x|z) \cdot p(z) \\ &\propto \exp\left(-\frac{1}{2}z^2\right) \cdot \exp\left(xz - \log(1 + \exp(z))\right) \\ &= \exp\left(-\frac{1}{2}z^2 + xz - \log(1 + \exp(z))\right) \end{aligned}$$

We can see that $p(z|x)$ is not a normal distribution because of the non-constant term $\log(1 + \exp(z))$ inside the exponent.Therefore, there doesn't exist a normal distribution in $\mathcal{Q}_1$ that has zero KL divergence to $p(z|x)$.

Problem 3 Robustness of Machine Learning Models (3 credits)

Consider a trained binary logistic regression model with weight vector $\boldsymbol{w} \in \mathbb{R}^d$ and bias $b \in \mathbb{R}$, where d is the data dimensionality. That is, the predicted probability of a sample $\boldsymbol{x} \in \mathbb{R}^d$ belonging to class 1 is:

$$p(y = 1|\boldsymbol{x}) = \sigma(\boldsymbol{w}^T\boldsymbol{x} + b),$$

where $\sigma(z) = \frac{1}{1+e^{-z}}$ is the logistic sigmoid function. An input sample is assigned to class 1 if $p(y = 1|\boldsymbol{x}) > 0.5$, else it is assigned to class 0.
We would like to perform **robustness certification** of the logistic regression model using its **Lipschitz constant**. That is, we want to certify that the predicted class of the input sample does not change w.r.t. some radius in the input space.

0 1

a) Briefly explain why it is sufficient to consider $F(\boldsymbol{x}) = \boldsymbol{w}^T\boldsymbol{x} + b$, i.e. the model without the sigmoid activation function $\sigma(\cdot)$ for this purpose.

Because we only need to know on which side of the decision boundary an input point is mapped, i.e. whether $\boldsymbol{w}^T\boldsymbol{x} + b > 0$.

0 1 2

b) Derive the (smallest possible) Lipschitz constant of $F(\boldsymbol{x}) = \boldsymbol{w}^T\boldsymbol{x} + b$ w.r.t. the L_2 norm. Justify your answer.

$$\begin{aligned}
\|\boldsymbol{w}^T\boldsymbol{x} - \boldsymbol{w}^T\tilde{\boldsymbol{x}}\| &\leq L\|\boldsymbol{x} - \tilde{\boldsymbol{x}}\| \\
\|\boldsymbol{w}^T\boldsymbol{\Delta}\| &\leq L\|\boldsymbol{\Delta}\|, \text{ where } \boldsymbol{\Delta} = \boldsymbol{x} - \tilde{\boldsymbol{x}} \\
\frac{\|\boldsymbol{w}^T\boldsymbol{\Delta}\|}{\|\boldsymbol{\Delta}\|} &\leq L \text{ assuming } \|\boldsymbol{\Delta}\| > 0 \\
\|\boldsymbol{w}\| \cdot \underbrace{\left\| \frac{\boldsymbol{w}^T}{\|\boldsymbol{w}\|} \frac{\boldsymbol{\Delta}}{\|\boldsymbol{\Delta}\|} \right\|}_{\leq 1} &\leq L
\end{aligned}$$

Therefore, the smallest possible Lipschitz constant is $L^* = \|\boldsymbol{w}\|$.

- It is also correct to derive the result using the operator norm ($\frac{\|\boldsymbol{w}^T\boldsymbol{\Delta}\|}{\|\boldsymbol{\Delta}\|} = \|\boldsymbol{w}\|$).
- It is also correct to derive the result by showing that the norm of the gradient of $F(\boldsymbol{x})$ is upper-bounded by $\|\boldsymbol{w}\|$.

Problem 4 Hidden Markov Model (4 credits)

0
1
2
3
4

Consider an HMM where hidden variables are in $\{1,2,3\}$ and observed variables are in $\{a,b\}$. Let the model parameters be as follows:

$$\pi = \begin{bmatrix} \frac{1}{3} \\ \frac{1}{3} \\ \frac{1}{3} \end{bmatrix}, \qquad A = \begin{bmatrix} \frac{1}{4} & \frac{1}{4} & \frac{1}{2} \\ \frac{1}{2} & \frac{1}{4} & \frac{1}{4} \\ \frac{1}{4} & \frac{1}{4} & \frac{1}{2} \end{bmatrix}, \qquad B = \begin{bmatrix} 1 & 0 \\ \frac{2}{3} & \frac{1}{3} \\ 0 & 1 \end{bmatrix}$$

Assume that the sequence $X_{1:3} = [aba]$ is observed. Find the most probable sequence $[Z_1, Z_2, Z_3]$ and compute its likelihood. Justify your answer.

We remark that it is impossible to be at state 3 a time $t = 1$, at state 1 a time $t = 2$, at state 3 a time $t = 3$ (all paths containing these state/time would have probability 0). We apply the viterbi algorithm . The most probable path to go at state 2 and 3 a time $t = 2$ are $[1,2]$ and $[1,3]$ with probabilities $\frac{1}{3} \times 1 \times \frac{1}{4} \times \frac{1}{3}$ and $\frac{1}{3} \times 1 \times \frac{1}{2} \times 1$. The most probable path to go at state 1 and 2 a time $t = 3$ are $[1,3,1]$ and $[1,3,2]$ with probabilities $\frac{1}{3} \times 1 \times \frac{1}{2} \times 1 \times \frac{1}{4} \times 1$ and $\frac{1}{3} \times 1 \times \frac{1}{2} \times 1 \times \frac{1}{4} \times \frac{2}{3}$. Without using the viterbi algorithm, note that an alternative valid (but less efficient) solution would be to compare all path probabilities. The most probable sequence of states is $[1,3,1]$ with probability $\frac{1}{3} \times 1 \times \frac{1}{2} \times 1 \times \frac{1}{4} \times 1$.

Problem 5 Temporal Point Process (3 credits)

0 1 2 3

Consider a temporal point process defined on the interval $[0, 10\pi]$ with the conditional intensity function

$$\lambda^*(t) = \sin(t) + 2$$

What is the expected number of events that will be generated from this TPP on the interval $[0, 2\pi]$? Justify your answer.

The intensity function $\lambda^*(t)$ is independent of the history, so the above TPP is an inhomogenous Poissson process. Therefore, the expected number of events in $[0, 2\pi]$ is equal to the total intensity on this interval.

$$\mathbb{E}[N([0, 2\pi])] = \int_0^{2\pi} \lambda^*(t)\, dt = \int_0^{2\pi} (\sin(t) + 2)\, dt = 4\pi$$

Problem 6 Neural Network approaches for temporal data (4 credits)

0 1 2 3 4

We trained the following models to produce latent representations:

- M1: Skip-gram word2vec model
- M2: LSTM
- M3: Self-attention without positional encoding
- M4: Self-attention with positional encoding
- M5: A convolutional NN which produces the latent representation at time t based on the input at time t and $t-1$.

At **inference time**, we use the following 6 sentences as inputs:

- S1: "I left"
- S2: "They left"
- S3: "I left yesterday"
- S4: "I go left"
- S5: "left I go"
- S6: "go left"

For each model (i.e. M1, M2, M3, M4, M5), which sets of sentences (e.g. (S1, S2), (S3, S4, S6),...) are guaranteed to produce the same latent representation for the word "left" ? Note that the answer may be zero, one or more sets of sentences. Justify your answer.

- Skip-gram word2vec model: embeddings are all the same regardless of the context after training.
- LSTM : "I left" and "I left yesterday" produce same embedding. The words "left" have the same context on the left side.
- Self-attention without positional encoding: "I go left" and "left I go" produce same embeddings. The context counts but not the order of the words.
- Self-attention with positional encoding: Embeddings could be all different.
- A convolutional NN with window size of 2: "I left" and "I left yesterday" produce same embedding. "I go left" and "go left" produce same embedding. The words "left" have the same window of size 2

Problem 7 PageRank Lollipop (7 credits)

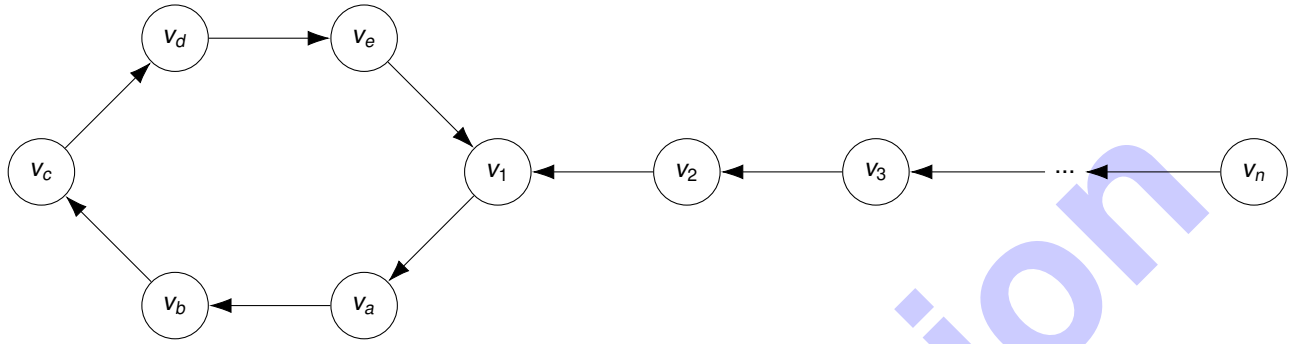


Figure 7.1: A directed "lollipop" graph with a tail of length n

Consider the directed, unweighted graph in Figure 7.1 with a "head" consisting of the six nodes v_1, and v_a through v_e. Its "tail" consists of n nodes $v_1, \ldots, v_n$ where n is a parameter. Note, that we consider v_1 to be part of the head as well as the tail.

0 1 2

a) Which of the PageRank problems of *dead end*, *spider trap*, and *periodic states* apply here? Justify your answer. For each problem that applies, give a set of edges (at most 3 per problem) that would resolve that problem if they were inserted.

The head forms a spider trap because of it is a group of connected nodes without links to nodes outside of the group. We solve this by adding an edge (v_e, v_n) that makes all other nodes reachable from within the previous spider trap.
The head also has the problem of periodic states because a random walker returns to a state exactly every 6 steps. We make v_a aperiodic by introducing the self-loop (v_a, v_a) which lets us return to v_a at arbitrary times. This way we can also reach any other node in the head at any time because we can offset our the period of 6 arbitrarily.

b) In the following, we want to compute the topic-sensitive PageRank of the nodes. For that, we introduce the topic-sensitive teleport vector π where the topic is the "tail", i.e.

$$\pi_a = ... = \pi_e = 0 \quad \text{and} \quad \pi_1 = ... = \pi_n = \frac{1}{n}.$$

State each node's PageRank equation for a teleport probability of $1 - \beta$.

0 1 2

First, the end and the inner nodes of the tail.

$$r_n = \frac{1-\beta}{n} \qquad r_i = \beta r_{i+1} + \frac{1-\beta}{n} \quad \forall i \in \{2, ..., n-1\}$$

Now the nodes making up the shell of the head.

$$r_a = \beta r_1 \qquad r_b = \beta r_a \qquad r_c = \beta r_b \qquad r_d = \beta r_c \qquad r_e = \beta r_d$$

And finally the node connecting the two.

$$r_1 = \beta (r_e + r_2) + \frac{1-\beta}{n}$$

c) Derive the sum of the PageRank of the "head" of the graph $v_1, v_a, ..., v_e$ as a function of n. Justify your answer.

Reminder: $\sum_{i=0}^{k} \beta^i = \frac{1-\beta^{k+1}}{1-\beta}$ for $\beta \neq 1$

0 1 2 3

An example solution would look like the following.
First, simple substituion gives

$$r_a = \beta r_1 \quad r_1 = \beta^2 r_1 \quad r_c = \beta^3 r_1 \quad r_d = \beta^4 r_1 \quad r_e = \beta^5 r_1$$

Plugging r_e into the equation for r_1, gets us

$$r_1 = \beta \left(\beta^5 r_1 + r_2\right) + \frac{1-\beta}{n} \Leftrightarrow r_1 = \frac{\beta r_2 + \frac{1-\beta}{n}}{1-\beta^6}$$

With this we can get an expression for the sum of the PageRank of the head.

$$r_1 + r_a + ... + r_e = \left(\sum_{i=0}^{5} \beta^i\right) r_1 = \frac{1-\beta^6}{1-\beta} \frac{\beta r_2 + \frac{1-\beta}{n}}{1-\beta^6} = \frac{\beta r_2 + \frac{1-\beta}{n}}{1-\beta}$$

This leaves us with r_2 as the only remaining unknown. By plugging in along the tail and generalizing, we find

$$r_i = \frac{1-\beta}{n} \left(\sum_{i=0}^{n-i} \beta^i\right) = \frac{1-\beta}{n} \cdot \frac{1-\beta^{n-i+1}}{1-\beta} = \frac{1-\beta^{n-i+1}}{n} \quad \text{for } i = 2, ..., n-1.$$

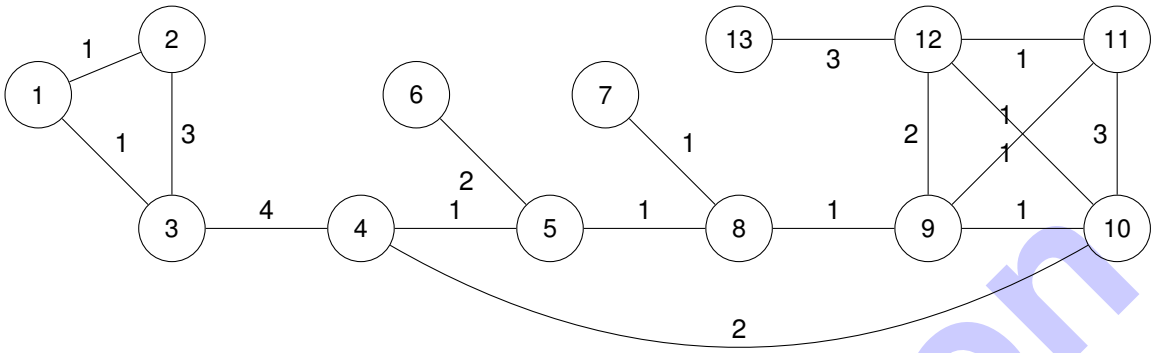
Finally, we plug this into the expression above and get the final answer

$$r_1 + r_a + ... + r_e = \frac{\beta \frac{1-\beta^{n-1}}{n} + \frac{1-\beta}{n}}{1-\beta} = \frac{1-\beta^n}{n(1-\beta)}.$$

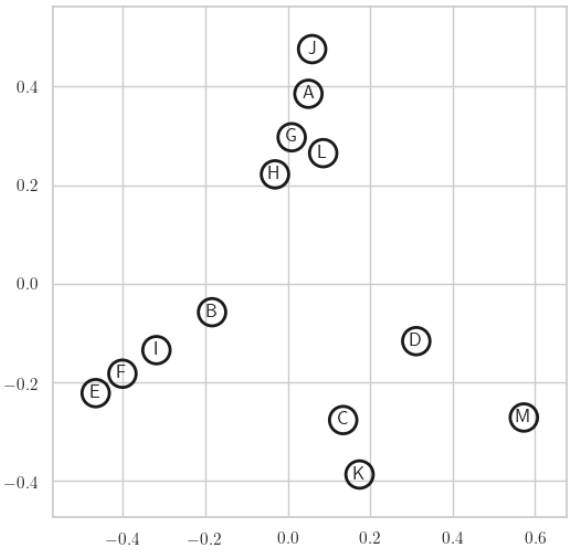
Problem 8 Spectral Embeddings (3 credits)

0
1
2
3

We use spectral embedding to map the following undirected, weighted graph into 2-dimensional space.



The following plot shows the embeddings as given by the eigenvectors belonging to the second and third smallest eigenvalues of the unnormalized Lapalacian.



Fill out the following table that maps nodes in the graph to points in the plot.

Node	1	2	3	4	5	6	7	8	9	10	11	12	13
Point	E	F	I	B	C	K	M	D	L	H	G	A	J

Nodes are mapped closer together the higher the edge weight between them.

Problem 9 Graph Neural Networks (4 credits)

a) Consider the following graph neural network.

0 1 2

$$\boldsymbol{Z} = \sigma\left(\sigma\left(\hat{\boldsymbol{A}}\sigma\left(\hat{\boldsymbol{A}}\sigma\left(\hat{\boldsymbol{A}}\boldsymbol{X}\right)\boldsymbol{W}_a\right)\right)\boldsymbol{W}_b\right)\boldsymbol{W}_c$$

$\boldsymbol{X} \in \mathbb{R}^{N\times D}$ are the node features, $\boldsymbol{Z}$ are the node predictions, σ is the sigmoid function, $\boldsymbol{W}_x$ are weight matrices of appropriate dimensions and $\hat{\boldsymbol{A}} = \tilde{\boldsymbol{D}}^{-\frac{1}{2}}\tilde{\boldsymbol{A}}\tilde{\boldsymbol{D}}^{-\frac{1}{2}}$ is the propagation matrix as defined for GCNs, i.e. $\tilde{\boldsymbol{A}} = \boldsymbol{A} + \boldsymbol{I}$ is the adjacency matrix with added self-loops and $\tilde{\boldsymbol{D}}_{ii} = \sum_{j=1}^{N}\tilde{\boldsymbol{A}}_{ij}$ is the degree matrix of the amended adjacency matrix. Give the transformation and propagation depths of this model. Justify your answer.
Hint: We define

- *propagation depth* as the maximum distance that a model propagates information in the graph,
- *transformation depth* as the number of non-linear transformations that are applied to the features.

Each application of $\hat{\boldsymbol{A}}$ propagates the node information one step, so the propagation depth is 3. Transformation depth is defined as the number of non-linear transformations of which there are four. Therefore the transformation depth is 4 regardless of the number of weight matrices applied.

b) Write down a formula for the node predictions $\boldsymbol{Z}$ of a graph neural network with propagation depth 2 and transformation depth 2 where the transformations use fully connected layers with weight matrices W_i and biases b_i. *Note*: You can assume that operations such as matrix-vector summation broadcast as expected.

0 1 2

A straightforward solution would be

$$\boldsymbol{Z} = \hat{\boldsymbol{A}}\hat{\boldsymbol{A}}\sigma\left(\sigma\left(\boldsymbol{X}\boldsymbol{W}_1 + \boldsymbol{b}_1\right)\boldsymbol{W}_2 + \boldsymbol{b}_2\right)$$

Problem 10 Randomized Smoothing on Graph Neural Networks (2 credits)

Consider an arbitrary but fixed graph neural network $f(\boldsymbol{X})$ as base classifier. Here, $\boldsymbol{X}$ *denotes the adjacency matrix* (i.e. the features are assumed to be constant).
For the the smooth classifier $g(\boldsymbol{X})$ we use the randomization scheme $\phi(\mathbf{x})$:

$$g(\mathbf{x})_c = \mathcal{P}(f(\phi(\mathbf{x})) = c) = \sum_{\tilde{\mathbf{x}} \text{ s.t. } f(\tilde{\mathbf{x}})=c} \prod_{i=1}^{n^2} \mathcal{P}(\tilde{\mathbf{x}}_i|\mathbf{x}_i) \tag{1}$$

with

$$\mathcal{P}(\tilde{\mathbf{x}}_i|\mathbf{x}_i) = \begin{cases} p & \tilde{\mathbf{x}}_i = 1 - \mathbf{x}_i \\ 1-p & \tilde{\mathbf{x}}_i = \mathbf{x}_i \end{cases} \tag{2}$$

Note that in contrast to the lecture here we do not distinguish between a probability for deleting p_d or adding p_a an edge. We simply use the "flip" probability p.

Furthermore, we consider the special case of $p = 0.5$.

0 1 a) What is the probability $\mathcal{P}(\tilde{\mathbf{x}}|\mathbf{x})$ for a directed graph or is there not enough information to determine it? Justify your answer.

Each graph is obtained equally likely: $\mathcal{P}(\tilde{\mathbf{x}}|\mathbf{x}) = \frac{1}{2^{n^2}}$ (n^2 entries in the adjacency matrix, 2 possible values each)

0 1 b) We are given the prediction for the original input $f(\boldsymbol{X}) = c$ and for a slightly different version $f(\boldsymbol{X}') \neq c$.
What does the randomization scheme entail about the (exact) prediction of the smooth classifier $g(\boldsymbol{X})_c$ and $g(\boldsymbol{X}')_c$ (i.e. no Monte Carlo approximation)? Specifically, how do the probabilities for class c of the smooth classifier $g(\boldsymbol{X})_c$ and $g(\boldsymbol{X}')_c$ relate?

1. $g(\boldsymbol{X})_c = g(\boldsymbol{X}')_c$
2. $g(\boldsymbol{X})_c < g(\boldsymbol{X}')_c$
3. $g(\boldsymbol{X})_c > g(\boldsymbol{X}')_c$
4. There is not enough information to determine how $g(\boldsymbol{X})_c$ and $g(\boldsymbol{X}')_c$ relate

Justify your answer.

Since the randomization scheme is somewhat uninformative and there is no such thing as a more similar or more dissimilar graph, the prediction of the smooth classifier is the same regardless of $\boldsymbol{X}$ or $\boldsymbol{X}'$. I.e. despite $f(\boldsymbol{X}) \neq f(\boldsymbol{X}')$, $g(\boldsymbol{X})_c = g(\boldsymbol{X}')_c$ holds.

Additional space for solutions–clearly mark the (sub)problem your answers are related to and strike out invalid solutions.