Professorship of Data Mining and Analytics
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.

Mining Massive Datasets

Exam:	IN2323 / Retake	**Date:**	Monday 30th September, 2019
Examiner:	Prof. Dr. Stephan Günnemann	**Time:**	08:00 – 09:30

	P 1	P 2	P 3	P 4	P 5	P 6	P 7	P 8
I								

Working instructions

- This exam consists of **12 pages** with a total of **8 problems**.
  Please make sure that you received a complete copy of the exam.
- You can earn 43 points.
- **Detaching pages from the exam is prohibited!**
- Allowed resources:
  - A4 sheet of handwritten notes (two sides)
  - **no other materials (e.g. books, cell phones, calculators) are allowed!**
- Only write on the sheets given to you by supervisors. If you need more paper, ask the supervisors.
- Last two pages can be used as scratch paper.
- All sheets (including scratch paper) have to be returned at the end.
- **Only use a black or a blue pen (no pencils, red or green pens)!**
- Write your answers only in the provided solution boxes or the scratch paper.
- **For problems that say "Justify your answer" or "Show your work" you only get points if you provide a valid explanation.** Otherwise it's sufficient to only provide the correct answer.
- Exam duration - 90 minutes.

Left room from __________ to __________ / Early submission at __________

Problem 1 AR models: stationarity (5 points)

Decide whether the following AR models are stationary or not. Everywhere $\epsilon_t \sim \mathcal{N}(0, \sigma^2)$ with a positive σ.

Mark correct answers with a cross ☒

To undo a cross, completely fill out the answer option ■

To re-mark an option, use a human-readable marking ✕■

a) $\mathcal{X}_t = c + 0.1\mathcal{X}_{t-1} + \epsilon_t$ with some $c \in \mathbb{R}$

☐ stationary for $|c| > 0.1$, otherwise non-stationary [✓ ☒ yes, always stationary] ☐ no, always non-stationary

characteristic polynome is $p(x) = 1 - 0.1x$
the only root is $x = 10$ with $|x| = 10 > 1$, therefore stationary

b) $\mathcal{X}_t = -3 + 0.2\mathcal{X}_{t-1} - 0.01\mathcal{X}_{t-2} + c\epsilon_t$ with some $c \in \mathbb{R} \setminus \{0\}$

☐ stationary for $c > 0$, otherwise non-stationary [✓ ☒ yes, always stationary] ☐ no, always non-stationary

characteristic polynome is $p(x) = 1 - 0.2x + 0.01x^2 = (1 - 0.1x)^2$
the only root is $x = 10$ with $|x| = 10 > 1$, therefore stationary

c) $\mathcal{X}_t = 1 + 0.3\mathcal{X}_{t-1} - 0.03\mathcal{X}_{t-2} + 0.001\mathcal{X}_{t-3} + \epsilon_t$

☐ no, non-stationary [✓ ☒ yes, stationary]

characteristic polynome is $p(x) = 1 - 3(0.1x) + 3(0.1x)^2 - (0.1x)^3 = (1 - 0.1x)^3$
the only root is $x = 10$ with $|x| = 10 > 1$, therefore stationary

d) $\mathcal{X}_t = -2 + 0.5\mathcal{X}_{t-n} + \epsilon_t$ with some $n \in \mathbb{N}$

☐ no, always non-stationary ☐ stationary for $n \leq 2$, otherwise non-stationary [✓ ☒ yes, always stationary]

characteristic polynome is $p(x) = 1 - 0.5x^n$
there are n roots with $|x| = 2^{\frac{1}{n}} > 1$, therefore stationary

e) $\mathcal{X}_t = 2019 - \sum_{i=1}^{n} a^i \mathcal{X}_{t-i} + \epsilon_t$ with some $n \in \mathbb{N}$, $a \in \mathbb{R} \setminus \{0\}$

[✓ ☒ stationary for $|a| < 1$, otherwise non-stationary] ☐ stationary for $|a|^n < 2019$, otherwise non-stationary ☐ stationary for $n = 1$, $|a| < 1$, otherwise non-stationary

characteristic polynome is $p(x) = 1 + \sum_{i=1}^{n}(ax)^i = \frac{1-(ax)^{n+1}}{1-ax}$
there are n roots with $|x| = \frac{1}{|a|}$, therefore stationary when $|a| < 1$, otherwise not

Problem 2 Hidden Markov Models (7 points)

Consider the following Hidden Markov Model where $Z_t \in \{0,1\}$ are latent variables and $X_t \in \{a,b\}$ are discrete observed variables. We parametrize the prior and transition probabilities $P(Z_1 = i) = \pi_i$, $P(Z_{t+1} = j|Z_t = i) = A_{ij}$ and $P(X_t = j|Z_t = i) = B_{ij}$ by:

$$\pi = \begin{bmatrix} 1/2 \\ 1/2 \end{bmatrix}, \quad A = \begin{bmatrix} \alpha & 1-\alpha \\ 1/2 & 1/2 \end{bmatrix}, \quad B = \begin{bmatrix} 3/4 & 1/4 \\ 1/4 & 3/4 \end{bmatrix}$$

We assume we observed $X = [a, b, a]$.

a) Compute $P(Z_2|X_1, X_2)$ and $P(Z_2|X_1, X_2, X_3)$ as a function of α.

☐ 0
☐ 1
☐ 2
☐ 3
☐ 4
☐ 5

Apply forward two times with initialisation and recursion formulas

$$\alpha_1 = \begin{bmatrix} 1/2 \\ 1/2 \end{bmatrix} \odot \begin{bmatrix} 3/4 \\ 1/4 \end{bmatrix} = \begin{bmatrix} 3/8 \\ 1/8 \end{bmatrix}$$

$$\alpha_2 = \begin{bmatrix} 1/4 \\ 3/4 \end{bmatrix} \odot \left(\begin{bmatrix} \alpha & 1/2 \\ 1-\alpha & 1/2 \end{bmatrix} \begin{bmatrix} 3/8 \\ 1/8 \end{bmatrix} \right) = \frac{1}{32} \begin{bmatrix} 3\alpha + 1/2 \\ 21/2 - 9\alpha \end{bmatrix}$$

Apply backward two times with initialisation and recursion formulas

$$\beta_1 = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

$$\beta_2 = \begin{bmatrix} \alpha & 1-\alpha \\ 1/2 & 1/2 \end{bmatrix} \left(\begin{bmatrix} 3/4 \\ 1/4 \end{bmatrix} \odot \begin{bmatrix} 1 \\ 1 \end{bmatrix} \right) = \frac{1}{2} \begin{bmatrix} \alpha + 1/2 \\ 1 \end{bmatrix}$$

$$\alpha_2\beta_2 = \frac{1}{64} \begin{bmatrix} 3\alpha + 1/2 \\ 21/2 - 9\alpha \end{bmatrix} \odot \begin{bmatrix} \alpha + 1/2 \\ 1 \end{bmatrix} = \frac{1}{64} \begin{bmatrix} 3\alpha^2 + 2\alpha + 1/4 \\ -9\alpha + 21/2 \end{bmatrix}$$

Correct results. $P(Z_2|X_1, X_2)$ and $P(Z_2|X_1, X_2, X_3)$ are equal to α_2 and $\alpha_2\beta_2$ after normalization.

0 ☐
1 ☐
2 ☐

b) What is the most probable state Z_2 according to $P(Z_2|X_1, X_2)$ and $P(Z_2|X_1, X_2, X_3)$ for any value of $\alpha \in [0, 1]$.
Hint: $\sqrt{244} \approx 15.62$

Correct results for online clustering. State 0 is more probable if $\alpha > 5/6$, otherwise it is state 1.
$P(Z_2 = 0|X_1, X_2) > P(Z_2 = 1|X_1, X_2) \Leftrightarrow 3\alpha + 1/2 > -9\alpha + 21/2 \Leftrightarrow \alpha > 5/2$
Correct results for offline clustering. State 0 is more probable if $\alpha > 4.62/6$, otherwise it is state 1.
$P(Z_2 = 0|X_1, X_2, X_3) > P(Z_2 = 1|X_1, X_2, X_3) \Leftrightarrow 3\alpha^2 + 2\alpha + 1/4 > -9\alpha + 21/2 \Leftrightarrow 3\alpha^2 + 11\alpha - 41/4 > 0$
The roots are $\frac{-11+\sqrt{244}}{6} \approx 4.62/6$ and $\frac{-11-\sqrt{244}}{6} < 0$.

Problem 3 RNNs & Word vectors (7 points)

You are solving a question-answering task. Given a context and a question, the goal is to find the answer **inside** the context. Bellow are two examples (1 and 2).

id	Context	Question	Answer
1	Mary was in the bathroom. Then she moved to the hallway.	Where is Mary?	hallway
2	John is in the hallway. Mary is there as well.	Where is Mary?	hallway

Assume that the question is represented with the vector $\boldsymbol{q}$. We want to know what is the probability that a word from the context is the answer. We decide to somehow represent every word w_i with an embedding $\boldsymbol{h}_i$ and pass it together with $\boldsymbol{q}$ through a neural network to get the probabilities. The only thing left to do is to decide how to get $\boldsymbol{h}_i$. We propose two approaches: sliding window and RNN.

a) **Sliding window** — Every word w_i is represented with a pretrained word vector $\boldsymbol{v}_i$. A sliding window of size 2 takes the neighbouring words and constructs the embedding for w_i as a sum of vectors: $\boldsymbol{h}_i = \sum_{j=i-2}^{i+2} \boldsymbol{v}_j$. Is it possible for this model to find the right answer in example 1? What about example 2? Justify.

☐ 0 ☐ 1 ☐ 2

Example 1: No, because "Mary" or "she" is outside the sliding window when on "hallway" and vice versa.
Example 2: Yes, because "hallway" and "Mary" are close to each other so sliding window covers them.

b) **RNN** — As an alternative we use an RNN that takes pretrained word vectors $\boldsymbol{v}_i$ from left to right and outputs $\boldsymbol{h}_i$ as a word embedding. Why is this model able to output the right answer in example 1? Explain why it could fail answering example 2?

☐ 0 ☐ 1 ☐ 2

Example 1: It can output the correct answer because RNN can save the information and model long term dependencies.
Example 2: It might fail because "Mary" is on the right of the answer so reading it from left to right, it might associate hallway to John and "there" to Mary. Again, since it's only going left to right, "there" will not be correctly connected to the hallway.

c) Propose another model that overcomes the shortcomings of the sliding window and the RNN. Describe what the input would be and how would you calculate the embedding $\boldsymbol{h}_i$. Explain why this model could work on both examples.

☐ 0 ☐ 1 ☐ 2 ☐ 3

E.g. bidirectional RNN, multilayer RNN, 1-D convolution (Wavenet) etc.
Describing how the model works in detail
Showing that it solves the problem of long term dependencies and future dependencies

Problem 4 Deep Generative Model (5 points)

0 ☐
1 ☐
2 ☐
3 ☐

a) You are given a pseudo code implementation of 4 different variants of an AutoEncoder. Here, $\boldsymbol{x}_i \in \mathbb{R}^d$ is the input data, and $f_\theta : \mathbb{R}^d \mapsto \mathbb{R}^k$ and $g_\phi : \mathbb{R}^k \mapsto \mathbb{R}^d$ are fully connected feed-forward neural networks where $\boldsymbol{\theta}$ and ϕ are learnable parameters. The final layers of f_θ and g_ϕ have no (i.e. have linear) activation functions. $\boldsymbol{I}_k$ is a $k \times k$ identity matrix, $\boldsymbol{0}_k$ is a k-dimensional vector of zeros, $\mathcal{N}$ is the Normal distribution, σ is the softmax function, and $diag(\boldsymbol{x})$ takes a vector $\boldsymbol{x} \in \mathbb{R}^k$ and returns a $k \times k$ diagonal matrix with the vector values on the diagonal.

AutoEncoder 1

```
$\epsilon_i \sim \mathcal{N}(\boldsymbol{x}_i, \boldsymbol{I}_d)$
$\boldsymbol{h}_i = f_\theta(\epsilon_i)$
$\tilde{\boldsymbol{x}}_i = g_\phi(\boldsymbol{h}_i)$
$\mathcal{L} = \sum_i \|\boldsymbol{x}_i - \tilde{\boldsymbol{x}}_i\|_2$
```

AutoEncoder 2

```
$\boldsymbol{h}_i = f_\theta(\boldsymbol{x}_i)$
$\epsilon_i \sim \mathcal{N}(\boldsymbol{h}_i, \boldsymbol{I}_k)$
$\tilde{\boldsymbol{x}}_i = g_\phi(\epsilon_i)$
$\mathcal{L} = \sum_i \|\boldsymbol{x}_i - \tilde{\boldsymbol{x}}_i\|_2$
```

AutoEncoder 3

```
$\boldsymbol{h}_i = f_\theta(\boldsymbol{x}_i)$
$\epsilon_i \sim \mathcal{N}(\boldsymbol{0}_k, \boldsymbol{I}_k)$
$\tilde{\boldsymbol{x}}_i = g_\phi(\boldsymbol{h}_i + \epsilon_i)$
$\mathcal{L} = \sum_i \|\boldsymbol{x}_i - \tilde{\boldsymbol{x}}_i\|_2$
```

Is it **necessary** to use the reparametrization trick in order to compute the gradients of $\mathcal{L}$ w.r.t. to **both** θ and ϕ in the above implementations? Answer with Yes or No and provide a justification. If the answer is Yes, modify the pseudo code to implement the reparametrization trick.

AutoEncoder 1

No, the sampling does not depend on the parameters.

AutoEncoder 2

Yes, the sampling depends on the parameters. We use the reparametrization trick. Specifically, we replace line 2 with the following lines:

$$\epsilon_i \sim \mathcal{N}(\boldsymbol{0}_k, \boldsymbol{I}_k)$$
$$\epsilon_i = \boldsymbol{h}_i + \epsilon_i$$

AutoEncoder 3

No, the sampling does not depend on the parameters.

b) Assume the same setup as in a). The model specified by the following pseudo code is **not well defined**. Specify the reason why, and modify the pseudo code such that the model becomes well-defined. In addition, if you think it is **necessary** to use the reparametrization trick, please include it in your implementation.

☐ 0
☐ 1
☐ 2

$$\begin{aligned}
\boldsymbol{h}_i &= f_\theta(\boldsymbol{x}_i)\\
\epsilon_i &\sim \mathcal{N}(\boldsymbol{0}_k, diag(\boldsymbol{h}_i))\\
\tilde{\boldsymbol{x}}_i &= g_\phi(\epsilon_i)\\
\mathcal{L} &= \sum_i \|\boldsymbol{x}_i - \tilde{\boldsymbol{x}}_i\|_2
\end{aligned}$$

We have to make sure that the covariance matrix is positive semi-definite in order for the model to be well-defined.
Since the sampling depends on the parameters we use the reparametrization trick.
Specifically, we replace line 2 with the following lines:

$$\begin{aligned}
\epsilon_i &\sim \mathcal{N}(\boldsymbol{0}_k, \boldsymbol{I}_k)\\
\epsilon_i &= \epsilon_i \cdot \exp(\boldsymbol{h}_i).
\end{aligned}$$

Note that the covariance matrix now is $diag(\sqrt{\exp(\boldsymbol{h}_i)})$ instead of $diag(\boldsymbol{h}_i)$ and therefore well-defined.
This output transformation will be absorbed into f_θ during training.

Problem 5 Spectral clustering (3 points)

0 1 2 3

Given is the following matrix $\boldsymbol{M} \in \mathbb{R}^{9\times 9}$.

$$\boldsymbol{M} = \begin{bmatrix} 1 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & -1 & 2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 2 & -1 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 2 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & -1 & 3 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 1 \end{bmatrix}$$

Write down the **exact** value of the three smallest eigenvalues λ_1, λ_2, λ_3 of $\boldsymbol{M}$ and the respective eigenvectors $\boldsymbol{x}_1, \boldsymbol{x}_2, \boldsymbol{x}_3$.

$\boldsymbol{M}$ is the unnormalized Laplacian matrix of a graph with 9 nodes. The graph consists of 3 disconnected components, namely (1, 2, 3), (4, 5, 6, 7) and (8, 9). Therefore the three smallest eigenvalues of the Laplacian are all zero: $\lambda_1 = \lambda_2 = \lambda_3 = 0$. The respective eigenvectors can be chosen as indicators of the disconnected components:

$$\boldsymbol{x}_1 = \begin{bmatrix} 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$\boldsymbol{x}_2 = \begin{bmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 \end{bmatrix}$$

$$\boldsymbol{x}_3 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \end{bmatrix}$$

Problem 6 Spectral clustering (3 points)

0 1 2 3

You are given an undirected graph $G = (V, E)$. It is known that the second smallest eigenvalue of the unnormalized Laplacian $\boldsymbol{L} = \boldsymbol{D} - \boldsymbol{W}$ is equal to 10.
Let $\phi(G)$ denote the best possible ratio cut achievable on the graph G

$$\phi(G) = \min_{S \subset V} \text{ratio-cut}(S, \bar{S})$$

What are possible values of $\phi(G)$ for the given graph? Select all that apply. Justify your answer.

a) 1

b) 2

c) 4

d) 8

e) 16

As shown in the lecture, the second smallest eigenvalue of the unnormalized Laplacian

$$\lambda_2 = \min_{\boldsymbol{x}:\boldsymbol{x}\perp\boldsymbol{1}} \boldsymbol{x}^T \boldsymbol{L} \boldsymbol{x}$$

is the relaxation of the minimum ratio cut problem $\phi(G)$.

Therefore, the true value of the minimal ratio cut $\phi(G)$ can only be **larger** than the solution to the relaxed problem. Hence, 16 is the only acceptable answer.

Problem 7 Ranking (5 points)

Given a graph G with 5 nodes, assume that you have access to several topic-sensitive PageRank vectors, each pre-computed using a **different** teleport set S, and the **same (fixed)** teleport parameter β, $0 < \beta < 1$.

- $\pi_{235} \in \mathbb{R}^5$, with teleport set $S = \{2, 3, 5\}$
- $\pi_{124} \in \mathbb{R}^5$, with teleport set $S = \{1, 2, 4\}$
- $\pi_{134} \in \mathbb{R}^5$, with teleport set $S = \{1, 3, 4\}$
- $\pi_3 \in \mathbb{R}^5$, with teleport set $S = \{3\}$

Assume that the random walker always teleports **uniformly at random** to each node in the teleport set.

Is it possible to compute each of the following PageRank vectors without access to the graph G, i.e. using only the above pre-computed vectors? If so, specify the exact equation as a function of $\pi_{235}, \pi_{124}, \pi_{134}$ and π_3. If not, justify why not.

a) Is it possible to compute $\pi_{14} \in \mathbb{R}^5$ with teleport set $S = \{1, 4\}$?

0
1

Yes, $\pi_{14} = 3\pi_{134} - \pi_3$.

b) Is it possible to compute $\pi_5 \in \mathbb{R}^5$ with teleport set $S = \{5\}$?

0
1

Yes, $\pi_5 = 3\pi_{235} - 3\pi_{124} + 3\pi_{134} - 2\pi_3$.

c) Is it possible to compute $\pi_1 \in \mathbb{R}^5$ with teleport set $S = \{1\}$?

0
1

No, we cannot distinguish between nodes 1 and 4.

d) Is it possible to compute $\pi_w \in \mathbb{R}^5$ with teleport set $S = \{1, 2, 3, 4, 5\}$, where we do not teleport to each node uniformly at random but rather with weights 0.2, 0.3, 0.1, 0.2, 0.2, respectively?

0
1
2

Yes, $\pi_w = 0.3(2\pi_{235} + \pi_{124} + \pi_{134}) - 0.2\pi_3$.

Problem 8 Graph Neural Networks (8 points)

Given an unweighted, undirected graph G with adjacency matrix $\boldsymbol{A} \in \{0,1\}^{N\times N}$ and node attribute matrix $\boldsymbol{X} \in \mathbb{R}^{N\times D}$, your task is to perform semi-supervised node classification with C classes using a graph convolutional network (GCN).
In matrix notation, a GCN with K layers is recursively defined as follows.

$$\boldsymbol{H}^{(0)} = \boldsymbol{X}$$
$$\boldsymbol{H}^{(k)} = \sigma^{(k)}\left(\tilde{\boldsymbol{A}}\boldsymbol{H}^{(k-1)}\boldsymbol{W}^{(k)}\right) \qquad \text{for } k \in 1,...,K$$

That is, $\boldsymbol{H}^{(K)} \in \mathbb{R}^{N\times C}$ contains the class predictions for **all nodes** stacked in a matrix. Here, $\sigma^{(k)}$ is the ReLU$(\cdot)$ activation function for $k \in \{1,...,K-1\}$ and the softmax$(\cdot)$ function for the final (output) layer $k = K$. $\boldsymbol{W}^{(k)}$ is the weight matrix of layer k.
$\tilde{\boldsymbol{A}} \in \mathbb{R}^{N\times N}$ is a degree-normalized version of the adjacency matrix whose entries are

$$\tilde{\boldsymbol{A}}_{uv} = \begin{cases} \frac{1}{\sqrt{d_u d_v}} & \text{if } \boldsymbol{A}_{uv} = 1 \\ \frac{1}{d_u} & \text{if } u = v \\ 0 & \text{else,} \end{cases}$$

where d_u is the degree of node u.

0 1 2 3

a) GCN is an instance of the differentiable message passing framework. Provide the corresponding message function M and update function U of GCN. For simplicity, you may write $\sigma^{(k)}$ for all layers.

$$M(h_v^{(k-1)}, h_u^{(k-1)}, E_{vu}) = \frac{1}{\sqrt{d_u d_v}} W^{(k)} h_u^{(k-1)}$$
$$m_v^{(k)} = \sum_{u\in N(v)} M(h_v^{(k-1)}, h_u^{(k-1)}, E_{vu})$$
$$U(h_v^{(k-1)}, m_v^{(k)}) = \sigma^{(k)}\left(\frac{1}{d_v} W^{(k)} h_v^{(k-1)} + m_v^{(k)}\right)$$

0 1

b) Assume you have a GCN with 3 layers, i.e. $K = 3$. Provide the non-recursive (unrolled) expression for $\boldsymbol{H}^{(3)}$. That is, write down the single-line equation of $\boldsymbol{H}^{(3)}$ in matrix form.

$$\boldsymbol{H}^{(3)} = \text{softmax}\left(\tilde{\boldsymbol{A}}\text{ReLU}\left(\tilde{\boldsymbol{A}}\text{ReLU}\left(\tilde{\boldsymbol{A}}\boldsymbol{X}\boldsymbol{W}^{(1)}\right)\boldsymbol{W}^{(2)}\right)\boldsymbol{W}^{(3)}\right)$$

c) We consider now the two following situations:

☐ 0
☐ 1
☐ 2

(1) $\boldsymbol{A}_{uv} = 1$ for $u = v$ and 0 else.

(2) $\sigma^{(k)}(x) = x$, i.e. identity activation function, for $k \in \{1, 2\}$

Apart from the additional information in each situation, the models are identical to the GCN defined above. Simplify the single-line, matrix-form expression of $\boldsymbol{H}^{(3)}$ given the additional information provided in each situation.

(1) $\boldsymbol{H}^{(3)} = \text{softmax}\left(\text{ReLU}\left(\text{ReLU}\left(\boldsymbol{X}\boldsymbol{W}^{(1)}\right)\boldsymbol{W}^{(2)}\right)\boldsymbol{W}^{(3)}\right)$

(2) $\boldsymbol{H}^{(3)} = \text{softmax}\left(\tilde{\boldsymbol{A}}\tilde{\boldsymbol{A}}\tilde{\boldsymbol{A}}\boldsymbol{X}\boldsymbol{W}^{(1)}\boldsymbol{W}^{(2)}\boldsymbol{W}^{(3)}\right) = \text{softmax}\left(\tilde{\boldsymbol{A}}^{3}\boldsymbol{X}\tilde{\boldsymbol{W}}\right)$

d) For each situation, find one equivalent model in the table below. **You may select each option only once**. Briefly justify your answer for each situation.

☐ 0
☐ 1
☐ 2

Recurrent neural network (RNN)	Linear regression
Feed-forward neural network (FFNN)	Label Propagation (LP)
Linear function	Deep Generative Model
Logistic regression on $\boldsymbol{X}$	Logistic regression on pre-processed features $\tilde{\boldsymbol{X}}$

(1) There is no message-passing since $\tilde{\boldsymbol{A}}$ was replaced by the identity matrix. This means that the model is a feed-forward neural network with three layers.

(2) The weight matrices can again be combined into one. We can further define $\tilde{\boldsymbol{X}} = \tilde{\boldsymbol{A}}^{3}\boldsymbol{X}$ as a preprocessed feature matrix. Thus, the model corresponds to logistic regression with pre-processed features

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