به نام خدا

تمرین سری سوم
درس یادگیری ماشین
دکتر علی شریفی زارچی

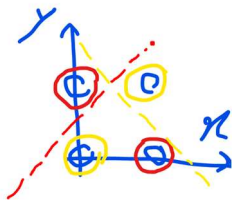
فرزان رحمانی
۴۰۳۲۱۰۷۲۵

سوال اول

الف) از ۱۶ تابع بولین مجزا در دو ورودی، ۱۴ تابع را می توان با یک **perceptron** نشان داد.

استدلال: یک پرسپترون می تواند هر تابع بولین خطی قابل تفکیک(linearly separable boolean function) را نشان دهد. در مورد دو ورودی، چهار ترکیب ورودی ممکن وجود دارد (۰،۰)، (۰،۱)، (۱،۰)، (۱،۱)، که منجر به ۲۸۴=۱۶ تابع بولین ممکن می شود. با این حال، تنها توابعی که می توانند با یک خط مستقیم در فضای ورودی از هم جدا شوند، به صورت خطی قابل تفکیک هستند.

دو تابعی که به صورت خطی قابل تفکیک نیستند، توابع XOR (exclusive OR) و XNOR (exclusive NOR) هستند. این توابع را نمی توان با پرسپترون نشان داد زیرا هیچ مرز خطی وجود ندارد که بتواند کلاس های خروجی آنها را از هم جدا کند. بنابراین، یک پرسپترون می تواند بقیه توابع یعنی ۱۴ تابع از ۱۶ تابع بولین را برای دو ورودی نشان دهد.



توضیحات بیشتر:

- ما می توانیم فضای ورودی را به صورت هندسی به صورت ۴ نقطه تجسم کنیم: (۰،۰)، (۰،۱)، (۱،۰)، و (۱،۱).
- یک تابع تفکیک پذیر خطی به این معنی است که می توانیم یک خط مستقیم رسم کنیم که خروجی های درست را از خروجی های نادرست جدا می کند.

بنابراین توابع که به صورت خطی قابل تفکیک هستند عبارتند از:

- توابع Single input (۴ تابع - بستگی به x1 یا x2 یا نقیض آنها دارند)
- توابع ثابت یا Constant (۲ تابع - همیشه درست یا همیشه نادرست)
- توابعی که شامل ۳ خروجی درست و ۱ خروجی نادرست یا بر عکس ۳ خروجی نادرست و ۱ خروجی درست هستند. (۸ تابع - شامل AND, OR, NAND, NOR, ...)
- ۸ + ۲ + ۴ = ۱۴

بنابراین، از ۱۶ تابع بولی ممکن، تنها ۱۴ تابع را می توان با یک پرسپترون نشان داد. ۲ تابع باقیمانده، یعنی XOR و XNOR، به صورت خطی قابل تفکیک نیستند و نمی توان آنها را با یک پرسپترون نشان داد.

ب) غلط. هر دو tanh و sigmoid می توانند از مشکلات vanishing gradient رنج ببرند. در حالی که tanh بر خلاف سیگموئید صفر محور است، همچنان دارای گرادیان هایی است که برای ورودی های مثبت یا منفی بزرگ بسیار کوچک می شوند.

ج) صحیح. هر تابع منطقی را می توان با یک شبکه عصبی دو لایه با یک لایه پنهان با اندازه کافی نشان داد. توابع فعال‌سازی ReLU می‌توانند مرزهای تصمیم‌گیری غیرخطی را بازنمایی کنند و میتوانند هر تابع منطقی را تقریب بزنند.
Universal Approximation Theorem: Any continuous function can be approximated by a two-layer neural) (network with appropriate activation function.

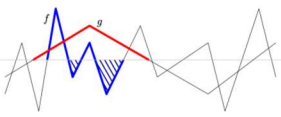
د) غلط. افزایش عمق و عرض می تواند منجر به overfitting ، افزایش زمان آموزش، و vanishing gradients (برای افزایش عمق) شود و همیشه عملکرد را بهبود نمی بخشد. همچنین ممکن است آموزش را دشوارتر کند. بنابراین، همیشه عملکرد بهتر را تضمین نمی کند.

هـ) غلط. در حالی که یک شبکه کم عمق و عریض(hallow and wide network) از نظر تئوری می تواند هر تابعی را تقریب بزند (the Universal Approximation Theorem)، شبکه های عمیق به دلیل ویژگی ها و ساختار سلسله مراتبی خود اغلب در نمایش توابع پیچیده با نورون های کمتر کارآمدتر هستند.

Depth Separation Theorem

- This theorem can be proved by showing that:
  1. Functions with few oscillations poorly approximate functions with many oscillations.
  2. Functions computed by networks with few layers must have few oscillations.
  3. Functions computed by networks with many layers can have many oscillations.

- The key idea is that just a few function compositions (layers) suffice to construct a highly oscillatory function, whereas function addition (adding nodes but keeping depth fixed) gives a function with few oscillations. Thereafter, an elementary counting argument suffices to show that low-oscillation functions can not approximate high-oscillation functions.



f crosses more than *g*

و) مشکل ناپدید شدن گرادیان: مشکل ناپدید شدن گرادیان در طول آموزش شبکه‌های عصبی عمیق زمانی رخ می‌دهد که گرادیان‌های مورد استفاده برای به‌روزرسانی وزن‌ها بسیار کوچک می‌شوند، زیرا به لایه‌های قبلی منتشر می‌شوند. این امر به‌روزرسانی وزن‌ها را در این لایه‌ها کاهش می‌دهد و به طور موثر شبکه را از یادگیری وابستگی‌های دوربرد جلوگیری می‌کند. به عبارت دیگر، این اتفاق می افتد زیرا:

۱. توابع فعال سازی سنتی مانند سیگموئید و tanh دارای گرادیان هایی هستند که برای ورودی های مثبت یا منفی بزرگ بسیار کوچک هستند.
۲. هنگامی که این شیب های کوچک در حین انتشار پس از انتشار چند بار ضرب می شوند، حتی کوچکتر می شوند.
۳. که منجر به یادگیری بسیار کند در لایه های اولیه می شود.

چگونه ReLU کمک می کند؟ توابع فعال سازی مانند ReLU (Rectified Linear Unit) مشکل گرادیان ناپدید شدن را کاهش می دهند زیرا مشتق آن به جای کاهش یافتن، ثابت است (۱ برای ورودی های مثبت و ۰ برای ورودی های منفی). برخلاف توابع سیگموئید یا tanh، که مقادیر ورودی را در محدوده کوچکی قرار می‌دهند و مشتقات کمتر از ۱ دارند، ReLU به گرادیان‌ها اجازه می‌دهد بدون کوچک کردن آنها عبور کنند و تضمین می‌کند که لایه‌های نزدیک‌تر به ورودی در طول آموزش به‌روزرسانی‌های قابل توجهی دریافت می‌کنند.

به عبارت دیگر، ReLU به حل این مشکل به شیوه زیر کمک می کند:

- داشتن گرادیان ثابت ۱ برای همه ورودی های مثبت
- محاسبه مشتق ساده (۱ برای $x > 0$، ۰ برای $x < 0$)
- عدم اشباع(saturation) در ناحیه مثبت
- فعال سازی Sparse (مقادیر منفی را صفر می کند)

ز) غلط. SGD با اندازه دسته‌ای کوچک معمولاً دارای واریانس بالاتری در به‌روزرسانی‌ها است و در حالی که به‌روزرسانی‌های مکرر را انجام می‌دهد، همگرایی سریع‌تری را نسبت به mini-batch gradient descent تضمین نمی‌کند. به عبارت دیگر، در حالی که اندازه‌های دسته‌ای کوچکتر در SGD منجر به به‌روزرسانی‌های مکرر می‌شوند، واریانس بالاتری را در برآورد گرادیان ایجاد می‌کنند که می‌تواند منجر به نوسانات و هم‌گرایی ناپایدار شود.

ی) از فرمول های زیر استفاده میکنیم:

First Moment (Momentum)

- **Definition**: The first moment, m_t, represents a moving average of past gradients. It builds "velocity" that propels learning in a consistent direction.
- **Update Rule**:

$$m_{t+1} = \beta_1 m_t + (1-\beta_1)\nabla_w J(w_t)$$

$$w_{t+1} = w_t - \eta m_{t+1}$$

where:

  - β_1: Decay rate, usually 0.9 or 0.99, which controls the weight of past gradients.
  - η: Learning rate.
- **Why Use First Momentum?**
  - Inspired by the idea of rolling momentum, it smooths and accelerates learning by sustaining direction from prior gradients.
  - This type of momentum is ideal for traversing narrow valleys or regions where standard gradient descent would oscillate.

داریم:

- $(y = 0.3x^4 - 0.1x^3 - 2x^2 - 0.8x)$
- Initial point: $(x_0 = -2.8)$
- Learning rate: $(\gamma = 0.05)$
- Momentum coefficient: $(\mu = 0.7)$
- Number of iterations: $(i = 2)$
- Initial momentum: $(m_0 = 0)$

ابتدا مشتق y را نسبت به x پیدا میکنیم.

$$\frac{dy}{dx} = y' = 1.2x^3 - 0.3x^2 - 4x - 0.8$$

محاسبات در هر مرحله به شکل زیر است:

$$g_i = 1.2x_i^3 - 0.3x_i^2 - 4x_i - 0.8$$

$$m_i = \mu m_{i-1} + (1-\mu)g_i$$

$$x_i = x_{i-1} - \gamma m_i$$

$$y_i = 0.3x_i^4 - 0.1x_i^3 - 2x_i^2 - 0.8x_i$$

توجه: اصلاح بایاس در هر مرحله انجام نشده است.

تکرار ۰ (وضعیت ابتدایی):

$$m_0 = 0 \quad , \quad x_0 = -2.8 \quad , \quad y_0 = 7.1949$$

تکرار ۱:

1. محاسبه مشتق (g_1) at $(x_0 = -2.8)$:

$$g_1 = 1.2(-2.8)^3 - 0.3(-2.8)^2 - 4(-2.8) - 0.8 = -18.2944$$

2. به روز رسانی ممان (m_1):

$$m_1 = \mu m_0 + (1-\mu)g_1 = 0.7 \times 0 + 0.3 \times (-18.2944) = -5.4883$$

3. به روز رسانی (x_1):

$$x_1 = x_0 - \gamma m_1 = -2.8 - 0.05 \times (-5.4883) = -2.5256$$

4. محاسبه (y_1):

$$y_1 = 0.3(-2.5256)^4 - 0.1(-2.5256)^3 - 2(-2.5256)^2 - 0.8(-2.5256) \approx 3.0801$$

تکرار ۲:

1. محاسبه مشتق (g_2) at $(x_1 = -2.5256)$:

$$g_2 = 1.2(-2.5256)^3 - 0.3(-2.5256)^2 - 4(-2.5256) - 0.8 \approx -11.9428$$

2. به روز رسانی ممان (m_2):

$$m_2 = \mu m_1 + (1-\mu)g_2 = 0.7 \times (-5.4883) + 0.3 \times (-11.9428) = -7.4247$$

3. به روز رسانی (x_2):

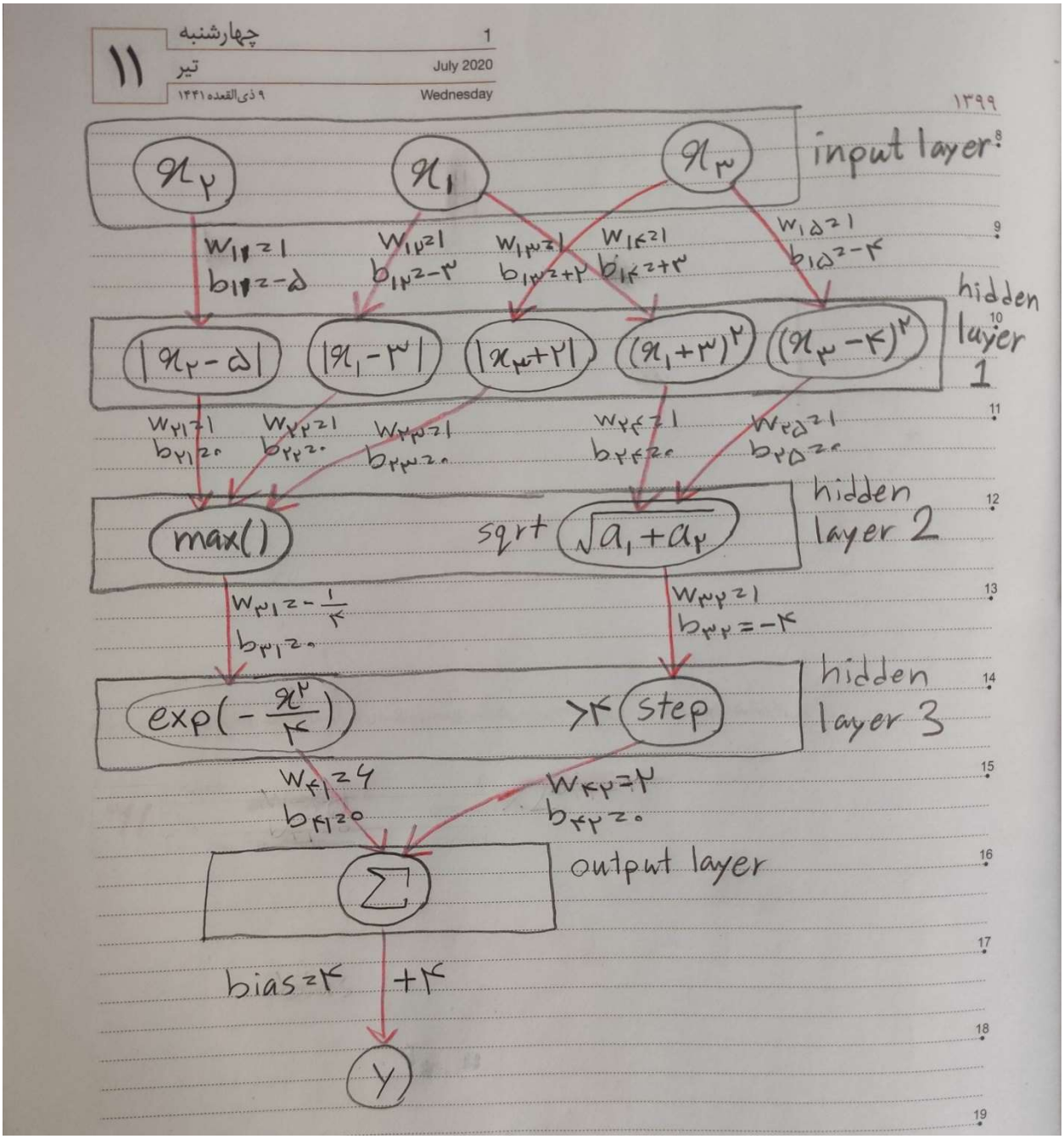
$$x_2 = x_1 - \gamma m_2 = -2.5256 - 0.05 \times (-7.4247) = -2.1544$$

4. محاسبه (y_2):

$$y_2 = 0.3(-2.1544)^4 - 0.1(-2.1544)^3 - 2(-2.1544)^2 - 0.8(-2.1544) \approx -0.0968$$

سوال دوم

الف)



همچین جزییات بیشتر هر نورون در ادامه آمده است:

چون بیشتر اسم ها به انگلیسی است ادامه سوال را به انگلیسی نگارش میکنیم تا مفهوم تر باشد.

input Layer:

شامل ورودی ها (x1, x2, x3) است.

1st hidden layer:

- Absolute value neurons (abs1, abs2, abs3: قدر مطلق ها):
    - f_net: w * x + b
    - f_act: |x|
    - f_out: identity
    - Parameters: w1=1, b1=-3 for abs1; w1=1, b1=-5 for abs2; w1=1, b1=2 for abs3
- Square neurons (توان دو ها):
    - f_net: w * x + b
    - f_act: x^2
    - f_out: identity
    - Parameters: w1=1, b1=3, w2=1, b2=-4

2nd hidden Layer:

- Maximum neuron (ماکزیمم میگیرد):
    - f_net: takes three inputs (w1*a1+b1 , w2*a2 + b2 , w3*a3 + b3)
    - f_act: max function (max(z1, z2, z3) + bias)
    - f_out: identity
    - Parameters: w1=w2=w3=1, b1=b2=b3=0, bias=0
- Square root neuron (جذر میگیرد):
    - f_net: takes two inputs (w1*a1 + w2*a2 + bias)
    - f_act: sqrt function
    - f_out: identity
    - Parameters: w1=w2=1, bias=0

3rd hidden Layer:

- Exponential neuron (توان نمایی):
    - f_net: $-x^2/4$
    - f_act: exp(x)
    - f_out: identity
    - Parameters: w=-1/4, bias=0
- Step function neuron:

- f_net: w*x + bias
- f_act: step function
- f_out: identity
- Parameters: w=1, bias=-4

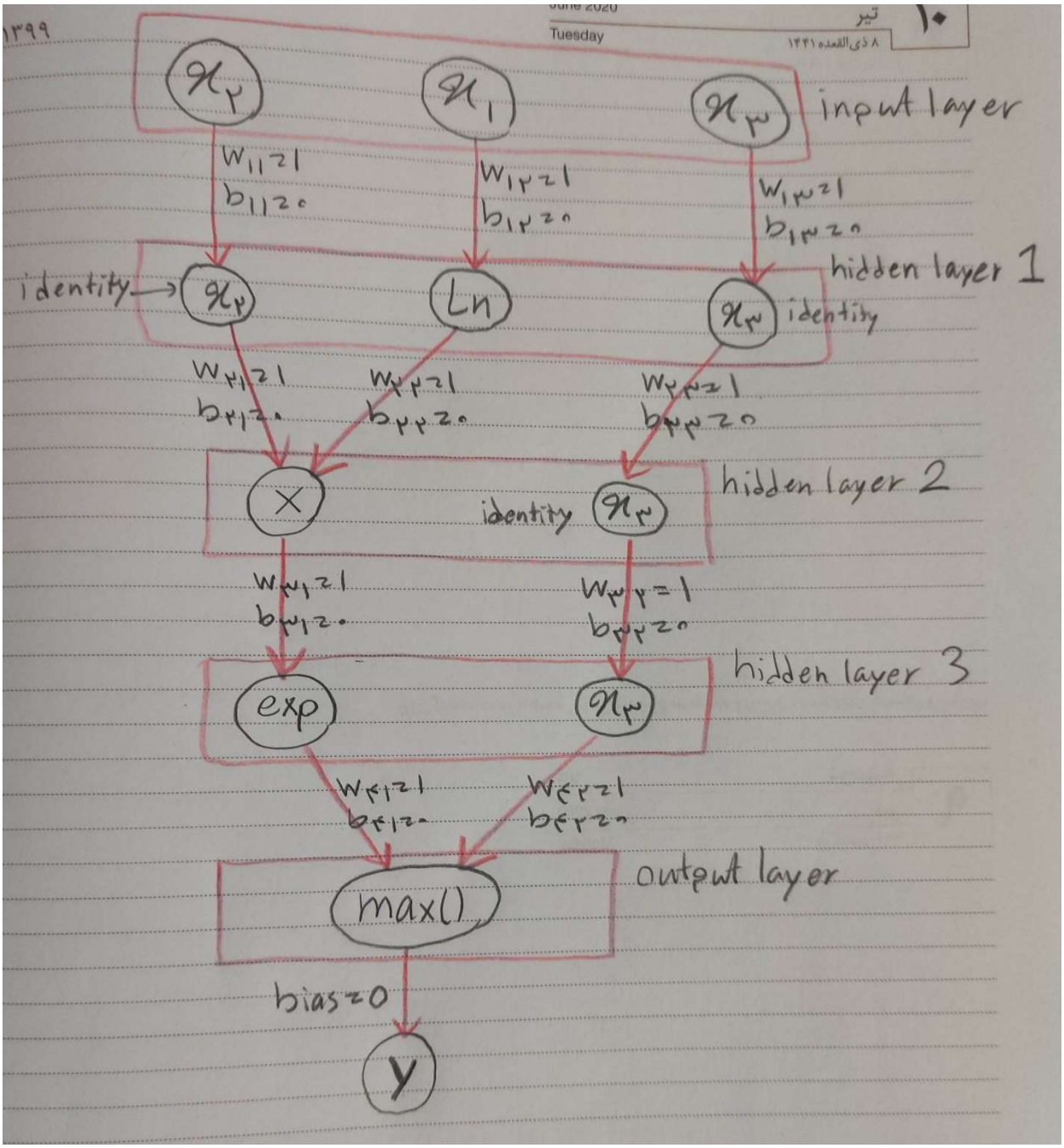
Output Layer:

- Sum neuron(جمع میکنیم):
  - f_net: w1*x1 + w2*x2 + b
  - f_act: identity
  - f_out: identity
  - Parameters: w1=6, w2=2, bias=4

ب) نکته: برای محاسبه $x_1^{x_2}$ از خاصیت زیر استفاده میکنیم:

$$x_1^{x_2} = e^{Ln(x_1^{x_2})} = e^{x_2 \times Ln(x_1)}$$

پس ابتدا لگاریتم میگیریم، سپس ضرب میکنیم و در نهایت از exp استفاده میکنیم.



همچین جزییات بیشتر هر نورون در ادامه آمده است:

چون بیشتر اسم ها به انگلیسی است ادامه سوال را به انگلیسی نگارش میکنیم تا مفهوم تر باشد.

input Layer:

شامل ورودی ها (x1, x2, x3) است.

1st hidden layer:

- Natural Log neuron (ln برای x1):
  - f_net: w*x + b
  - f_act: ln(x)
  - f_out: identity
  - Parameters: w=1, b=0
- Identity neurons (x2, x3 تابع همانی برای):
  - f_net: w*x + b
  - f_act: x (identity)
  - f_out: identity
  - Parameters: w=1, b=0

2nd hidden layer:

- Multiplication neuron (ضرب):
  - f_net: a1=w1*x1+b1 , a2=w2**x2 + b2
  - f_act: multiplication(a1 * a2)
  - f_out: identity
  - Parameters: w1=1, w2=1, b1=0, b2=0
- Identity neurons (x3 تابع همانی برای):
  - f_net: w*x + b
  - f_act: x (تابع همانی)
  - f_out: identity
  - Parameters: w=1, b=0

3rd hidden layer

- Exponential neuron (exp):
  - f_net: w*x + b
  - f_act: exp(x)
  - f_out: identity

- Parameters: w=1, b=0

- Identity neurons (x3 تابع همانی برای):
  - f_net: w*x + b
  - f_act: x (تابع همانی)
  - f_out: identity
  - Parameters: w=1, b=0

Output layer:

- Maximum neuron (max):
  - f_net: takes two inputs (w1*a1+b1 , w2*a2 + b2)
  - f_act: max function (max(a1, a2) + bias)
  - f_out: identity
  - Parameters: w1=w2=1, b1=b2=0, bias=0

سوال سوم

پنجشنبه ۱۶ بهمن ۲۱ جمادی‌الثانیه ۱۴۴۲ — 4 February 2021 Thursday — ۱۳۹۹

$$z_1 = W_1 x^{(i)} + b_1 \;,\; a_1 = ReLU(z_1) = \max(0, z_1)$$

$$z_2 = W_2 a_1 + b_2 \;,\; \hat{y}^{(i)} = \sigma(z_2) = \frac{1}{1+e^{-z_2}}$$

$$L^{(i)} = y^{(i)} \log \hat{y}^{(i)} + (1-y^{(i)}) \log(1-\hat{y}^{(i)})$$

$$J = -\frac{1}{m} \sum_{i=1}^{m} L^{(i)} \;,\; x^{(i)} \in \mathbb{R}^{D_x \times 1} \;,\; z_1 \in \mathbb{R}^{D_{a_1} \times 1}$$

---

(۱)

$$W_1 \in \mathbb{R}^{D_{a_1} \times D_x} \;,\; b_1 \in \mathbb{R}^{D_{a_1} \times 1} \;,\; W_2 \in \mathbb{R}^{1 \times D_{a_1}}$$

$$b_2 \in \mathbb{R}^{1 \times 1}$$

اگر بخواهیم vectorize کنیم شکل ابعاد وزن‌ها و بایاس‌ها تغییری نخواهد کرد و فقط ابعاد ورودی و خروجی یعنی X و Y تغییر خواهد کرد. (unchanged)

جمعه ۱۷ بهمن ۲۲ جمادی‌الثانیه ۱۴۴۲ — 5 February 2021 Friday

* توجه کنید که بایاس‌ها broadcast خواهند شد برای جمع شدن.

بعد از vectorizing ابعاد ورودی و خروجی به شکل زیری شود.

$$X \in \mathbb{R}^{D_x \times m} \;,\; Y \in \mathbb{R}^{1 \times m}$$

---

(۲)

$$\frac{\partial J}{\partial \hat{y}^{(i)}} = \frac{\partial J}{\partial L^{(i)}} \times \frac{\partial L^{(i)}}{\partial \hat{y}^{(i)}} = \frac{-1}{m} \times \left[\frac{y^{(i)}}{\hat{y}^{(i)}} - \frac{1-y^{(i)}}{1-\hat{y}^{(i)}} \right]$$

$$= \frac{-1}{m} \times \left[\frac{y^{(i)} - y^{(i)}\hat{y}^{(i)} - \hat{y}^{(i)} + y^{(i)}\hat{y}^{(i)}}{\hat{y}^{(i)}(1-\hat{y}^{(i)})} \right] = \frac{-1}{m} \times \left[\frac{y^{(i)} - \hat{y}^{(i)}}{\hat{y}^{(i)}(1-\hat{y}^{(i)})} \right]$$

۱۳۹۹

$$\delta_1^{(i)} = \frac{-1}{m}\left[\frac{y^{(i)} - \hat{y}^{(i)}}{\hat{y}^{(i)}(1-\hat{y}^{(i)})}\right] \xrightarrow{\text{vectorized form}} \frac{\partial J}{\partial \hat{y}} = \frac{-1}{m}\left[\frac{Y - \hat{Y}}{\hat{Y}(1-\hat{Y})}\right]$$

(۳)
$$\frac{\partial \hat{y}^{(i)}}{\partial z_2} = \frac{\partial}{\partial z_2}\left(\sigma(z_2)\right) = \sigma(z_2)\left(1-\sigma(z_2)\right)$$

$$\delta_2^{(i)} = \sigma(z_2)\left(1-\sigma(z_2)\right)$$

(۴)
$$\frac{\partial z_2}{\partial a_1} = \frac{\partial}{\partial a_1}\left(W_2 a_1 + b_2\right) = W_2$$

$$\delta_3^{(i)} = W_2$$

(۵)
$$\frac{\partial a_1}{\partial z_1} = \frac{\partial}{\partial z_1}\left(\text{ReLU}(z_1)\right) = \mathbb{I}(z_1 > 0) = \begin{cases} 1 & \text{if } z_1 > 0 \\ 0 & \text{if } z_1 \le 0 \end{cases}$$

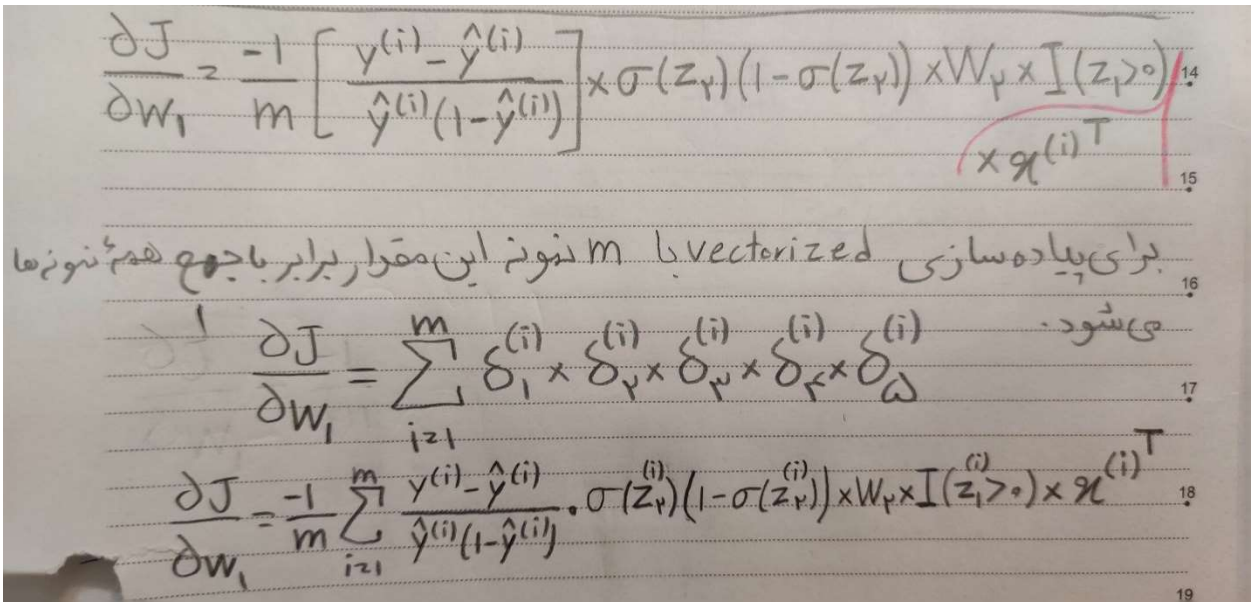
$\mathbb{I}$: Indicator function

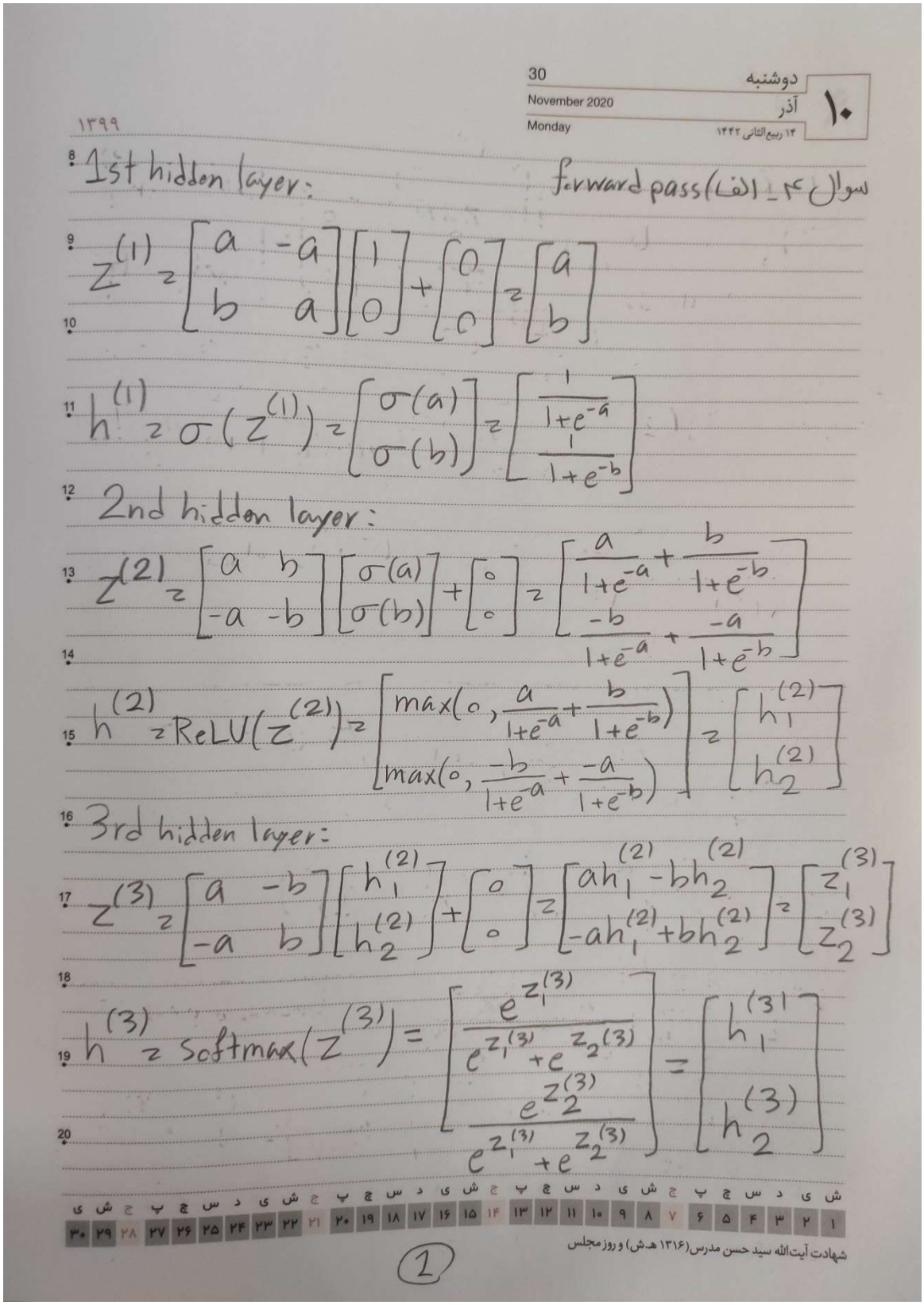
$$\delta_4^{(i)} = \mathbb{I}(z_1 > 0)$$

(۶)
$$\frac{\partial z_1}{\partial W_1} = \frac{\partial}{\partial W_1}\left(W_1 x^{(i)} + b_1\right) = x^{(i)T} \Rightarrow \delta_5^{(i)} = x^{(i)T}$$

(۷)
$$\frac{\partial J}{\partial W_1} = \frac{\partial J}{\partial \hat{y}^{(i)}} \times \frac{\partial y^{(i)}}{\partial z_2} \times \frac{\partial z_2}{a_1} \times \frac{\partial a_1}{\partial z_1} \times \frac{\partial z_1}{\partial W_1}$$

$$\frac{\partial J}{\partial W_1} = \delta_1^{(i)} \times \delta_2^{(i)} \times \delta_3^{(i)} \times \delta_4^{(i)} \times \delta_5^{(i)}$$


$\frac{\partial J}{\partial w_1} = \frac{-1}{m}\left[\frac{y^{(i)} - \hat{y}^{(i)}}{\hat{y}^{(i)}(1-\hat{y}^{(i)})}\right] \times \sigma(z_2)(1-\sigma(z_2)) \times W_2 \times \mathbb{I}(z_1>0) \times x^{(i)T}$
برای پیاده‌سازی vectorized با m نمونه این مقدار برابر با جمع همهٔ نمونه‌ها می‌شود.
$\frac{\partial J}{\partial w_1} = \sum_{i=1}^{m} \delta_1^{(i)} \times \delta_2^{(i)} \times \delta_3^{(i)} \times \delta_4^{(i)} \times \delta_5^{(i)}$
$\frac{\partial J}{\partial w_1} = \frac{-1}{m}\sum_{i=1}^{m} \frac{y^{(i)} - \hat{y}^{(i)}}{\hat{y}^{(i)}(1-\hat{y}^{(i)})} \cdot \sigma(z_2^{(i)})(1-\sigma(z_2^{(i)})) \times W_2 \times \mathbb{I}(z_1^{(i)}>0) \times x^{(i)T}$



سوال ۴ـ الف) forward pass

1st hidden layer:

$$z^{(1)} = \begin{bmatrix} a & -a \\ b & a \end{bmatrix}\begin{bmatrix} 1 \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \end{bmatrix} = \begin{bmatrix} a \\ b \end{bmatrix}$$

$$h^{(1)} = \sigma(z^{(1)}) = \begin{bmatrix} \sigma(a) \\ \sigma(b) \end{bmatrix} = \begin{bmatrix} \frac{1}{1+e^{-a}} \\ \frac{1}{1+e^{-b}} \end{bmatrix}$$

2nd hidden layer:

$$z^{(2)} = \begin{bmatrix} a & b \\ -a & -b \end{bmatrix}\begin{bmatrix} \sigma(a) \\ \sigma(b) \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \end{bmatrix} = \begin{bmatrix} \frac{a}{1+e^{-a}} + \frac{b}{1+e^{-b}} \\ \frac{-b}{1+e^{-a}} + \frac{-a}{1+e^{-b}} \end{bmatrix}$$

$$h^{(2)} = \mathrm{ReLU}(z^{(2)}) = \begin{bmatrix} \max(0, \frac{a}{1+e^{-a}} + \frac{b}{1+e^{-b}}) \\ \max(0, \frac{-b}{1+e^{-a}} + \frac{-a}{1+e^{-b}}) \end{bmatrix} = \begin{bmatrix} h_1^{(2)} \\ h_2^{(2)} \end{bmatrix}$$

3rd hidden layer:

$$z^{(3)} = \begin{bmatrix} a & -b \\ -a & b \end{bmatrix}\begin{bmatrix} h_1^{(2)} \\ h_2^{(2)} \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \end{bmatrix} = \begin{bmatrix} ah_1^{(2)} - bh_2^{(2)} \\ -ah_1^{(2)} + bh_2^{(2)} \end{bmatrix} = \begin{bmatrix} z_1^{(3)} \\ z_2^{(3)} \end{bmatrix}$$

$$h^{(3)} = \mathrm{Softmax}(z^{(3)}) = \begin{bmatrix} \frac{e^{z_1^{(3)}}}{e^{z_1^{(3)}} + e^{z_2^{(3)}}} \\ \frac{e^{z_2^{(3)}}}{e^{z_1^{(3)}} + e^{z_2^{(3)}}} \end{bmatrix} = \begin{bmatrix} h_1^{(3)} \\ h_2^{(3)} \end{bmatrix}$$

output layer:

$$z^{(4)} = \begin{bmatrix} a & -b \\ 0.5a & a \end{bmatrix} \begin{bmatrix} \frac{e^{z_1^{(3)}}}{e^{z_1^{(3)}} + e^{z_2^{(3)}}} \\ \frac{e^{z_2^{(3)}}}{e^{z_1^{(3)}} + e^{z_2^{(3)}}} \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \end{bmatrix} = \begin{bmatrix} ah_1^{(3)} - bh_2^{(3)} \\ 0.5ah_1^{(3)} + ah_2^{(3)} \end{bmatrix} \begin{matrix} z_1^{(4)} \\ z_2^{(4)} \end{matrix}$$

$$a^{(4)} = h^{(4)} = \text{identity}(z^{(4)}) = \begin{bmatrix} z_1^{(4)} \\ z_2^{(4)} \end{bmatrix} = \begin{bmatrix} ah_1^{(3)} - bh_2^{(3)} \\ 0.5ah_1^{(3)} + ah_2^{(3)} \end{bmatrix} = \begin{bmatrix} h_1^{(4)} \\ h_2^{(4)} \end{bmatrix}$$

($a^{(4)}$ خواسته شده است همان خروجی $h^{(4)}$ است؛ ما به جای a از h استفاده کردیم.)

$$\boxed{a^{(4)} = h^{(4)}}$$

ب) محاسبهٔ خطای E

$$E = \frac{1}{2}\left((a_1^{(4)} - 0)^2 + (a_2^{(4)} - 1)^2\right)$$

$$E = \frac{1}{2}\left[(h_1^{(4)} - 0)^2 + (h_2^{(4)} - 1)^2\right)$$

backward pass (2

$$\frac{\partial E}{\partial w^{(4)}} = \frac{\partial E}{\partial h^{(4)}} \times \frac{\partial h^{(4)}}{\partial z^{(4)}} \times \frac{\partial z^{(4)}}{\partial w^{(4)}} \qquad dz^{(4)}$$

$$\frac{\partial E}{\partial w^{(4)}} = \begin{bmatrix} h_1^{(4)} \\ h_2^{(4)} - 1 \end{bmatrix} \times 1 \times h^{(3)T} = \begin{bmatrix} h_1^{(4)} \\ h_2^{(4)} - 1 \end{bmatrix} \begin{bmatrix} h_1^{(3)} & h_2^{(3)} \end{bmatrix}$$

$$\frac{\partial E}{\partial b^{(4)}} = dz^{(4)} \times \frac{\partial z^{(4)}}{\partial b^{(4)}} = dz^{(4)}$$

$$\frac{\partial E}{\partial b^{(4)}} = \frac{\partial E}{\partial h^{(4)}} \times \frac{\partial h^{(4)}}{\partial z^{(4)}} \times \frac{\partial z^{(4)}}{\partial b^{(4)}} = \begin{bmatrix} h_1^{(4)} \\ h_2^{(4)} - 1 \end{bmatrix} \times 1 \times 1 = \begin{bmatrix} h_1^{(4)} \\ h_2^{(4)} - 1 \end{bmatrix}$$

$dz^{(3)}$

$$\frac{\partial E}{\partial W^{(3)}} = \frac{\partial E}{\partial h^{(4)}} \times \frac{\partial h^{(4)}}{\partial z^{(4)}} \times \frac{\partial z^{(4)}}{\partial h^{(3)}} \times \frac{\partial h^{(3)}}{\partial z^{(3)}} \times \frac{\partial z^{(3)}}{\partial W^{(3)}}$$

$$\frac{\partial E}{\partial W^{(3)}} = \begin{bmatrix} h_1^{(4)} \\ h_2^{(4)} \end{bmatrix} \times 1 \times W^{(4)} \times \frac{\partial h^{(3)}}{\partial z^{(3)}} \times h^{(2)T}$$

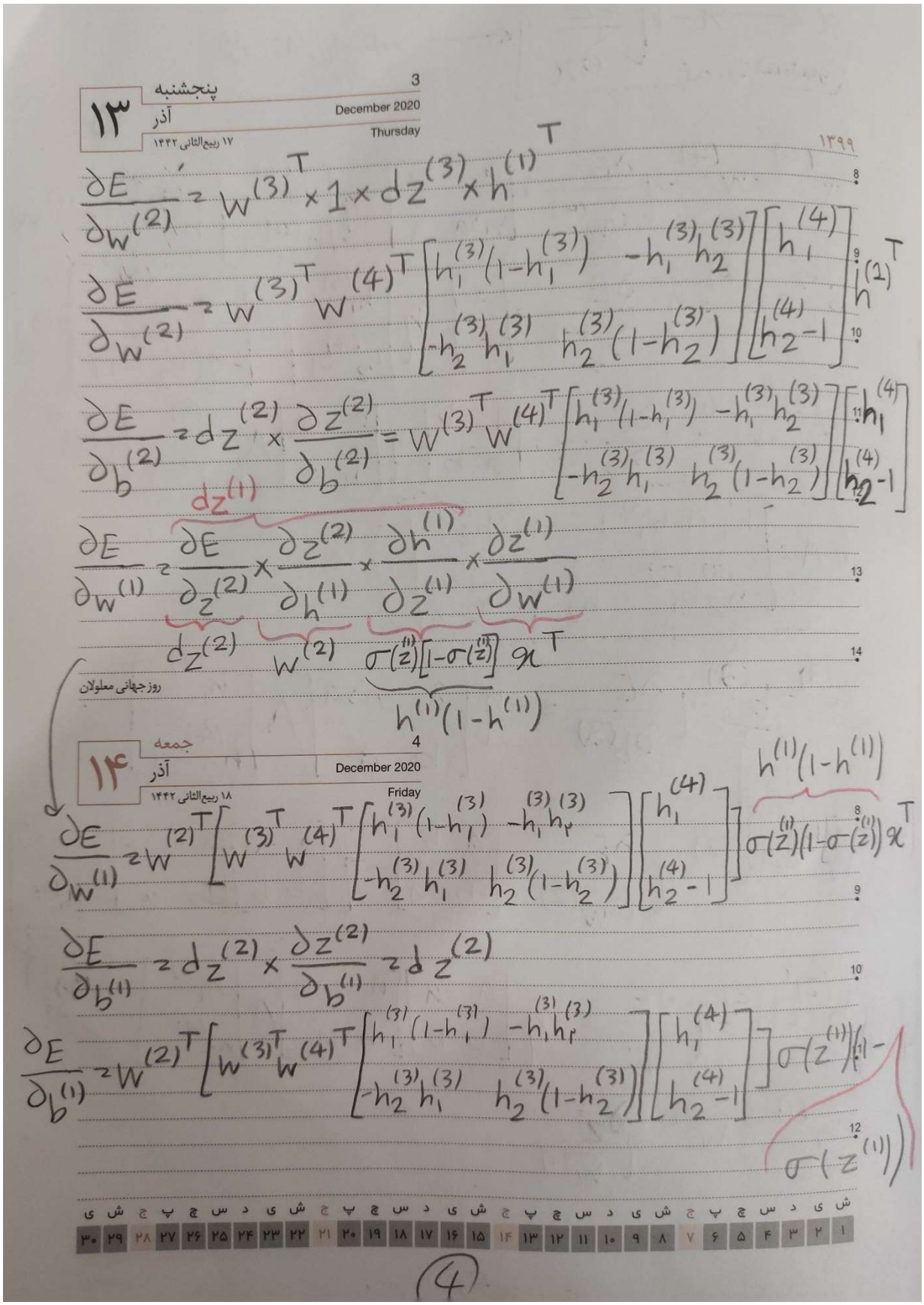
$$\begin{cases} h^{(3)} = \text{Softmax}(z^{(3)}) \\ \frac{\partial h_i^{(3)}}{\partial z_j^{(3)}} = \begin{cases} h_i^{(3)}(1-h_i^{(3)}) & i=j \\ -h_i^{(3)} h_j^{(3)} & i \neq j \end{cases} \rightarrow \frac{\partial h^{(3)}}{\partial z^{(3)}} = \begin{bmatrix} h_1^{(3)}(1-h_1^{(3)}) & -h_1^{(3)} h_2^{(3)} \\ -h_2^{(3)} h_1^{(3)} & h_2^{(3)}(1-h_2^{(3)}) \end{bmatrix}_{2\times 2} \end{cases}$$

$$\frac{\partial E}{\partial W_1^{(3)}} = W^{4T} \begin{bmatrix} h_1^{(3)}(1-h_1^{(3)}) & -h_1^{(3)} h_2^{(3)} \\ -h_2^{(3)} h_1^{(3)} & h_2^{(3)}(1-h_2^{(3)}) \end{bmatrix} \begin{bmatrix} h_1^{(4)} \\ h_2^{(4)} - 1 \end{bmatrix} h^{2T}$$

$$\frac{\partial E}{\partial b^{(3)}} = dz^{(3)} \times \frac{\partial z^{(3)}}{\partial b^{(3)}} = W^{4T} \begin{bmatrix} h_1^{(3)}(1-h_1^{(3)}) & -h_1^{(3)} h_2^{(3)} \\ -h_2^{(3)} h_1^{(3)} & h_2^{(3)}(1-h_2^{(3)}) \end{bmatrix} \begin{bmatrix} h_1^{(4)} \\ h_2^{(4)} - 1 \end{bmatrix}$$

$dz^{(2)}$

$$\frac{\partial E}{\partial W^{(2)}} = \underbrace{\frac{\partial E}{\partial h^{(3)}} \times \frac{\partial h^{(3)}}{\partial z^{(3)}}}_{dz^{(3)}} \times \underbrace{\frac{\partial z^{(3)}}{\partial h^{(2)}}}_{W^{(3)}} \times \underbrace{\frac{\partial h^{(2)}}{\partial z^{(2)}}}_{1} \times \underbrace{\frac{\partial z^{(2)}}{\partial W^{(2)}}}_{h^{(1)T}}$$


پنجشنبه
۱۳
آذر
۱۷ ربیع‌الثانی ۱۴۴۲
3
December 2020
Thursday
۱۳۹۹
روز جهانی معلولان
جمعه
۱۴
آذر
۱۸ ربیع‌الثانی ۱۴۴۲
4
December 2020
Friday
(4)

pg. 17

$$x' \leftarrow x - \eta \frac{\partial E}{\partial x}$$ gradient decent ← از این قانون استفاده می‌کنیم

شنبه ۱۵ آذر ۱۳۹۹ — Saturday 5 December 2020 — ۱۹ ربیع‌الثانی ۱۴۴۲

$$W^{(4)'} = W^{(4)} - \eta \frac{\partial E}{\partial W^{(4)}} = \begin{bmatrix} a & -b \\ 0{,}5a & a \end{bmatrix} - \eta \begin{bmatrix} h_1^{(4)} \\ h_2^{(4)} - 1 \end{bmatrix} h^{(3)T}$$

$$b^{(4)'} = b^{(4)} - \eta \frac{\partial E}{\partial b^{(4)}} = \begin{bmatrix} 0 \\ 0 \end{bmatrix} - \eta \begin{bmatrix} h_1^{(4)} \\ h_2^{(4)} - 1 \end{bmatrix} = \begin{bmatrix} -\eta\, h_1^{(4)} \\ -\eta (h_2^{(4)} - 1) \end{bmatrix}$$

$$W^{(3)'} = W^{(3)} - \eta \frac{\partial E}{\partial W^{(3)}} = \begin{bmatrix} a & -b \\ -a & b \end{bmatrix} - \eta \begin{bmatrix} a & 0{,}5a \\ -b & a \end{bmatrix} A \begin{bmatrix} h_1^{(4)} \\ h_2^{(4)} - 1 \end{bmatrix} h^{(2)T}$$

$$A = \begin{bmatrix} h_1^{(3)}(1 - h_1^{(3)}) & -h_1^{(3)} h_2^{(3)} \\ -h_2^{(3)} h_1^{(3)} & h_2^{(3)}(1 - h_2^{(3)}) \end{bmatrix}$$

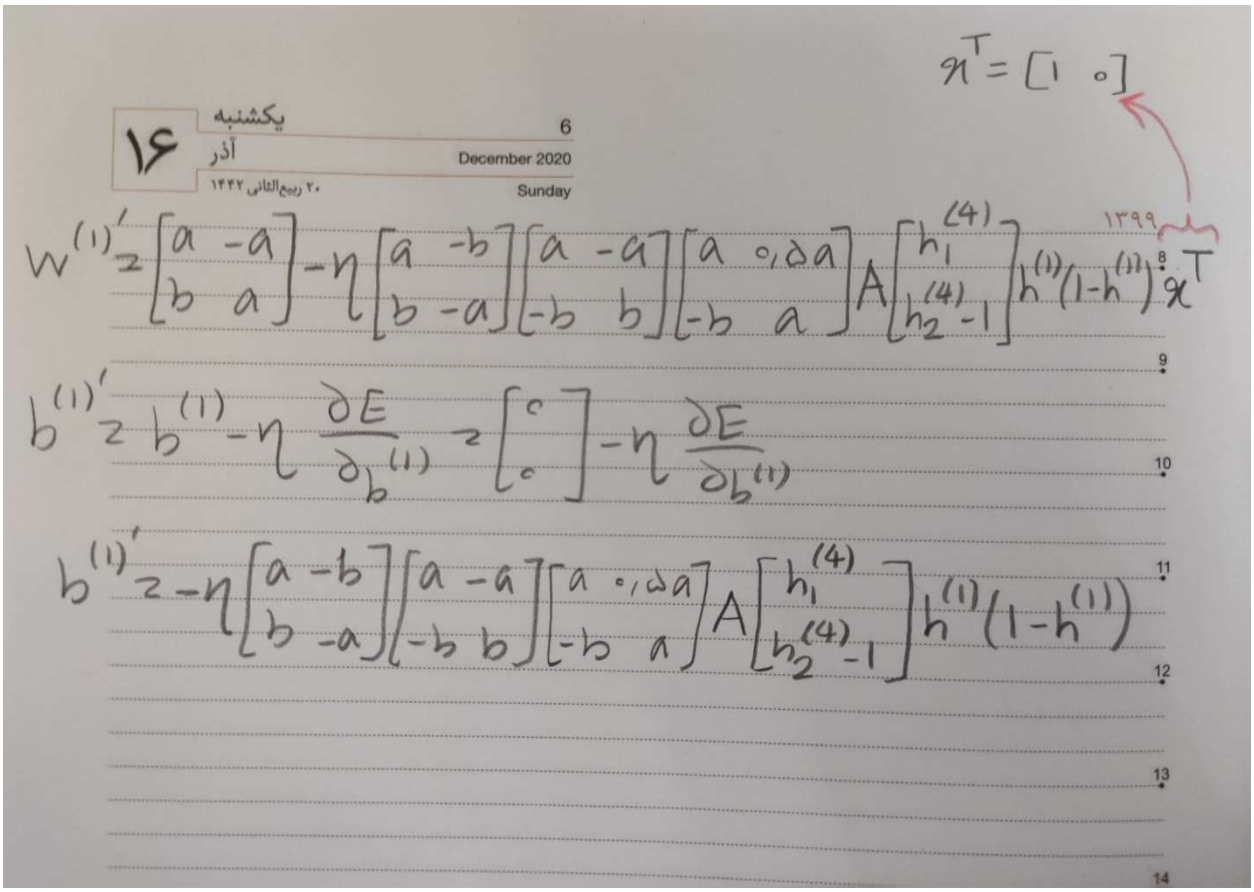
$$b^{(3)'} = \underbrace{b^{(3)}}_{\begin{bmatrix} 0 \\ 0 \end{bmatrix}} - \eta \frac{\partial E}{\partial b^{(3)}} = -\eta \begin{bmatrix} a & 0{,}5a \\ -b & a \end{bmatrix} A \begin{bmatrix} h_1^{(4)} \\ h_2^{(4)} - 1 \end{bmatrix}$$

$$W^{(2)'} = W^{(2)} - \eta \frac{\partial E}{\partial W^{(2)}} = \begin{bmatrix} a & b \\ -b & -a \end{bmatrix} - \eta \begin{bmatrix} a & -a \\ -b & b \end{bmatrix} \begin{bmatrix} a & 0{,}5a \\ -b & a \end{bmatrix} A \begin{bmatrix} h_1^{(4)} \\ h_2^{(4)} - 1 \end{bmatrix} \times h^{(1)T}$$

$$b^{(2)'} = b^{(2)} - \eta \frac{\partial E}{\partial b^{(2)}} = -\eta \begin{bmatrix} a & -a \\ -b & b \end{bmatrix} \begin{bmatrix} a & 0{,}5a \\ -b & a \end{bmatrix} A \begin{bmatrix} h_1^{(4)} \\ h_2^{(4)} - 1 \end{bmatrix}$$

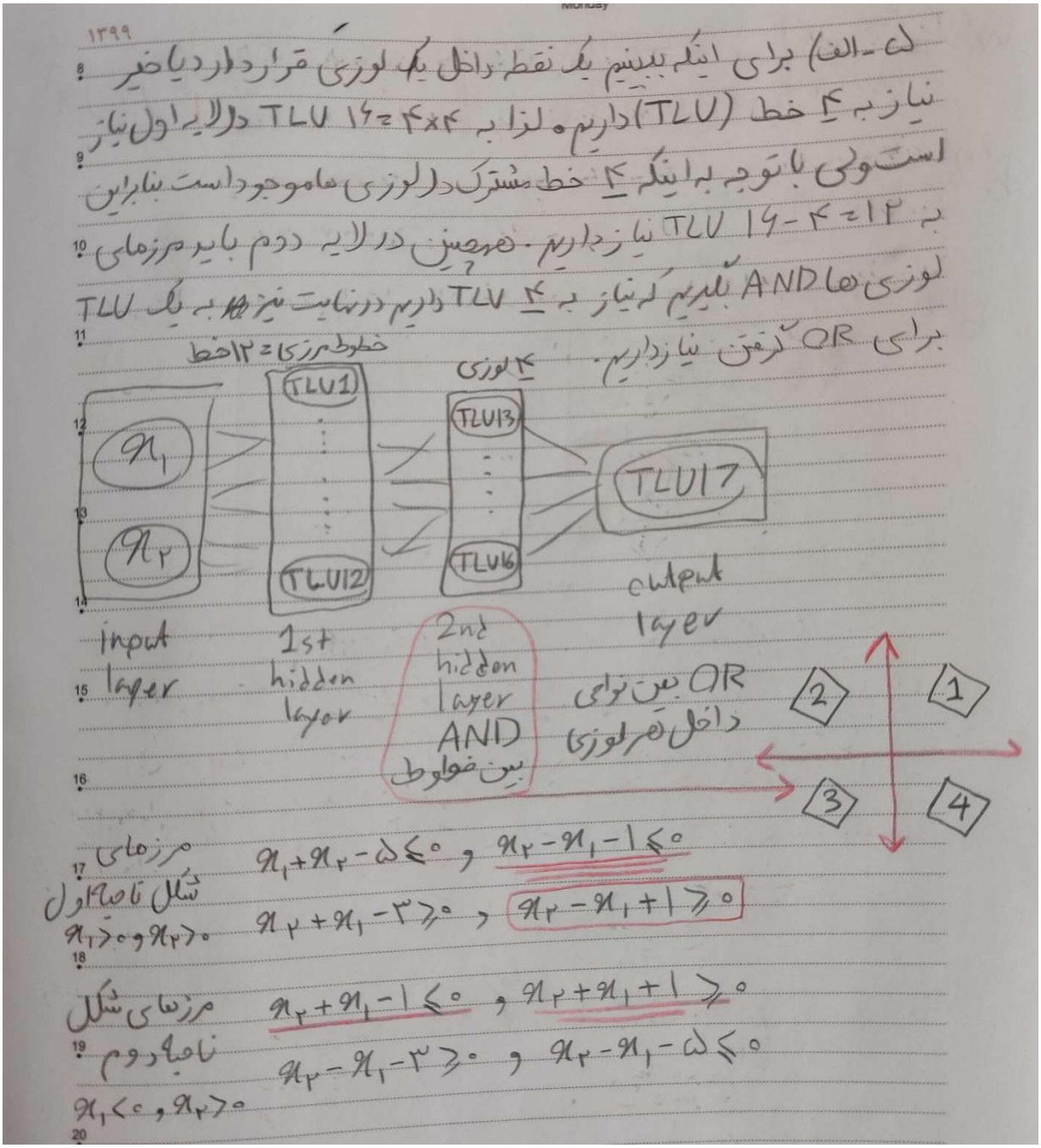
$$W^{(1)'} = W^{(1)} - \eta \frac{\partial E}{W^{(1)}}$$

5


یکشنبه
۱۶ آذر
۲۰ ربیع‌الثانی ۱۴۴۲
6
December 2020
Sunday
$x^T = [1 \quad 0]$
۱۳۹۹
$W^{(1)'} = W^{(1)} - \eta \begin{bmatrix} a & -b \\ b & -a \end{bmatrix} \begin{bmatrix} a & -a \\ -b & b \end{bmatrix} \begin{bmatrix} a & 0.5a \\ -b & a \end{bmatrix} A \begin{bmatrix} h_1^{(4)} \\ h_2^{(4)} - 1 \end{bmatrix} h^{(1)}(1-h^{(1)}) x^T$
$b^{(1)'} = b^{(1)} - \eta \frac{\partial E}{\partial b^{(1)}} = \begin{bmatrix} 0 \\ 0 \end{bmatrix} - \eta \frac{\partial E}{\partial b^{(1)}}$
$b^{(1)'} = -\eta \begin{bmatrix} a & -b \\ b & -a \end{bmatrix} \begin{bmatrix} a & -a \\ -b & b \end{bmatrix} \begin{bmatrix} a & 0.5a \\ -b & a \end{bmatrix} A \begin{bmatrix} h_1^{(4)} \\ h_2^{(4)} - 1 \end{bmatrix} h^{(1)}(1-h^{(1)})$

سوال پنجم

الف)



سه‌شنبه ۱۸ آذر ۲۲ ربیع‌الثانی ۱۴۴۲
8 December 2020 Tuesday
۱۳۹۹

شکل ناحیه سوم
$x_1 < 0, x_2 < 0$

$x_2 + x_1 + 3 \leq 0$, $x_2 + x_1 + 5 \geq 0$

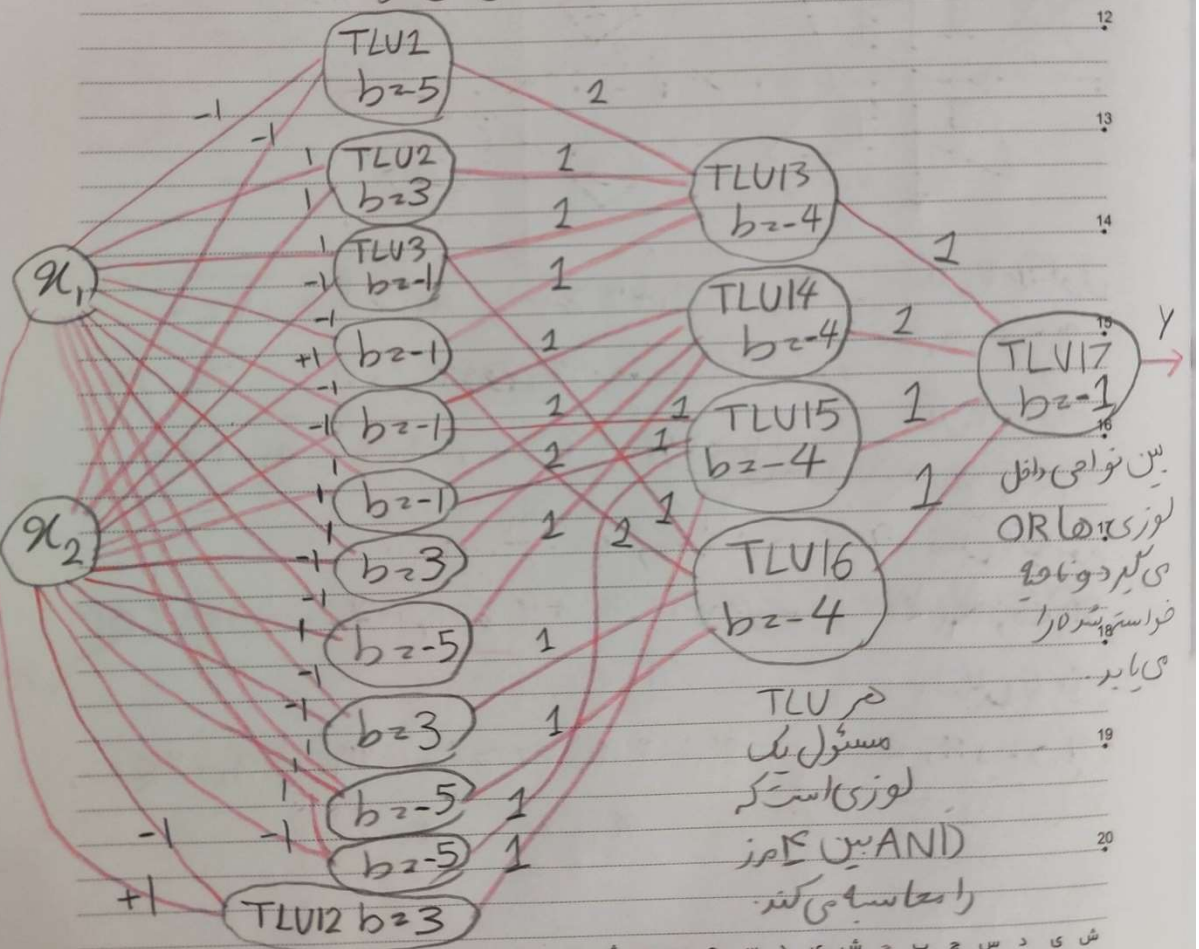
$x_2 - x_1 - 1 \leq 0$, $x_2 - x_1 + 1 \geq 0$

شکل ناحیه چهارم
$x_1 > 0, x_2 < 0$

$x_2 + x_1 - 1 \leq 0$, $x_2 + x_1 + 1 \geq 0$

$x_2 - x_1 + 3 \leq 0$, $x_2 - x_1 + 5 \geq 0$

چهار خط مشترک در بالا نشان داده شده است پس ما ۱۲ تا Unique boundary داریم. در ادامه threshold هر نورون داخل آن نوشته شده است.



بین نواحی داخل لوزی‌ها OR می‌گیرد و ناحیه خواسته شده را می‌یابد.

هر TLU مسئول یک لوزی است که AND بین ۴ مرز را محاسبه می‌کند.

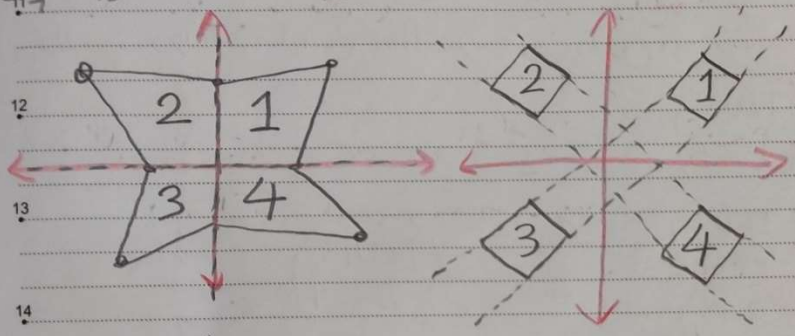
Unique decision boundaries

2

ب)

۱۳۹۹

ب) بله می توانیم با تغییر پارامترهای شکل الف ، ناحیه خواسته شده در ناحیه ب را به دست آوریم. شکل را مشابه الف به ۴ قسمت تقسیم می کنیم و با توجه به اینکه محورهای مختصات چهار منطقه مشترک تشکیل می دهند مشابه قسمت الف ۱۲ decision boundary خواهیم داشت که با AND کردن آنها چهار شکل تقسیم شده را می یابیم و سپس OR می کنیم.



شکل ناحیه اول

$x_2 - 3x_1 + 4 \geq 0$, $x_2 - \frac{1}{3}x_1 - 2 \leq 0$

$x_1 > 0, x_2 > 0$

$x_2 \geq 0$, $x_1 \geq 0$

شکل ناحیه دوم

$x_2 + 3x_1 + 4 \geq 0$, $x_2 + \frac{1}{3}x_1 - 2 \leq 0$

$x_1 < 0, x_2 > 0$

$x_1 \leq 0$, $x_2 \geq 0$

شکل ناحیه سوم

$-x_2 + 3x_1 + 4 \geq 0$, $-x_2 + \frac{1}{3}x_1 - 2 \leq 0$

$x_1 < 0, x_2 < 0$

$x_1 \leq 0$, $x_2 \leq 0$

شکل ناحیه چهارم

$-x_2 - 3x_1 + 4 \geq 0$, $-x_2 - \frac{1}{3}x_1 - 2 \leq 0$

$x_1 > 0$ و $x_2 < 0$

$x_1 \geq 0$, $x_2 \leq 0$

ش ی د س چ پ ج ش ی د س چ پ ج ش ی د س چ پ ج ش ی د س چ پ ج ش ی

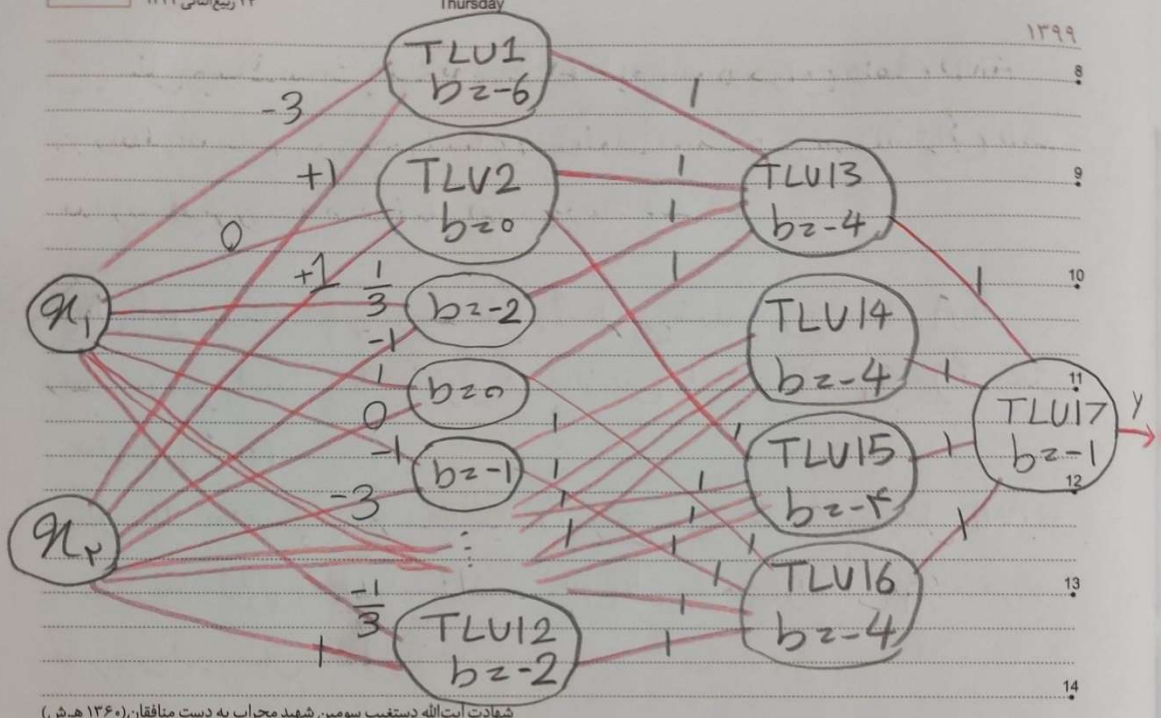
۱ ۲ ۳ ۴ ۵ ۶ ۷ ۸ ۹ ۱۰ ۱۱ ۱۲ ۱۳ ۱۴ ۱۵ ۱۶ ۱۷ ۱۸ ۱۹ ۲۰ ۲۱ ۲۲ ۲۳ ۲۴ ۲۵ ۲۶ ۲۷ ۲۸ ۲۹ ۳۰

تشکیل شورای عالی انقلاب فرهنگی به فرمان حضرت امام خمینی (رحمة‌الله‌علیه) (۱۳۶۳ هـ ش)

2 3

پنجشنبه ۲۰ آذر ۲۴ ربیع‌الثانی ۱۴۴۲
10 December 2020 Thursday

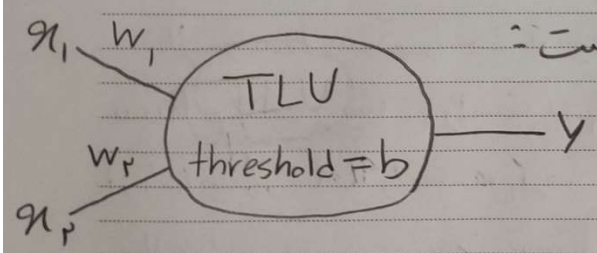
۱۳۹۹



شهادت آیت‌الله دستغیب سومین شهید محراب به دست منافقان(۱۳۶۰ هـ.ش)

جمعه ۲۱ آذر ۲۵ ربیع‌الثانی ۱۴۴۲
11 December 2020 Friday

ساختار هر TLU به شکل زیر است:



$$Z = w_1 x_1 + w_2 x_2 + b$$

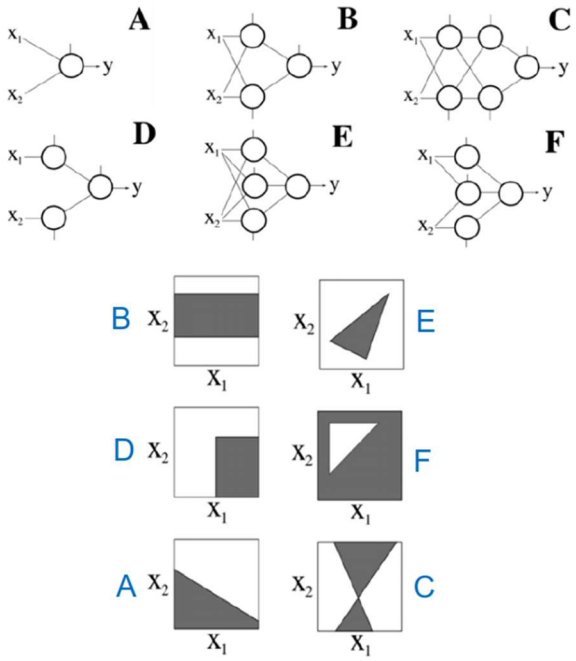
(step function)

$$y = \begin{cases} 1 & \text{if } z \geq 0 \\ 0 & \text{if } z < 0 \end{cases}$$

۱ ۲ ۳ ۴ ۵ ۶ ۷ ۸ ۹ ۱۰ ۱۱ ۱۲ ۱۳ ۱۴ ۱۵ ۱۶ ۱۷ ۱۸ ۱۹ ۲۰ ۲۱ ۲۲ ۲۳ ۲۴ ۲۵ ۲۶ ۲۷ ۲۸ ۲۹ ۳۰

4

ج) با توجه به اینکه هرکدام از شبکه ها فقط یک پاسخ را ایجاد کرده، هر ناحیه توسط شبکه ای که در کنارش در شکل زیر نوشته شده است ایجاد شده.



دلایل و علت ها در صفحه بعد آمده است.

ج) شکل وسط سمت چپ توسط D طراحی شده چون حاصل AND دو دسته بند خطی در دو راستای متفاوت است و دو مرز اشتراکی با هم ندارند هرنورون فقط با یک بعد در ارتباط است.

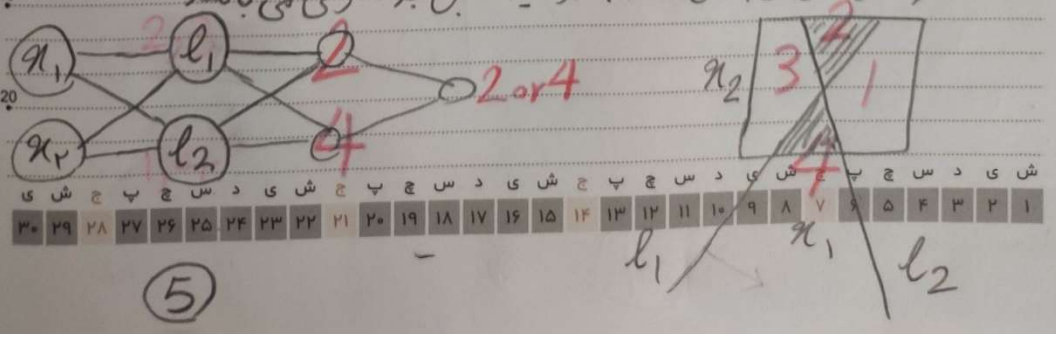
شکل پایین سمت چپ توسط شبکه A طراحی شده - زیرا شبکه A یک دسته بند خطی است و با بیشتر از یک خط نمی تواند جداسازی کند (یک دسته بند خطی ساده)

شکل بالا سمت چپ توسط شبکه B طراحی شده است. چون حاصل AND دو دسته بند خطی در یک راستا است. پس لازم است راستای x_2 به هردو نورون لایه پنهان منتقل باشد.

شکل راست بالا توسط E طراحی شده. چون سه راستای وابسته به x_1 و x_2 داریم که بین آنها AND گرفته شده است.

شکل وسط سمت راست توسط F طراحی شده است. چرا دو راستای مستقل و یک راستای وابسته به x_1 و x_2 داریم که با NOT AND گرفتن از آنها می توان ناحیه را مشخص کرد.

شکل پایین سمت راست توسط شبکه C طراحی شده است. چرا که تلاقی دو خط، چهار ناحیه ایجاد می کند که با دو لایه قابل جداسازی می باشد



(5)

سوال ششم

الف) ابتدا بیاید تابع مفاهیم ReLU وابر صفحه ها و Activation Patterns را بیان کنیم و سپس با استفاده از آنها به اثبات بپردازیم.

تابع فعال سازی ReLU:

تابع فعال سازی واحد خطی اصلاح شده (ReLU) به صورت زیر تعریف می شود:

$$\mathrm{ReLU}(x) = \max(0, x)$$

این تابع تکه ای x را وقتی x > 0 و در غیر این صورت ۰ را خروجی می دهد. ویژگی های کلیدی ReLU که در اینجا مرتبط هستند عبارتند از:

- خطی در ناحیه مثبت: وقتی x >= 0 باشد، ReLU به عنوان یک تابع identity عمل می کند.
- صفر در ناحیه منفی: وقتی x < 0، خروجی ReLU صفر(constatnt) است.

Activation Patterns و ابرصفحه:

هر نورون در شبکه بر اساس حالت فعال‌سازی خود، یک ابر صفحه را تعریف می‌کند:

- نورون فعال: ورودی پیش از فعال سازی نورون (z) غیر منفی است (z >= 0).
- نورون غیر فعال: ورودی قبل از فعال سازی نورون منفی است (z <0).

این شرایط را می توان به صورت نابرابری های خطی نوشت. برای یک نورون i در لایه اول، ورودی پیش از فعال سازی به صورت زیر است:

$$z_i = w_i^T x + b_i$$

- w_i بردار وزن نورون i است.
- x بردار ورودی است.
- b_i بایاس است.

ابرصفحه تعریف شده با z_i = 0 فضای ورودی را به دو نیم فضا تقسیم می کند:

- نیم فضای مثبت: w_i^T x + b_i >= 0 (نورون فعال است).
- نیم فاصله منفی: w_i^T x + b_i < 0 (نورون غیرفعال است).

حال به اثبات می پردازیم:
برای اثبات اینکه F_A($\mathbb{R}^m$;W) فضای ورودی را به چند polytopes محدب(convex) تقسیم می کند که در آن F_A($\mathbb{R}$;W) در هر ناحیه خطی است:

۱. ابتدا بیایید ببینیم که چه چیزی یک activation pattern را تعیین می کند:

برای هر نورون ReLU، فعال سازی با مثبت یا منفی بودن ورودی آن تعیین می شود.

ورودی هر نورون تابعی خطی از ورودی های آن است: wx + b

بنابراین، هر نورون به طور موثر یک ابر صفحه wx + b = 0 ایجاد می کند که فضا را تقسیم می کند.

۲. خواص کلیدی:

هر ابر صفحه فضا را به دو ناحیه (مثبت و منفی) تقسیم می کند.

تابع ReLU خطی است (identity) وقتی input > 0 باشد و
constant (صفر) است وقتی input ≤ 0 باشد.

۳. اثبات تحدب(convexity):

بیایید دو نقطه x1 و x2 را در یک ناحیه مشترک activation pattern در نظر بگیریم.

برای هر نورون در شبکه، اگر برای x1 و x2 فعال باشد، پس:

$wx1 + b > 0$ و $wx2 + b > 0$

برای هر نقطه $x_t = tx1 + (1-t)x2$ که در آن $t \in [0,1]$:

$$wx_t + b = w(tx1 + (1-t)x2) + b$$

$$wx_t + b = t(wx1 + b) + (1-t)(wx2 + b)$$

$$wx1 + b > 0 \text{ و } wx2 + b > 0 \quad , \quad t > 0 \quad , \quad 1-t > 0 \quad \rightarrow \quad t(wx1 + b) + (1-t)(wx2 + b) > 0$$

$$wx_t + b > 0$$

این نشان می دهد که x_t همان activation pattern مشترک را حفظ می کند.

بنابراین، هر ناحیه محدب است.

۴. اثبات خطی بودن در نواحی مختلف:

در یک activation pattern ثابت:

- نورون های فعال از ورودی های خود به صورت خطی عبور می کنند (identity function) که رفتاری خطی است.
- نورون های غیرفعال خروجی صفر دارند (constant = 0) که رفتاری خطی است.

که این ترکیبی از توابع خطی ایجاد می کند.

بنابراین، F_A(x;W) در هر ناحیه، خطی است.

ب)

برای اثبات حد بالای $O(k^{mn})$ برای $A(F_{A(n,k)}(\mathbb{R}^m; W)$:

بیایید لایه به لایه آنالیز کنیم:

- لایه اول: ورودی را از $\mathbb{R}^m$ می گیرد.
- هر لایه بعدی از k نورون ورودی می گیرد.

۱. تحلیل لایه اول:
   - ابعاد ورودی: m
   - تعداد نورون ها: k
   - هر نورون یک ابر صفحه را در $\mathbb{R}$^m تعریف می کند.
   - با استفاده از حد داده شده در راهنمایی: $r(k,m) \leq \sum_{i=0}^{m} \binom{k}{i} \leq \sum_{i=0}^{m} k^i \leq O(k^m)$
۲. برای لایه های بعدی (i = 2,...,n):
   - هر نورون یک تابع خطی و به دنبال آن ReLU را محاسبه می کند
   - بعد ورودی لایه i ، k است (عرض لایه قبلی)
   - تعداد نورون های لایه i ، k است
   - هر نورون می تواند حداکثر ۲ حالت فعال سازی مختلف ایجاد کند (۰ یا ۱)

۳. بینش کلیدی برای حد:
- برای هر لایه (i (i ≥ 2:
  - هر نورون می تواند حداکثر ۲ حالت داشته باشد
  - با k نورون در هر لایه، حداکثر 2^k الگوی فعال سازی ممکن در هر لایه داریم

۴. ترکیب حد ها:
- لایه اول: تعداد pattern ها O(k^m).
- هر لایه بعدی: تعداد pattern ها O(2^k).
- Total number of patterns ≤ O(k^m) * O(2^k)^{n-1}

۵. از آنجایی که k نورون در هر لایه دلالت میکند 2^k ≤ k^k:

$$O(k^m) * O\left(2^k\right)^{n-1} \leq O(k^m) * O\left(k^k\right)^{n-1} = O\left(k^{m+k(n-1)}\right)$$

برای هر عدد ثابت k ≤ 1، داریم:

$$k^{m+k(n-1)} \leq k^{mn}$$

این به این دلیل است که برای شبکه های عصبی، معمولاً m ≤ k (عرض کوچک تر از اندازه ابعاد ورودی است)، بنابراین:

$$Total\ number\ of\ regions = A(F_{A(n,k)}(\mathbb{R}^{\mathrm{m}}; W) \leq O(k^m) * O\left(k^k\right)^{(n-1)} = O\left(k^{m+k(n-1)}\right) = O(k^{mn})$$

کران نهایی $O(k^{mn})$ به این دلیل است که:

- تعداد نواحی نمی تواند از حاصل ضرب حداکثر مناطق ممکن در هر لایه بیشتر شود.
- نواحی هر لایه با مجموع دو جمله ای ارائه شده در راهنمایی محدود می شوند.
- ترکیب این مرزها در سراسر n لایه این نتیجه را به ما می دهد.

پایان