به نام خدا

سیستم‌های هوشمند (دکتر رشاد حسینی)

تمرین سری **چهارم**

نیم‌سال اول ۱۴۰۲-۱۴۰۱

عرفـان پـنـــاهی ۸۱۰۱۹۸۳۶۹

این فایل شامل گزارش و نتایج شبیه سازی های انجام شده است.

*** **فایل شبیه سازی با پایتون مربوط به هر قسمت این تمرین با عنوان** HW4_Q#x_810198369.ipynb **پیوست شده است.**

سوال ۱: شبکه عصبی پرسپترون با چند لایه مخفی (لینک گزارش)

سوال ۲: کاربرد شبکه های عصبی در طبقه بندی (لینک گزارش)

سوال ۳: یادگیری انتقال یافته برای شبکه EfficientNet (لینک گزارش)

سوال ۱: شبکه عصبی پرسپترون با چند لایه مخفی

*** حل دستی:

در این قسمت ابتدای کار مقادیر a و b را مشخص می‌کنیم و سپس با استفاده از آن بخش feed forward را پیش می‌بریم.

$$Student-number = 810198369 \rightarrow \boldsymbol{a = 9}\,, \qquad \boldsymbol{b = 6}$$

پیش از آغاز حل، برای پیشروی راحت تر در مسئله وزن های مسئله را transpose می‌کنیم و به صورت زیر تعریف می‌کنیم.

$$\overline{W_1} = W_1^T = \begin{bmatrix} 0.9 & 2.7 & 4.5 \\ 1.2 & 2.4 & 2.4 \end{bmatrix}$$

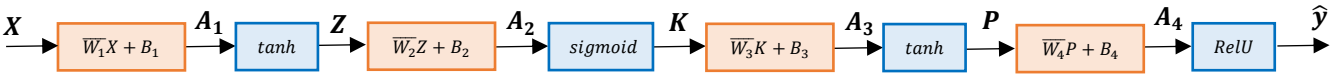
$$\overline{W_2} = W_2^T = \begin{bmatrix} 9.15 & 6.45 \\ 9.25 & 6.55 \\ 9.35 & 6.65 \end{bmatrix}$$

$$\overline{W_3} = W_1^T = \begin{bmatrix} 54.12 & 54.32 & 64.52 \\ 54.22 & 54.42 & 54.62 \end{bmatrix}$$

$$\overline{W_4} = W_4^T = [3.16 \quad 3.36]$$

$$B_1 = \begin{bmatrix} 0.91 \\ 0.62 \end{bmatrix}, \qquad B_2 = \begin{bmatrix} 15.15 \\ 15.25 \\ 15.35 \end{bmatrix}, \qquad B_3 = \begin{bmatrix} \frac{9}{7} + 0.12 \\ \frac{9}{7} + 0.22 \end{bmatrix}, \qquad B_4 = [-2.74]$$

مطابق با تصویر ۱-۱، برای درک راحت تر و نام گذاری بهتر شکل بلوکی برای شبکه رسم می‌کنیم.



تصویر ۱-۱: شکل بلوکی شبکه و نام گذاری هر بخش

ابتدا ورودی X_1 را وارد شبکه می‌کنیم و خروجی شبکه و خطای آن را بدست می‌آوریم.

$$X = X_1 = \begin{bmatrix} 9 \\ 6 \\ 9 \end{bmatrix}, \qquad y = y_1 = [9]$$

$$A_1 = \overline{W_1}X + B_1 = \begin{bmatrix} 0.9 & 2.7 & 4.5 \\ 1.2 & 2.4 & 2.4 \end{bmatrix} \begin{bmatrix} 9 \\ 6 \\ 9 \end{bmatrix} + \begin{bmatrix} 0.91 \\ 0.62 \end{bmatrix} = \begin{bmatrix} 65.71 \\ 47.42 \end{bmatrix}$$

$$\rightarrow Z = \tanh\left(\begin{bmatrix}65.71\\47.42\end{bmatrix}\right) = \begin{bmatrix}1\\1\end{bmatrix}$$

$$A_2 = \overline{W_2}Z + B_2 = \begin{bmatrix}9.15 & 6.45\\9.25 & 6.55\\9.35 & 6.65\end{bmatrix}\begin{bmatrix}1\\1\end{bmatrix} + \begin{bmatrix}15.15\\15.25\\15.35\end{bmatrix} = \begin{bmatrix}30.75\\31.05\\31.35\end{bmatrix}$$

$$\rightarrow K = \text{sigmoid}\left(\begin{bmatrix}30.75\\31.05\\31.35\end{bmatrix}\right) = \begin{bmatrix}1\\1\\1\end{bmatrix}$$

$$A_3 = \overline{W_3}P + B_3 = \begin{bmatrix}54.12 & 54.32 & 64.52\\54.22 & 54.42 & 54.62\end{bmatrix}\begin{bmatrix}1\\1\\1\end{bmatrix} + \begin{bmatrix}1.41\\1.51\end{bmatrix} = \begin{bmatrix}164.37\\164.77\end{bmatrix}$$

$$\rightarrow P = \tanh\left(\begin{bmatrix}164.37\\164.77\end{bmatrix}\right) = \begin{bmatrix}1\\1\end{bmatrix}$$

$$A_4 = \overline{W_4}Z + B_4 = [3.16 \quad 3.36]\begin{bmatrix}1\\1\end{bmatrix} + [-2.74] = [3.78]$$

$$\rightarrow \hat{y} = \text{RelU}(3.78) = [3.78]$$

$$\boldsymbol{E = \frac{1}{2}(\hat{y} - y)^2 = \frac{1}{2}(-5.22)^2 = 13.6242}$$

حال وارد مرحله Back Propagation می‌شویم.

در ابتدا مشتقات مهم را می‌نویسیم و در طول مسئله از آن استفاده می‌کنیم.

$$\frac{\partial E}{\partial \hat{y}} = \hat{y} - y\,, \qquad \frac{\partial \hat{y}}{\partial A_4} = u(A_4), \qquad \frac{\partial A_4}{\partial \overline{W_4}} = \frac{\partial(\overline{W_4}P)}{\partial \overline{W_4}} = P^T, \qquad \frac{\partial A_4}{\partial P} = \frac{\partial(\overline{W_4}P)}{\partial P} = \overline{W_4}^T$$

$$\frac{\partial P}{\partial A_3} = 1 - P^2 \;\; (bitwise), \qquad \frac{\partial A_3}{\partial K} = \frac{\partial(\overline{W_3}K)}{\partial K} = \overline{W_3}^T, \qquad \frac{\partial K}{\partial A_2} = K(1-K) \;\; (bitwise)$$

$$\frac{\partial A_2}{\partial Z} = \frac{\partial(\overline{W_2}Z)}{\partial Z} = \overline{W_2}^T\,, \qquad \frac{\partial Z}{\partial A_1} = 1 - Z^2 \;\; (bitwise)$$

نکته قابل توجه در محاسبه مشتق های برداری، استفاده از ضرب *bitwise* هنگام مستق گیری از *activation function* است:

$$\frac{\partial E}{\partial \overline{W_l}} = \frac{\partial E}{\partial A_l}(a_{l-1})^T\,, \qquad \frac{\partial E}{\partial A_l} = \frac{\partial E}{\partial a_l} \odot \frac{\partial a_l}{\partial A_l}\,, \qquad \frac{\partial E}{\partial B_l} = \frac{\partial E}{\partial A_l}\,, \qquad \frac{\partial E}{\partial a_{l-1}} = (\overline{W_l})^T \frac{\partial E}{\partial A_l}$$

$$a_{1:4} = \{Z, K, P, \hat{y}\}$$

برای سادگی ادامه محاسبات برخی روابط را به صورت زیر ترکیب می‌کنیم.

$$\frac{\partial E}{\partial P} = (\overline{W_4})^T \frac{\partial E}{\partial A_4} = (\overline{W_4})^T \left[\frac{\partial E}{\partial \hat{y}} \odot \frac{\partial \hat{y}}{\partial A_4}\right]$$

$$\frac{\partial E}{\partial K} = (\overline{W_3})^T \frac{\partial E}{\partial A_3} = (\overline{W_3})^T \left[\frac{\partial E}{\partial P} \odot \frac{\partial P}{\partial A_3}\right]$$

$$\frac{\partial E}{\partial Z} = (\overline{W_2})^T \frac{\partial E}{\partial A_2} = (\overline{W_2})^T \left[\frac{\partial E}{\partial K} \odot \frac{\partial K}{\partial A_2}\right]$$

$$\frac{\partial E}{\partial \overline{W_4}} = \frac{\partial E}{\partial A_4}(P)^T = \left(\frac{\partial E}{\partial \hat{y}} \odot \frac{\partial \hat{y}}{\partial A_4}\right) P^T = (\hat{y} - y)u(A_4)P^T$$

$$\rightarrow \boldsymbol{\frac{\partial E}{\partial \overline{W_4}} = (\hat{y} - y)P^T\, u(A_4)}$$

$$\frac{\partial E}{\partial \overline{W_3}} = \frac{\partial E}{\partial A_3}(K)^T = \left[\frac{\partial E}{\partial P} \odot \frac{\partial P}{\partial A_3}\right] K^T = \left[\left((\overline{W_4})^T \left[\frac{\partial E}{\partial \hat{y}} \odot \frac{\partial \hat{y}}{\partial A_4}\right]\right) \odot \frac{\partial P}{\partial A_3}\right] K^T$$

$$= \left[(\hat{y} - y)u(A_4)\overline{W_4}^T \odot (1 - P^2)\right] K^T$$

$$\rightarrow \boldsymbol{\frac{\partial E}{\partial \overline{W_3}} = \left[(\hat{y} - y)\overline{W_4}^T \odot (1 - P^2)\right] K^T\, u(A_4)}$$

$$\frac{\partial E}{\partial \overline{W_2}} = \frac{\partial E}{\partial A_2}(Z^T) = \left[\frac{\partial E}{\partial K} \odot \frac{\partial K}{\partial A_2}\right] Z^T = \left[\left((\overline{W_3})^T \frac{\partial E}{\partial A_3}\right) \odot \frac{\partial K}{\partial A_2}\right] Z^T$$

$$= \left[\left[\overline{W_3}^T \left[\left((\overline{W_4})^T \left[\frac{\partial E}{\partial \hat{y}} \odot \frac{\partial \hat{y}}{\partial A_4}\right]\right) \odot \frac{\partial P}{\partial A_3}\right]\right] \odot \frac{\partial K}{\partial A_2}\right] Z^T$$

$$= \left[\left[\overline{W_3}^T \left[(\hat{y} - y)u(A_4)\overline{W_4}^T \odot (1 - P^2)\right]\right] \odot \big(K(1 - K)\big)\right] Z^T$$

$$\rightarrow \boldsymbol{\frac{\partial E}{\partial \overline{W_2}} = \left[\left[\overline{W_3}^T \left[(\hat{y} - y)u(A_4)\overline{W_4}^T \odot (1 - P^2)\right]\right] \odot \big(K(1 - K)\big)\right] Z^T}$$

$$\frac{\partial E}{\partial \overline{W_1}} = \frac{\partial E}{\partial A_1}(X^T) = \left[\frac{\partial E}{\partial Z} \odot \frac{\partial Z}{\partial A_1}\right] X^T = \left[\left((\overline{W_2})^T \frac{\partial E}{\partial A_2}\right) \odot \frac{\partial Z}{\partial A_1}\right] X^T$$

$$= \left[\left((\overline{W_2})^T \left[\left[\overline{W_3}^T \left[\left((\overline{W_4})^T \left[\frac{\partial E}{\partial \hat{y}} \odot \frac{\partial \hat{y}}{A_4}\right]\right) \odot \frac{\partial P}{\partial A_3}\right]\right] \odot \frac{\partial K}{\partial A_2}\right]\right) \odot \frac{\partial Z}{\partial A_1}\right] X^T$$

$$= \left[\left((\overline{W_2})^T \left[\left[\overline{W_3}^T \left[(\hat{y} - y)u(A_4)\overline{W_4}^T \odot (1 - P^2)\right]\right] \odot \big(K(1 - K)\big)\right]\right)\right.$$

$$\left. \odot (1 - Z^2)\right] X^T$$

$$\rightarrow \boldsymbol{\frac{\partial E}{\partial \overline{W_1}} = \left[\left((\overline{W_2})^T \left[\left[\overline{W_3}^T \left[(\hat{y} - y)u(A_4)\overline{W_4}^T \odot (1 - P^2)\right]\right] \odot \big(K(1 - K)\big)\right]\right) \odot (1 - Z^2)\right] X^T}$$

$$\frac{\partial E}{\partial B_4} = \frac{\partial E}{\partial A_4} = \frac{\partial E}{\partial \hat{y}} \odot \frac{\partial \hat{y}}{\partial A_4} = (\hat{y} - y)u(A_4)$$

$$\rightarrow \boldsymbol{\frac{\partial E}{\partial B_4} = (\hat{y} - y)u(A_4)}$$

$$\frac{\partial E}{\partial B_3} = \frac{\partial E}{\partial A_3} = \frac{\partial E}{\partial P} \odot \frac{\partial P}{\partial A_3} = \left((\overline{W_4})^T \left[\frac{\partial E}{\partial \hat{y}} \odot \frac{\partial \hat{y}}{\partial A_4}\right]\right) \odot \frac{\partial P}{\partial A_3} = (\hat{y} - y)u(A_4)\left[\overline{W_4}^T \odot (1 - P^2)\right]$$

$$\rightarrow \boldsymbol{\frac{\partial E}{\partial B_3} = (\hat{y} - y)u(A_4)\left[\overline{W_4}^T \odot (1 - P^2)\right]}$$

$$\frac{\partial E}{\partial B_2} = \frac{\partial E}{\partial A_2} = \frac{\partial E}{\partial K} \odot \frac{\partial K}{\partial A_2} = \left[\overline{W_3}^T \left[\left((\overline{W_4})^T \left[\frac{\partial E}{\partial \hat{y}} \odot \frac{\partial \hat{y}}{\partial A_4}\right]\right) \odot \frac{\partial P}{\partial A_3}\right]\right] \odot \frac{\partial K}{\partial A_2}$$

$$= \left[\overline{W_3}^T \left[(\hat{y} - y)u(A_4)\overline{W_4}^T \odot (1 - P^2)\right]\right] \odot \left(K(1-K)\right)$$

$$\rightarrow \boldsymbol{\frac{\partial E}{\partial B_2} = \left[\overline{W_3}^T \left[(\hat{y} - y)u(A_4)\overline{W_4}^T \odot (1 - P^2)\right]\right] \odot \left(K(1-K)\right)}$$

$$\frac{\partial E}{\partial B_1} = \frac{\partial E}{\partial A_1} = \frac{\partial E}{\partial Z} \odot \frac{\partial Z}{\partial A_1} = \left((\overline{W_2})^T \left[\left[\overline{W_3}^T \left[\left((\overline{W_4})^T \left[\frac{\partial E}{\partial \hat{y}} \odot \frac{\partial \hat{y}}{A_4}\right]\right) \odot \frac{\partial P}{\partial A_3}\right]\right] \odot \frac{\partial K}{\partial A_2}\right]\right) \odot \frac{\partial Z}{\partial A_1}$$

$$= \left((\overline{W_2})^T \left[\left[\overline{W_3}^T \left[(\hat{y} - y)u(A_4)\overline{W_4}^T \odot (1 - P^2)\right]\right] \odot \left(K(1-K)\right)\right]\right) \odot (1 - Z^2)$$

$$\rightarrow \boldsymbol{\frac{\partial E}{\partial B_1} = \left((\overline{W_2})^T \left[\left[\overline{W_3}^T \left[(\hat{y} - y)u(A_4)\overline{W_4}^T \odot (1 - P^2)\right]\right] \odot \left(K(1-K)\right)\right]\right) \odot (1 - Z^2)}$$

حال با استفاده از مقادیر بدست آمده در مرحله Feed Forward و هچنین مشتقات بدست آمده، مقادیر وزن ها و بایاس ها را به روز می‌کنیم. طول گام را $\eta = 0.1$ در نظر می‌گیریم.

$$\frac{\partial E}{\partial B_4} = (\hat{y} - y)u(A_4) = (3.78 - 9)u(3.78) = -5.22$$

$$B_4^{new} = B_4^{old} - \eta \frac{\partial E}{\partial B_4} = -2.74 + 0.522 = -2.218$$

$$\boldsymbol{B_4^{new} = -2.218}$$

$$\frac{\partial E}{\partial \overline{W_4}} = (\hat{y} - y)P^T\, u(A_4) = (3.78 - 9)[1 \quad 1]u(3.78) = [-5.22 \quad -5.22]$$

$$\overline{W_4}^{new} = \overline{W_4}^{old} - \eta \frac{\partial E}{\partial \overline{W_4}} = [3.16 \quad 3.36] + [0.522 \quad 0.522] = [3.682 \quad 3.882]$$

$$\frac{\partial E}{\partial B_3} = (\hat{y} - y)u(A_4)\left[\overline{W_4}^T \odot (1 - P^2)\right] = (3.78 - 9)u(3.78)\left[\begin{bmatrix}3.682\\3.882\end{bmatrix} \odot \begin{bmatrix}0\\0\end{bmatrix}\right] = 0$$

$$B_3^{new} = B_3^{old} - \eta \frac{\partial E}{\partial B_3} = B_3^{old} = \begin{bmatrix}1.41\\1.51\end{bmatrix}$$

با توجه به اینکه در تمامی مشتقات وزن ها و بایاس های در مراحل بعدی $(1 - P^2) = \begin{bmatrix}0\\0\end{bmatrix}$ وجود دارد، تمامی مشتق ها در مراحل بعدی صفر خواهد بود و ضرایب و وزن ها به روز نمی‌شوند.

$$\boldsymbol{\overline{W_1}^{new} = \begin{bmatrix}0.9 & 2.7 & 4.5\\1.2 & 2.4 & 2.4\end{bmatrix}}$$

$$\boldsymbol{\overline{W_2}^{new} = \begin{bmatrix}9.15 & 6.45\\9.25 & 6.55\\9.35 & 6.65\end{bmatrix}}$$

$$\overline{\boldsymbol{W_3}}^{\boldsymbol{new}} = \begin{bmatrix} \boldsymbol{54.12} & \boldsymbol{54.32} & \boldsymbol{64.52} \\ \boldsymbol{54.22} & \boldsymbol{54.42} & \boldsymbol{54.62} \end{bmatrix}$$

$$\overline{\boldsymbol{W_4}}^{\boldsymbol{new}} = [\boldsymbol{3.682} \quad \boldsymbol{3.882}]$$

$$\boldsymbol{B_1^{new}} = \begin{bmatrix} \boldsymbol{0.91} \\ \boldsymbol{0.62} \end{bmatrix}, \qquad \boldsymbol{B_2^{new}} = \begin{bmatrix} \boldsymbol{15.15} \\ \boldsymbol{15.25} \\ \boldsymbol{15.35} \end{bmatrix}, \qquad \boldsymbol{B_3^{new}} = \begin{bmatrix} \boldsymbol{1.41} \\ \boldsymbol{1.51} \end{bmatrix}, \qquad \boldsymbol{B_4^{new}} = [\boldsymbol{-2.218}]$$

اکنون ورودی X_2 را وارد شبکه می‌کنیم و ضرایب را برای این ورودی به روز می‌کنیم:

$$X = X_2 = \begin{bmatrix} 6 \\ 9 \\ 6 \end{bmatrix}, \qquad y = y_2 = [6]$$

- **Feed Forward:**

$$A_1 = \overline{W_1}X + B_1 = \begin{bmatrix} 0.9 & 2.7 & 4.5 \\ 1.2 & 2.4 & 2.4 \end{bmatrix} \begin{bmatrix} 6 \\ 9 \\ 6 \end{bmatrix} + \begin{bmatrix} 0.91 \\ 0.62 \end{bmatrix} = \begin{bmatrix} 57.61 \\ 43.82 \end{bmatrix}$$

$$\rightarrow Z = \tanh\left(\begin{bmatrix} 57.61 \\ 43.82 \end{bmatrix}\right) = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

$$A_2 = \overline{W_2}Z + B_2 = \begin{bmatrix} 9.15 & 6.45 \\ 9.25 & 6.55 \\ 9.35 & 6.65 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \end{bmatrix} + \begin{bmatrix} 15.15 \\ 15.25 \\ 15.35 \end{bmatrix} = \begin{bmatrix} 30.75 \\ 31.05 \\ 31.35 \end{bmatrix}$$

$$\rightarrow K = \text{sigmoid}\left(\begin{bmatrix} 30.75 \\ 31.05 \\ 31.35 \end{bmatrix}\right) = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$$

$$A_3 = \overline{W_3}P + B_3 = \begin{bmatrix} 54.12 & 54.32 & 64.52 \\ 54.22 & 54.42 & 54.62 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} + \begin{bmatrix} 1.41 \\ 1.51 \end{bmatrix} = \begin{bmatrix} 164.37 \\ 164.77 \end{bmatrix}$$

$$\rightarrow P = \tanh\left(\begin{bmatrix} 164.37 \\ 164.77 \end{bmatrix}\right) = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

$$A_4 = \overline{W_4}Z + B_4 = [3.682 \quad 3.882] \begin{bmatrix} 1 \\ 1 \end{bmatrix} + [-2.218] = [5.346]$$

$$\rightarrow \hat{y} = \text{RelU}(5.346) = [5.346]$$

- **Back Propagation:**

$$\frac{\partial E}{\partial B_4} = (\hat{y} - y)u(A_4) = (5.346 - 6)u(5.346) = -0.654$$

$$B_4^{new} = B_4^{old} - \eta \frac{\partial E}{\partial B_4} = -2.218 + 0.0654 = -2.1526$$

$$\boldsymbol{B_4^{new}} = \boldsymbol{-2.1526}$$

$$\frac{\partial E}{\partial \overline{W_4}} = (\hat{y} - y)P^T\, u(A_4) = (5.346 - 6)[1 \quad 1]u(5.346) = [-0.654 \quad -0.654]$$

$$\overline{W_4}^{new} = \overline{W_4}^{old} - \eta \frac{\partial E}{\partial \overline{W_4}} = [3.682 \quad 3.882] + [0.0654 \quad 0.0654] = [3.7474 \quad 3.9474]$$

$$\frac{\partial E}{\partial B_3} = (\hat{y} - y)u(A_4)\left[\overline{W_4}^T \odot (1 - P^2)\right] = (5.346 - 6)u(5.346)\left[\begin{bmatrix}3.7474\\3.9474\end{bmatrix} \odot \begin{bmatrix}0\\0\end{bmatrix}\right] = 0$$

$$B_3^{new} = B_3^{old} - \eta \frac{\partial E}{\partial B_3} = B_3^{old} = \begin{bmatrix}1.41\\1.51\end{bmatrix}$$

با توجه به اینکه در تمامی مشتقات وزن ها و بایاس های در مراحل بعدی $(1 - P^2) = \begin{bmatrix}0\\0\end{bmatrix}$ وجود دارد، تمامی مشتق ها در مراحل بعدی صفر خواهد بود و ضرایب و وزن ها به روز نمی‌شوند.

در نتیجه بعد از دو مرحله به روز رسانی ضرایب، در نهایت ضرایب به صورت زیر خواهد بود:

$$\overline{\boldsymbol{W_1}}^{\boldsymbol{new}} = \begin{bmatrix}\mathbf{0.9} & \mathbf{2.7} & \mathbf{4.5}\\ \mathbf{1.2} & \mathbf{2.4} & \mathbf{2.4}\end{bmatrix} \rightarrow \boldsymbol{W_1^{new}} = \begin{bmatrix}\mathbf{0.9} & \mathbf{1.2}\\ \mathbf{2.7} & \mathbf{2.4}\\ \mathbf{4.5} & \mathbf{2.4}\end{bmatrix}$$

$$\overline{\boldsymbol{W_2}}^{\boldsymbol{new}} = \begin{bmatrix}\mathbf{9.15} & \mathbf{6.45}\\ \mathbf{9.25} & \mathbf{6.55}\\ \mathbf{9.35} & \mathbf{6.65}\end{bmatrix} \rightarrow \boldsymbol{W_2^{new}} = \begin{bmatrix}\mathbf{9.15} & \mathbf{9.25} & \mathbf{9.35}\\ \mathbf{6.45} & \mathbf{6.55} & \mathbf{6.65}\end{bmatrix}$$

$$\overline{\boldsymbol{W_3}}^{\boldsymbol{new}} = \begin{bmatrix}\mathbf{54.12} & \mathbf{54.32} & \mathbf{54.52}\\ \mathbf{54.22} & \mathbf{54.42} & \mathbf{54.62}\end{bmatrix} \rightarrow \boldsymbol{W_3^{new}} = \begin{bmatrix}\mathbf{54.12} & \mathbf{54.22}\\ \mathbf{54.32} & \mathbf{54.42}\\ \mathbf{54.52} & \mathbf{54.62}\end{bmatrix}$$

$$\overline{\boldsymbol{W_4}}^{\boldsymbol{new}} = [\mathbf{3.7474} \quad \mathbf{3.9474}] \rightarrow \boldsymbol{W_4^{new}} = \begin{bmatrix}\mathbf{3.7474}\\ \mathbf{3.9474}\end{bmatrix}$$

$$\boldsymbol{B_1^{new}} = \begin{bmatrix}\mathbf{0.91}\\ \mathbf{0.62}\end{bmatrix}, \quad \boldsymbol{B_2^{new}} = \begin{bmatrix}\mathbf{15.15}\\ \mathbf{15.25}\\ \mathbf{15.35}\end{bmatrix}, \quad \boldsymbol{B_3^{new}} = \begin{bmatrix}\mathbf{1.41}\\ \mathbf{1.51}\end{bmatrix}, \quad \boldsymbol{B_4^{new}} = [\mathbf{-2.1526}]$$

***** پیاده سازی در پایتون:**

در این قسمت، با استفاده از ضرایب اولیه و همچنین روابط مشتق ها سعی می‌کنیم تابعی برای شبکه داده شده بنویسیم و سپس در چند epoch ضرایب را بدست آورده و نشان دهیم خطا در حال کاهش است.

برای ساده سازی نوشتن تابع شبکه سعی می‌کنیم دو تابع برای بخش های Feed-Forward و Back-Propagation بنویسیم. روابط مربوط به تابع Feed-Forward واضح است. برای تابع Back-Propagation از روابط بازگشتی مشتق ها (که در قسمت قبل به آن اشاره شد) استفاده می‌کنیم.

$$\frac{\partial E}{\partial \overline{W_l}} = \frac{\partial E}{\partial A_l}(a_{l-1})^T, \quad \frac{\partial E}{\partial A_l} = \frac{\partial E}{\partial a_l} \odot \frac{\partial a_l}{\partial A_l}, \quad \frac{\partial E}{\partial B_l} = \frac{\partial E}{\partial A_l}, \quad \frac{\partial E}{\partial a_{l-1}} = (\overline{W_l})^T \frac{\partial E}{\partial A_l}$$

$$a_{1:4} = \{Z, K, P, \hat{y}\}$$

در نهایت خروجی را به ازای ورودی های داده شده و تکرار های مختلف بررسی می‌کنیم.

مرحله اول: به روز شدن ضرایب با استفاده از ورودی X_1

تصویر ۱-۲، ضرایب به روز شده را به همراه خطای مرحله Feed-Forward را به ازای ورودی X_1 نشان می‌دهد. همانطور که مشاهده می‌شود ضرایب و خطا دقیقاً مطابق با محاسبات دستی می‌باشد.

```
epoch (1):
       step 1 (X1, y1): y_hat = [[3.78]] , Error = [[13.6242]]

B1_new =
[[0.91]
 [0.62]]
W1_new =
[[0.9 1.2]
 [2.7 2.4]
 [4.5 2.4]]
B2_new =
[[15.15]
 [15.25]
 [15.35]]
W2_new =
[[9.15 9.25 9.35]
 [6.45 6.55 6.65]]
B3_new =
[[1.40571429]
 [1.50571429]]
W3_new =
[[54.12 54.22]
 [54.32 54.42]
 [54.52 54.62]]
B4_new =
[[-2.218]]
W4_new =
[[3.682]
 [3.882]]
```

تصویر ۱-۲: ضرایب به روز شده به همراه خطای مرحله Feed-Forward را به ازای ورودی X_1

مرحله دوم: به روز شدن ضرایب با استفاده از ورودی X_2

تصویر ۱-۳، ضرایب به روز شده شبکه جدید را به همراه خطای مرحله Feed-Forward را به ازای ورودی X_2 نشان می‌دهد. همانطور که مشاهده می‌شود ضرایب و خطا دقیقاً مطابق با محاسبات دستی می‌باشد.

```
epoch (1):
       step 1 (X1, y1): y_hat = [[3.78]] , Error = [[13.6242]]
       step 2 (X2, y2): y_hat = [[5.346]] , Error = [[0.213858]]

B1_new =
[[0.91]
 [0.62]]
W1_new =
[[0.9 1.2]
 [2.7 2.4]
 [4.5 2.4]]
B2_new =
[[15.15]
 [15.25]
 [15.35]]
W2_new =
[[9.15 9.25 9.35]
 [6.45 6.55 6.65]]
B3_new =
[[1.40571429]
 [1.50571429]]
W3_new =
[[54.12 54.22]
 [54.32 54.42]
 [54.52 54.62]]
B4_new =
[[-2.1526]]
W4_new =
[[3.7474]
 [3.9474]]
```

تصویر۱-۳: ضرایب به روز شده شبکه جدید به همراه خطای مرحله Feed-Forward را به ازای ورودی X_2

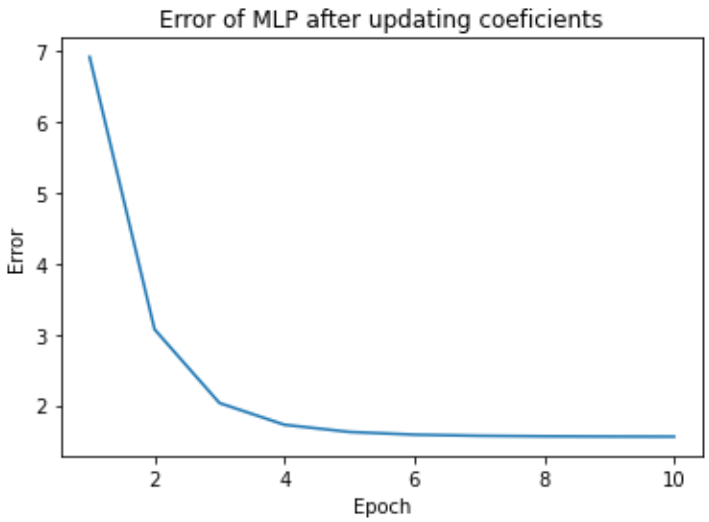
حال می‌خواهیم به ازای هر دو ورودی تا ۱۰ دوره شبکه را به روز کنیم و تا حد امکان خطا را کمینه کنیم. تصویر ۱-۴ خطا و خروجی های بدست آمده به ازای هر ورودی را به همراه ضرایب به روز شده نشان می‌دهد.

```
B1_new =
[[0.91]
 [0.62]]
W1_new =
[[0.9 1.2]
 [2.7 2.4]
 [4.5 2.4]]
B2_new =
[[15.15]
 [15.25]
 [15.35]]
W2_new =
[[9.15 9.25 9.35]
 [6.45 6.55 6.65]]
B3_new =
[[1.40571429]
 [1.50571429]]
W3_new =
[[54.12 54.22]
 [54.32 54.42]
 [54.52 54.62]]
B4_new =
[[-1.58915431]]
W4_new =
[[4.31084569]
 [4.51084569]]
```

```
epoch (1):
        step 1 (X1, y1): y_hat = [[3.78]] , Error = [[13.6242]]
        step 2 (X2, y2): y_hat = [[5.346]] , Error = [[0.213858]]
epoch (2):
        step 1 (X1, y1): y_hat = [[5.5422]] , Error = [[5.97819042]]
        step 2 (X2, y2): y_hat = [[6.57954]] , Error = [[0.16793331]]
epoch (3):
        step 1 (X1, y1): y_hat = [[6.405678]] , Error = [[3.36525332]]
        step 2 (X2, y2): y_hat = [[7.1839746]] , Error = [[0.70089793]]
epoch (4):
        step 1 (X1, y1): y_hat = [[6.82878222]] , Error = [[2.35709332]]
        step 2 (X2, y2): y_hat = [[7.48014755]] , Error = [[1.09541839]]
epoch (5):
        step 1 (X1, y1): y_hat = [[7.03610329]] , Error = [[1.92844515]]
        step 2 (X2, y2): y_hat = [[7.6252723]] , Error = [[1.32075503]]
epoch (6):
        step 1 (X1, y1): y_hat = [[7.13769061]] , Error = [[1.73409813]]
        step 2 (X2, y2): y_hat = [[7.69638343]] , Error = [[1.43885837]]
epoch (7):
        step 1 (X1, y1): y_hat = [[7.1874684]] , Error = [[1.6426354]]
        step 2 (X2, y2): y_hat = [[7.73122788]] , Error = [[1.49857499]]
epoch (8):
        step 1 (X1, y1): y_hat = [[7.21185952]] , Error = [[1.5987232]]
        step 2 (X2, y2): y_hat = [[7.74830166]] , Error = [[1.52827935]]
epoch (9):
        step 1 (X1, y1): y_hat = [[7.22381116]] , Error = [[1.57742339]]
        step 2 (X2, y2): y_hat = [[7.75666781]] , Error = [[1.5429409]]
epoch (10):
        step 1 (X1, y1): y_hat = [[7.22966747]] , Error = [[1.56703863]]
        step 2 (X2, y2): y_hat = [[7.76076723]] , Error = [[1.55015062]]
```

تصویر ۱-۴: ضرایب به روز شده شبکه جدید به همراه خطای مرحله Feed-Forward را به ازای هر دو ورودی پس از ۱۰ دوره

همچنین تصویر ۱-۵، نمودار خطا به ازای هر دوره را نشان می‌دهد. همانطور که در تصویر مشاهده می‌شود، خطا پس از چند دوره کمینه شده و همگرا می‌شود.



تصویر ۱-۵: نمودار خطا برحسب دوره

سوال ۲: کاربرد شبکه های عصبی در طبقه بندی

ابتدا با استفاده از دستورات صورت گزارش به‌شکل زیر دادگان را وارد می‌کنیم.

```
from keras.datasets import cifar10
(x_train, y_train), (x_test, y_test) = cifar10.load_data()
```

سپس داده های آموزش را به دو بخش train و validation تقسیم می‌کنیم. به این نکته دقت می‌کنیم که با توجه به اینکه سایز batch ها از ۳۲، ۶۴ یا ۲۵۶ خواهد بود، داده های train مضربی از این سه طول باشد. برای این کار از ۵۰۰۰۰ داده آموزشی، تقریباً ۹۰٪ آن معادل با ۴۵۰۵۶ داده را به train و مابقی را به validation اختصاص می‌دهیم.

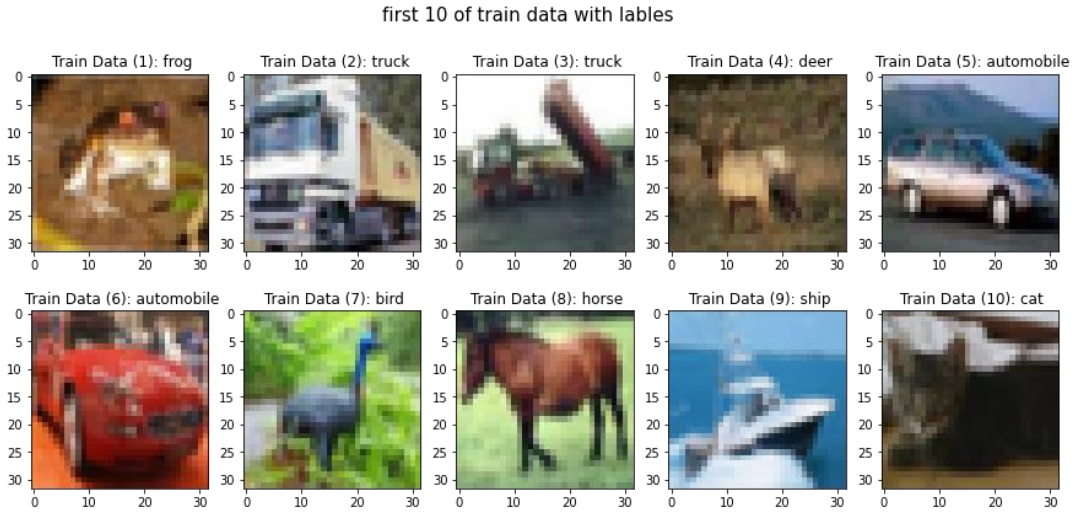
*** البته می‌توانیم این کار را نکنیم و دقیقاً ۹۰٪ داده ها را به train اختصاص دهیم اما به دو دلیل سعی می‌کنیم تخصیص به شکل بالا باشد:

۱- آموزش روی تمامی داده ها باشد.

۲- در صورتی که تعداد داده های train مضربی از batch ها نباشد، یک سری داده ها کلاً برای آموزش استفاده نمی‌شوند و شاید وجود این داده ها در دقت ماشین مؤثر باشد.

همچنین با استفاده از دستور to_categorical() کتابخانه keras نیز Label ها را به شکل one-hot در می‌آوریم.

در نهایت مطابق با تصویر ۲-۱، ۱۰ تصویر اول مجموعه داده های train را نمایش می‌دهیم.



تصویر ۲-۱: ۱۰ تصویر اول مجموعه داده های train

حال می‌خواهیم با استفاده از کتابخانه keras شبکه ای طراحی کنیم که بتوانیم به کمک آن داده ها را طبقه بندی کنیم. در قسمت اول یک شبکه با دو لایه مخفی طراحی می‌کنیم و در قسمت دوم، به شبکه طراحی شده لایه های کانولوشنی اضافه می‌کنیم و عملکرد ان را بهبود می‌دهیم.

❋❋❋ قسمت الف: استفاده از شبکه MLP

در این قسمت ابتدا یک شبکه با دو لایه مخفی طراحی می‌کنیم. با آزمون و خطا نتیجه می‌گیریم که طول لایه های مخفی را ۲۰۰ و ۱۰۰ در نظر بگیریم (پارامتر های مؤثر در انتخاب این طول ها دقت و سرعت یادگیری است). برای پیاده سازی شبکه یک تابع به اسم MLP() نوشته شده است که پارامتر های متغیر در مسئله را به‌عنوان ورودی به آن می‌دهیم.

سوال ۱. انتخاب مناسب ترین Batch-size

در این سوال، پارامتر های دیگر را به صورت زیر در نظر می‌گیریم.

Batch-Size: {32, 64, 256}

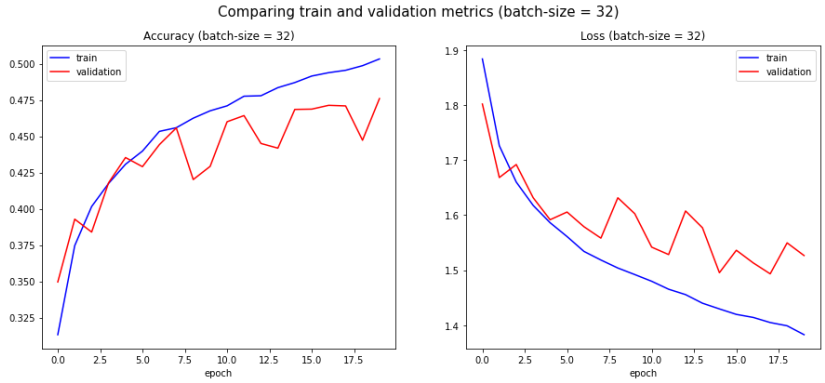
Activation Functions: {Layer #1: ‘RelU’, Layer #2: ‘RelU’}

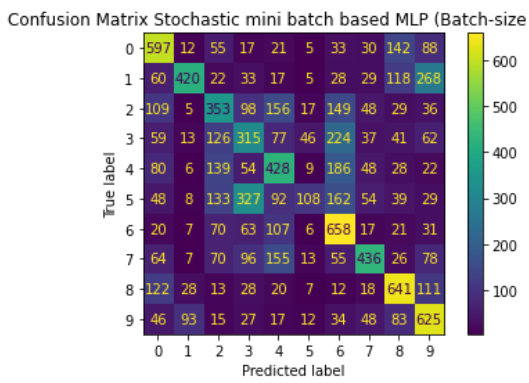
Optimizer: SGD (learning-rate = 0.01, momentum = 0.9)

Loss Function: Categorical Cross Entropy

(تعداد دوره ها (Epoch) در این قسمت را ۲۰ در نظر می‌گیریم.)

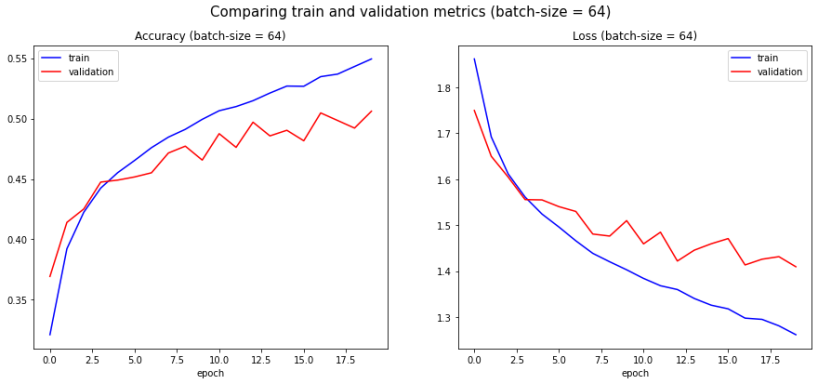
تصاویر ۲-۲ خروجی های خواسته شده برای این سوال را نشان می‌دهد. همانطور که مشاهده می‌شود، برای طول Batch برابر ۲۵۶ دقت و سرعت یادگیری ماشین بهتر است.

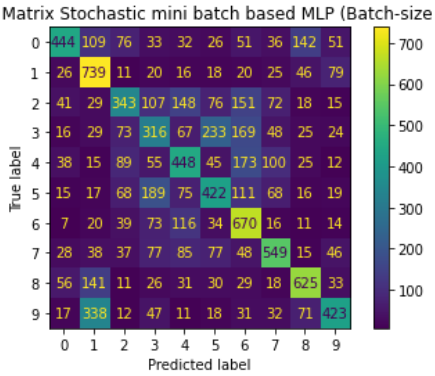





```
Stochastic mini batch based MLP (Batch-size = 32) on test data:
 Accuracy = 45.809999108314514%
 Test-loss = 1.5368760824203491
 Precision = 47.237927305144225%
 Recall = 45.81%
 f-score = 44.47320570608554%
```

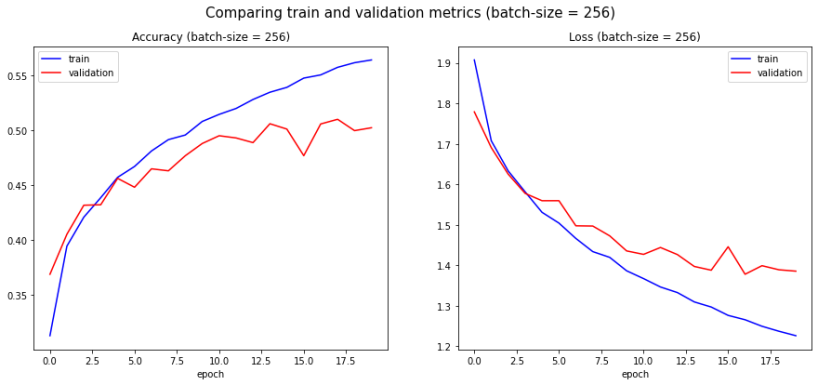
الف) batch-size = 32

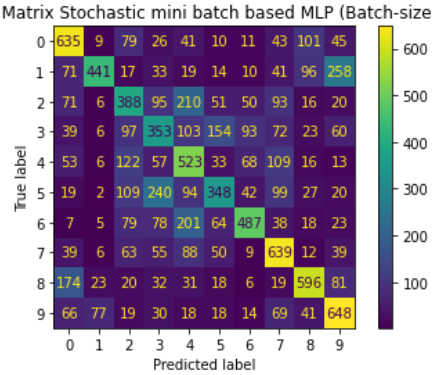





```
Stochastic mini batch based MLP (Batch-size = 64) on test data:
 Accuracy = 49.79000091552734%
 Test-loss = 1.4227638244628906
 Precision = 50.49974948765814%
 Recall = 49.79%
 f-score = 49.31798093506379%
```

ب) batch-size = 64






```
Stochastic mini batch based MLP (Batch-size = 256) on test data:
 Accuracy = 50.58000087738037%
 Test-loss = 1.398481845855713
 Precision = 52.019889055489145%
 Recall = 50.580000000000005%
 f-score = 50.4841062246878%
```

ج) batch-size = 256

تصویر ۲-۲: دقت و سایر پارامتر های خواسته شده برای *batch-size* های مختلف شبکه

سوال ۲. انتخاب مناسب ترین Activation function ها برای لایه های مخفی

در این سوال، پارامتر های دیگر را به صورت زیر در نظر می‌گیریم.

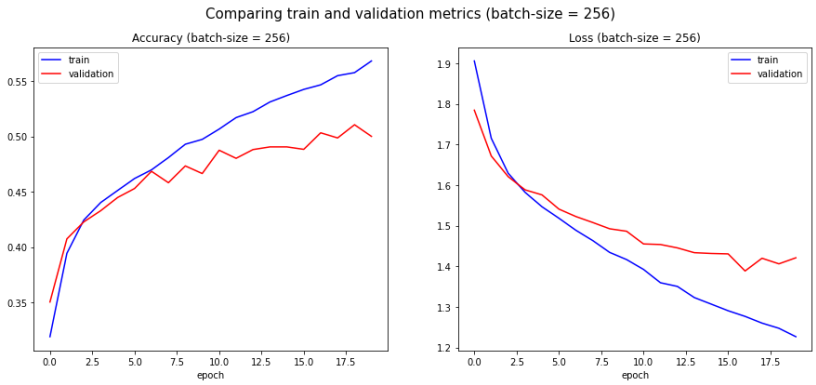
Batch-Size = 256

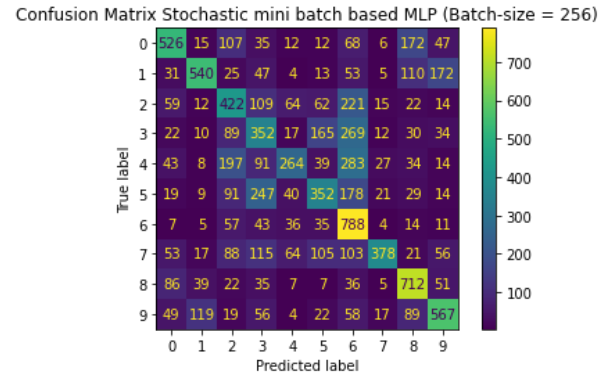
Activation Functions: {Layer #1: ‘RelU’, Layer #2: ‘RelU’}
{Layer #1: ‘tanh’, Layer #2: ‘tanh’}
{Layer #1: ‘RelU’, Layer #2: ‘sigmoid’}

Optimizer: SGD (learning-rate = 0.01, momentum = 0.9)

Loss Function: Categorical Cross Entropy

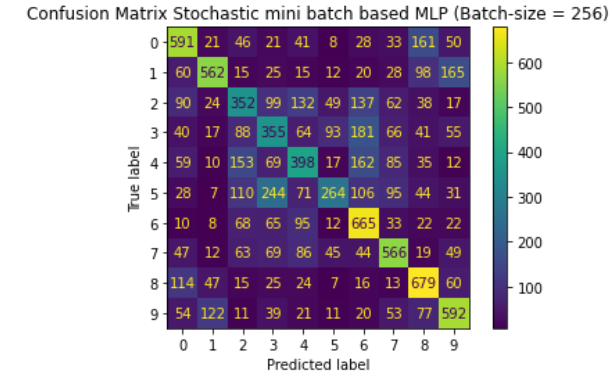
تصاویر ۲-۳ خروجی های خواسته شده برای این سوال را نشان می‌دهد. همچنین حالت اول (RelU, RelU) در سوال ۱ بررسی شد. همانطور که مشاهده می‌شود، برای حالت اول (RelU, RelU) عملکرد یادگیری ماشین بهتر است.






```
Stochastic mini batch based MLP (Batch-size = 256) on test data:
 activation functions : (tanh, tanh)
 Accuracy = 49.00999963283539%
 Test-loss = 1.4366286993026733
 Precision = 52.34344074130307%
 Recall = 49.01%
 f-score = 48.64214984179133%
```

activation functions = {tanh, tanh} الف)




```
Stochastic mini batch based MLP (Batch-size = 256) on test data:
 activation functions : (relu, sigmoid)
 Accuracy = 50.23999810218811%
 Test-loss = 1.4018125534057617
 Precision = 50.32213467956454%
 Recall = 50.239999999999995%
 f-score = 49.608251572407625%
```

activation functions = {RelU, sigmoid} ب)

تصویر ۲-۳: دقت و سایر پارامتر های خواسته شده برای *activation function* های مختلف شبکه

سوال ۳. انتخاب مناسب ترین تابع خطا

در این سوال، پارامتر های دیگر را به صورت زیر در نظر می‌گیریم.

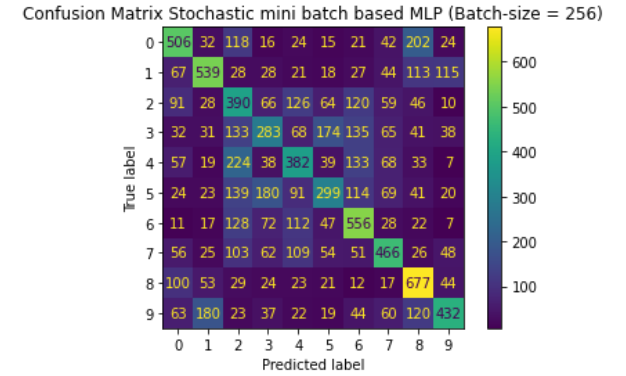
Batch-Size = 256

Activation Functions: {Layer #1: ‘RelU’, Layer #2: ‘RelU’}

Optimizer: SGD (learning-rate = 0.01, momentum = 0.9)

Loss Function: {Categorical Cross Entropy, Poisson}

تصاویر ۲-۴ خروجی های خواسته شده برای این سوال را نشان می‌دهد. همچنین تابع اول Categorical Cross Entropy در سوال ۱ بررسی شد. همانطور که مشاهده می‌شود، برای تابع خطا Categorical Cross Entropy عملکرد یادگیری ماشین بهتر است.




```
Stochastic mini batch based MLP (Batch-size = 256) on test data:
 activation functions : (relu, relu)
 loss : Poisson
 Accuracy = 45.3000009059906%
 Test-loss = 0.25522226095199585
 Precision = 45.669519591673144%
 Recall = 45.300000000000004%
 f-score = 45.031096665157555%
```

تصویر ۲-۴: دقت و سایر پارامتر های خواسته شده برای *loss function* های مختلف شبکه

سوال ۴. انتخاب مناسب ترین بهینه ساز

در این سوال، پارامتر های دیگر را به صورت زیر در نظر می‌گیریم.

Batch-Size = 256

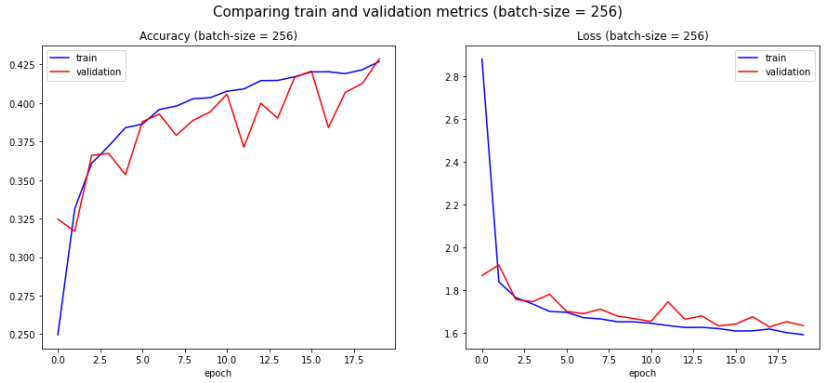
Activation Functions: {Layer #1: ‘RelU’, Layer #2: ‘RelU’}

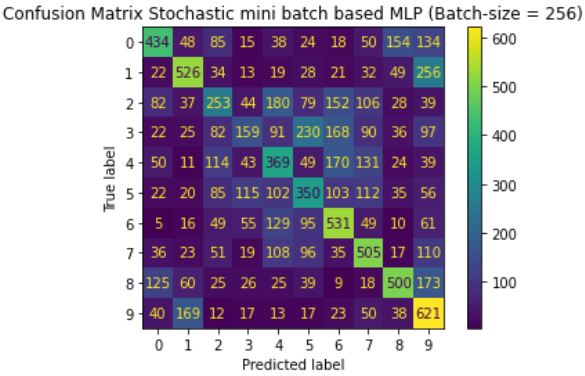
Optimizer: SGD (learning-rate = 0.01, momentum = 0.9)

Adam (learning-rate = 0.01)

Loss Function: {Categorical Cross Entropy, Poisson}

تصاویر ۲-۵ خروجی های خواسته شده برای این سوال را نشان می‌دهد. همچنین بهینه ساز اول SGD در سوال ۱ بررسی شد. همانطور که مشاهده می‌شود، برای بهینه ساز SGD عملکرد یادگیری ماشین بهتر است.






```
Stochastic mini batch based MLP (Batch-size = 256) on test data:
 activation functions : (tanh, tanh)
 loss : categorical_crossentropy
 optimizer : Adam
 Accuracy = 42.48000085353851%
 Test-loss = 1.6113728284835815
 Precision = 42.323368589184 84%
 Recall = 42.480000000000004%
 f-score = 41.710063002794726%
```

تصویر ۲-۵: دقت و سایر پارامتر های خواسته شده برای *loss function* های مختلف شبکه

سوال ۵. انتخاب مناسب ترین پارامتر های شبکه

با توجه به نتایج بدست آمده بهترین پارامتر ها که منجر به دقت و سرعت بهتری در عملکرد یادگیری شبکه دارند به صورت زیر خواهد بود.

Batch-Size = 256

Activation Functions: {Layer #1: ‘RelU’, Layer #2: ‘RelU’}

Optimizer: SGD (learning-rate = 0.01, momentum = 0.9)

Loss Function: {Categorical Cross Entropy, Poisson}

تصویر ۲-۶ خلاصه لایه های شبکه با پارامتر های بالا را نشان می‌دهد.

```
Layer (type)                 Output Shape              Param #
=================================================================
flatten_2 (Flatten)          (None, 3072)              0

dense_6 (Dense)              (None, 200)               614600

dense_7 (Dense)              (None, 100)               20100

dense_8 (Dense)              (None, 10)                1010

=================================================================
Total params: 635,710
Trainable params: 635,710
Non-trainable params: 0
```

تصویر ۲-۶: خلاصه لایه های شبکه طراحی شده با بهترین پارامتر ها

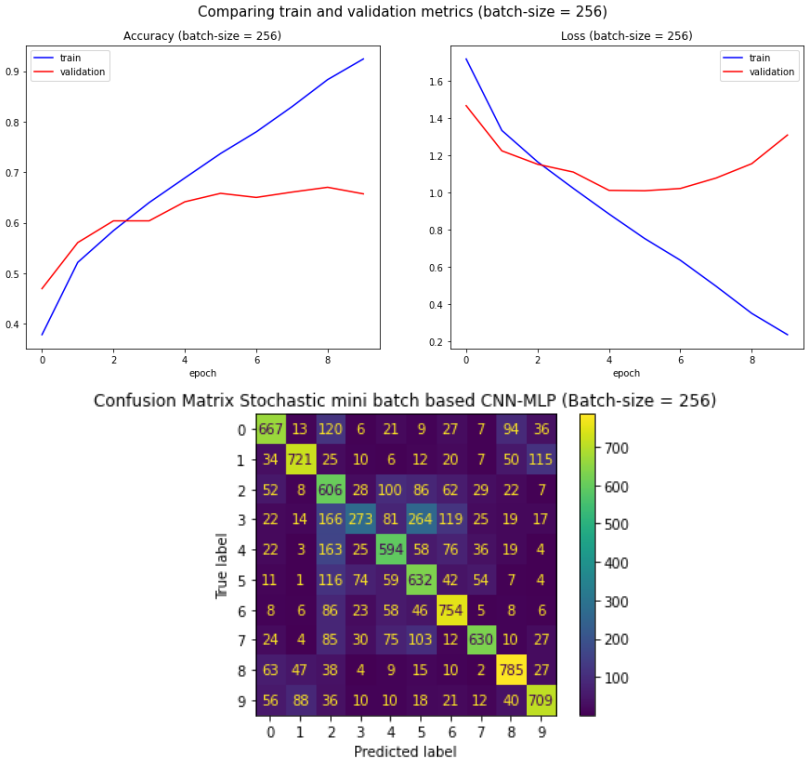
❊❊❊ قسمت ب: استفاده از شبکه MLP+CNN

سوال ۱. تأثیر اضافه شدن لایه های کانولوشنی

حال می‌خواهیم به بهترین شبکه کانولوشنی طراحی شده در قسمت قبل که در سوال ۵ قسمت الف نشان داده شده است، دو لایه کانولوشنی اضافه کنیم و تأثیر آن‌را روی دقت شبکه بررسی کنیم. با توجه به اینکه در این قسمت شبکه خیلی کند می‌شود تعداد epoch ها را ۱۰ در نظر می‌گیریم. (تابع نوشته شده برای این قسمت به صورت CNN() می‌باشد.)

تصویر ۲-۷ نمودار خطا و دقت به همراه پارامتر های مقایسه را به همراه خلاصه لایه های شبکه نشان می‌دهد. همانطور که مشاهده می‌شود دقت بسیار افزایش پیدا کرده و خطا نیز کم شده است اما نکته قابل توجه اینجاست که شبکه بسیار کند عمل می‌کند. در سوالات بعد می‌خواهیم تکنیک هایی پیاده کنیم که علی رغم بالا رفتن سرعت شبکه، از دقت آن نکاهد.

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 32, 32, 32)	896
conv2d_1 (Conv2D)	(None, 32, 32, 32)	9248
flatten (Flatten)	(None, 32768)	0
dense (Dense)	(None, 200)	6553800
dense_1 (Dense)	(None, 100)	20100
dense_2 (Dense)	(None, 10)	1010



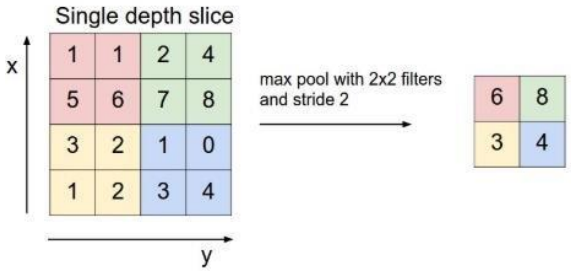

```
Stochastic mini batch based CNN-MLP (Batch-size = 256) on test data:
 Accuracy = 63.70999813079834%
 Test-loss = 1.3793452978134155
 Precision = 65.02650699633617%
 Recall = 63.71%
 f-score = 63.480718076014355%
```

تصویر ۲-۷: خلاصه لایه های شبکه کانولوشنی طراحی شده به همراه دقت و خطا و سایر پارامتر ها

سوال ۲. تأثیر اضافه شدن لایه های Pooling و Batch Normailzation

در این سوال ابتدا در خصوص این دو لایه توضیح می‌دهیم و سپس تأثیر آن را روی شبکه بررسی می‌کنیم.

لایه Pooling: در شبکه های عصبی بعد از اضافه شدن لایه های کانولوشنی باتوجه به بالا رفتن ابعاد و سایز در ورودی لایه های مخفی، از یک لایه Pooling استفاده می‌شود تا سایز و شکل خروجی لایه کانولوشنی کمی کوچک تر شود. تصویر ۲-۸ چگونگی عملکرد این لایه را نشان می‌دهد. اضافه کردن این لایه علی رغم اینکه ممکن است تأثیر زیادی روی دقت نداشته باشد از پیچیدگی به شدت می‌کاهد و سرعت شبکه را بالاتر می‌برد.

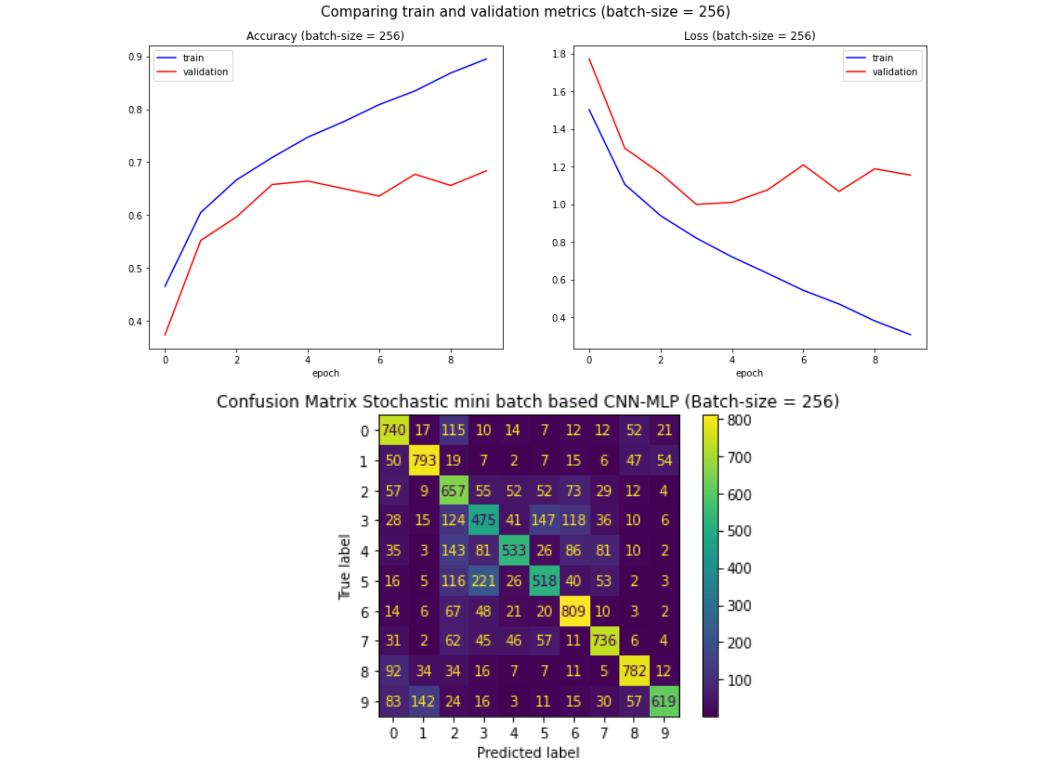


تصویر ۲-۸: عملکرد لایه Pooling

لایه Batch Normalization: در شبکه های عصبی از این لایه برای افزایش سرعت یادگیری استفاده می‌شود. به این منظور که می‌توانیم از نرخ یادگیری بالاتر در بهینه سازی استفاده کنیم. در این لایه یک تکنیک Normalization استفاده می‌شود که در آن به جای Normal کردن کل داده ها سعی می‌کنیم در هر mini batch داده ها را Normal کنیم.

تصویر ۲-۹ نمودار خطا و دقت به همراه پارامتر های مقایسه را به همراه خلاصه لایه های شبکه پس از اضافه شدن لایه های Pooling و Batch Normalization را نشان می‌دهد. همانطور که مشاهده می‌شود سرعت یادگیری شبکه بسیار افزایش یافته است و این در صورتی است که دقت و خطا علاوه بر اینکه کاهش نیافته اند بلکه بهبود نیز پیدا کرده اند. نکته قابل توجه در این سوال و همچنین سوال قبل این است که از یک epoch خاص به بعد، مدل دچار overfitting شده و دقت آن کاهش می‌یابد.

Layer (type)	Output Shape	Param #
conv2d_2 (Conv2D)	(None, 32, 32, 32)	896
batch_normalization (BatchNormalization)	(None, 32, 32, 32)	128
max_pooling2d (MaxPooling2D)	(None, 16, 16, 32)	0
conv2d_3 (Conv2D)	(None, 16, 16, 32)	9248
batch_normalization_1 (BatchNormalization)	(None, 16, 16, 32)	128
max_pooling2d_1 (MaxPooling2D)	(None, 8, 8, 32)	0
flatten_1 (Flatten)	(None, 2048)	0
dense_3 (Dense)	(None, 200)	409800
dense_4 (Dense)	(None, 100)	20100
dense_5 (Dense)	(None, 10)	1010

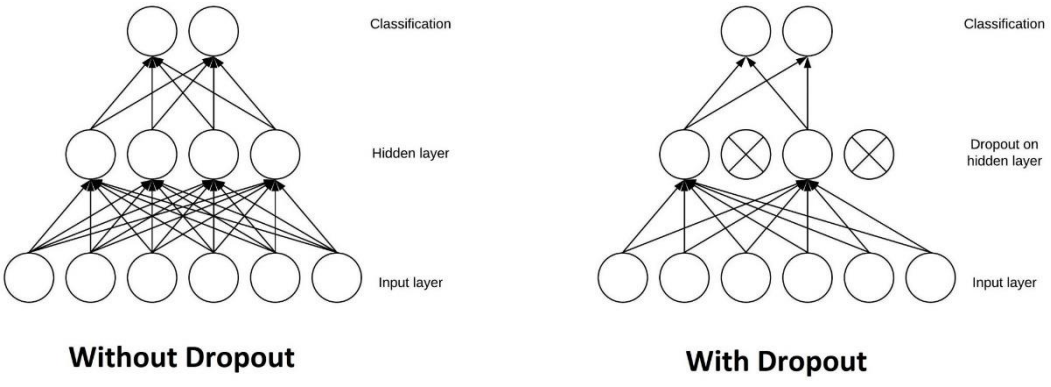



```
Stochastic mini batch based CNN-MLP with pooling and batch normalization(Batch-size = 256) on test data:
 Accuracy = 66.61999821662903%
 Test-loss = 1.207744836807251
 Precision = 67.78356103701262%
 Recall = 66.62%
 f-score = 66.62461943231934%
```

تصویر ۲-۹: خلاصه لایه های شبکه پس از اضافه شدن لایه های Pooling و Batch Normalization به همراه دقت و خطا و سایر پارامتر ها

سوال ۳. تأثیر اضافه شدن لایه های Dropout

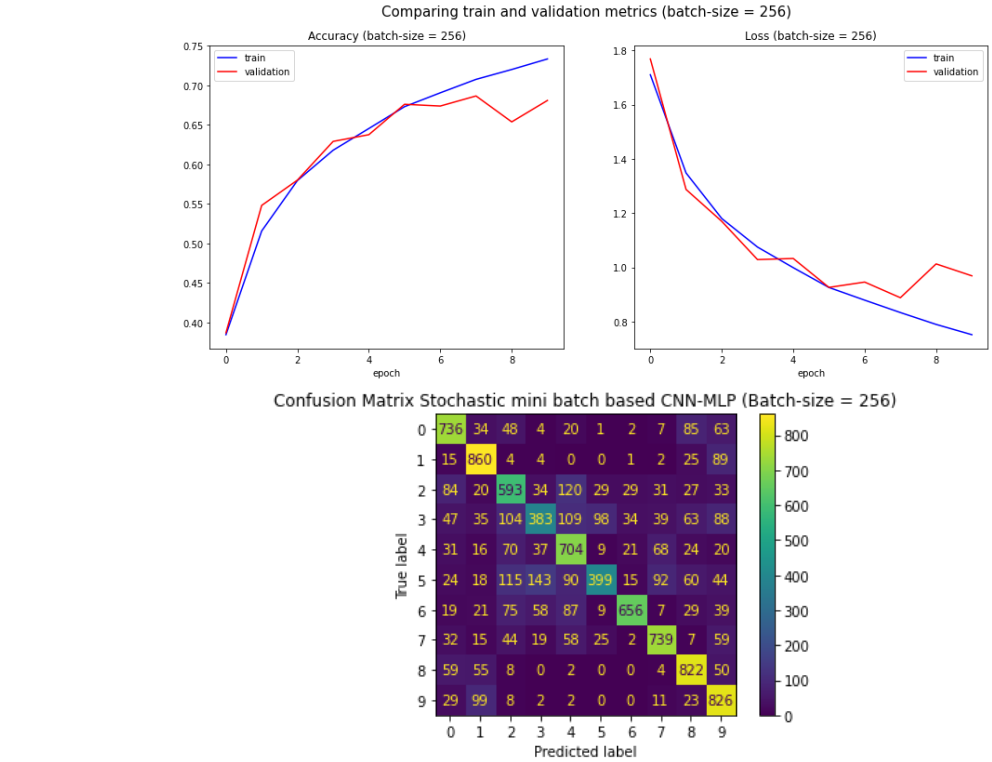
لایه Dropout: در شبکه های عصبی بعضی نورون ها ممکن است اثر منفی در ادامه شبکه داشته باشند و نیاز است در حالت Feed-Forward آن‌ها را در لایه های بعدی بی اثر کنیم. به این منظور از لایه dropout استفاده می‌شود. در این لایه درصدی از نورون های خروجی لایه قبل را در لایه بعدی بی اثر می‌کنیم. این کار علاوه بر افزایش سرعت شبکه روی دقت نیز مؤثر خواهد بود. تصویر ۲-۱۰ نمونه ای از عملکرد این لایه را نشان می‌دهد. از مهمترین شاخصه های این لایه می‌توان به جلوگیری آن از Overfitting اشاره کرد.



تصویر ۲-۱۰: عملکرد لایه Dropout

تصویر ۲-۱۱ نمودار خطا و دقت به همراه پارامتر های مقایسه را به همراه خلاصه لایه های شبکه پس از اضافه شدن لایه های Dropout را نشان می‌دهد. همانطور که مشاهده می‌شود سرعت یادگیری شبکه بسیار افزایش یافته است دقت نیز کاهش نداشته است. همچنین در دفعات تست کردن شبکه، مشاهده می‌شود که دقت آن کاهش ندارد و نسبت به overfittng مقاوم تر شده است.

Layer (type)	Output Shape	Param #
conv2d_4 (Conv2D)	(None, 32, 32, 32)	896
batch_normalization_2 (BatchNormalization)	(None, 32, 32, 32)	128
max_pooling2d_2 (MaxPooling2D)	(None, 16, 16, 32)	0
conv2d_5 (Conv2D)	(None, 16, 16, 32)	9248
batch_normalization_3 (BatchNormalization)	(None, 16, 16, 32)	128
max_pooling2d_3 (MaxPooling2D)	(None, 8, 8, 32)	0
dropout (Dropout)	(None, 8, 8, 32)	0
flatten_2 (Flatten)	(None, 2048)	0
dense_6 (Dense)	(None, 200)	409800
dense_7 (Dense)	(None, 100)	20100
dropout_1 (Dropout)	(None, 100)	0
dense_8 (Dense)	(None, 10)	1010

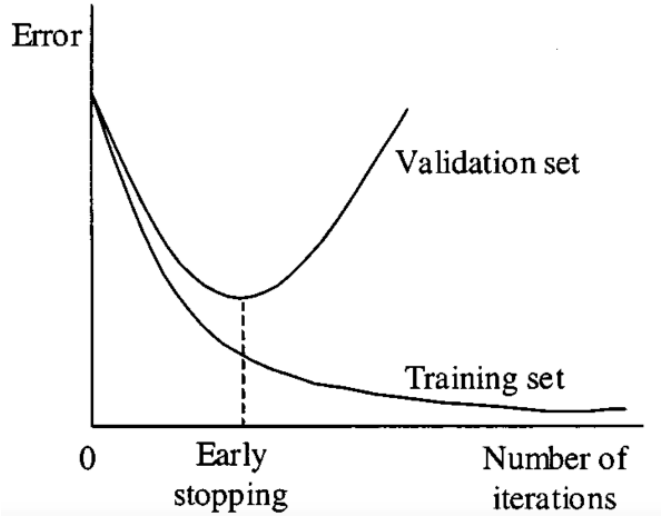



```
Stochastic mini batch based CNN-MLP with pooling, batch normalization, and dropout (Batch-size = 256) on test data:
 Accuracy = 67.18000173568726%
 Test-loss = 0.9824645519256592
 Precision = 67.60237976945372%
 Recall = 67.17999999999999%
 f-score = 66.37947433684938%
```

تصویر ۲-۱۱: خلاصه لایه های شبکه پس از اضافه شدن لایه های Dropout به همراه دقت و خطا و سایر پارامتر ها

سوال ۴. توقف زود هنگام (Early Stop)

در شبکه های عصبی پس از چند epoch مشاهده می‌شود که نمودار خطای داده های validation (یا test) از نمودار خطای داده های train فاصله می‌گیرد و برعکس آن صعودی می‌شود. این یعنی عملکرد شبکه روی داده های test در حال ضعیف شدن است و به این منظور باید در یادگیری شبکه توقف کنیم. راهکار حل این مشکل توقف زودهنگام (Early Stop) است. تصویر ۲-۱۲ نشان می‌دهد که چه جایی نیاز به توقف زود هنگام در یادگیری شبکه داریم.



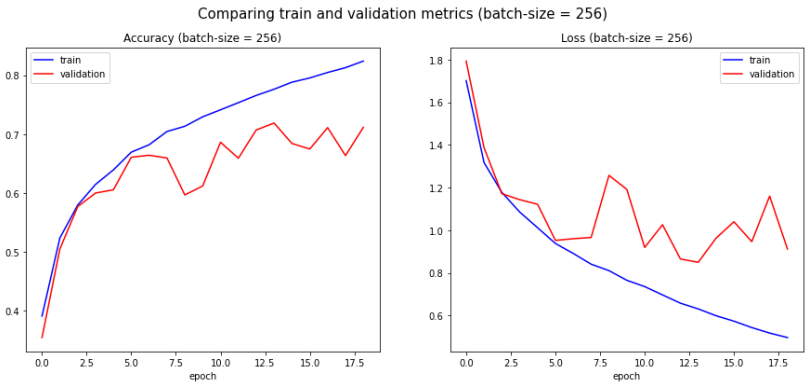
تصویر ۲-۱۲: توقف زودهنگام در شبکه های عصبی

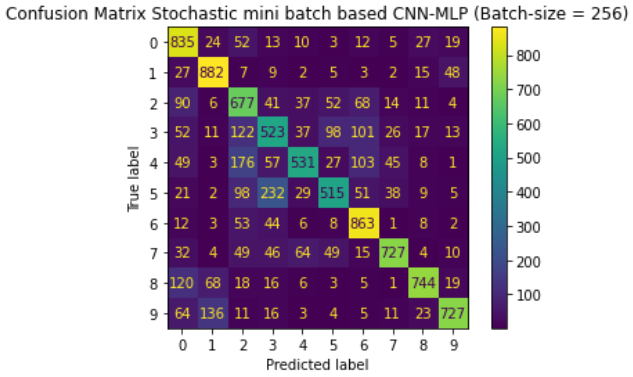
معیار های مورد استفاده در توقف زودهنگام:

برای تشخیص جایی که باید در یادگیری شبکه توقف کنیم دو معیار داریم. یکی فاصله گرفتن نمودار دقت داده های ارزیابی و یادگیری و دیگری فاصله گرفتن نمودار خطای این دو مجموعه داده. علاوه بر آن باید توجه کنیم که تا چه حد روی فاصله گرفتن این دو نمودار حساس باشیم. با استفاده از دستور EarlyStopping() زیر مجموعه کتابخانه keras می‌توانیم این معیار ها را تعیین کنیم.

تصویر ۲-۱۳ پارامتر های خروجی پس از پیاده سازی early stopping روی شبکه عصبی را نشان می‌دهد. همانطور که مشاهده می‌شود در epoch، ۱۹م بعد از فاصله گرفتن نمودار خطای داده validation (حساسیت با پارامتر patience تعیین می‌شود) این نمودار صعودی نمی‌شود و فرآیند یادگیری متوقف می‌شود. در کل اگر تعداد epoch ها را هم زیاد تر در نظر می‌گرفتیم، در صورتی که فاصله افزایش پیدا می‌کرد یادگیری شبکه متوقف می‌شد.

```
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_16 (Conv2D)           (None, 32, 32, 32)        896

batch_normalization_14 (Bat  (None, 32, 32, 32)        128
chNormalization)

max_pooling2d_14 (MaxPoolin  (None, 16, 16, 32)        0
g2D)

conv2d_17 (Conv2D)           (None, 16, 16, 32)        9248

batch_normalization_15 (Bat  (None, 16, 16, 32)        128
chNormalization)

max_pooling2d_15 (MaxPoolin  (None, 8, 8, 32)          0
g2D)

dropout_12 (Dropout)         (None, 8, 8, 32)          0

flatten_8 (Flatten)          (None, 2048)              0

dense_24 (Dense)             (None, 200)               409800

dense_25 (Dense)             (None, 100)               20100

dropout_13 (Dropout)         (None, 100)               0

dense_26 (Dense)             (None, 10)                1010

=================================================================
```






```
Stochastic mini batch based CNN-MLP with early stopping (Batch-size = 256) on test data:
 Accuracy = 70.24000287055969%
 Test-loss = 0.9256798624992371
 Precision = 71.38762219480752%
 Recall = 70.24000000000001%
 f-score = 70.08945596689409%
```

تصویر ۲-۱۳: خلاصه لایه های شبکه پس از اضافه شدن early stop به همراه دقت و خطا و سایر پارامتر ها

*** منابع استفاده شده در این بخش:

- https://machinelearningmastery.com/how-to-stop-training-deep-neural-networks-at-the-right-time-using-early-stopping/#:~:text=Early%20stopping%20is%20a%20method,a%20hold%20out%20validation%20dataset.
- https://machinelearningmastery.com/how-to-develop-a-cnn-from-scratch-for-cifar-10-photo-classification/

- https://github.com/laplacetw/vgg-like-cifar10/blob/master/keras_sample_cifar10.py
- https://marcinbogdanski.github.io/ai-sketchpad/KerasNN/1200_CNN_BN_CIFAR10.html
- https://gaussian37.github.io/ML_DL-Code-EarlyStopping-with-Keras/#:~:text=In%20order%20to%20early%20stop,not%20improve%20the%20learning%20status

سوال ۳: یادگیری انتقال یافته برای شبکه EfficientNet

در این سوال از تمرین قصد داریم با شبکه EfficientNetB0 آشنا شویم. ابتدا در مورد معماری شبکه توضیحاتی می‌دهیم و سپس از آن در طبقه بندی تصاویر استفاده می‌کنیم. در نهایت نیز سعی می‌کنیم ایده Transfer Learning را با استفاده از این شبکه پیاده سازی کنیم.

قسمت الف: آشنایی با شبکه EfficientNetB0

۱- معماری شبکه: این شبکه تحت عنوان یک شبکه عصبی کانولوشنی به‌وجود آمده است که در لایه های آن علاوه بر لایه های کانولوشنی از لایه های MBConv (mobile inverted bottleneck) نیز استفاده شده است. مجموعاً ۱۸ لایه کانولوشنی تحت ۹ مرحله (۹ سایز خروجی مختلف) در معماری این شبکه استفاده شده است. خروجی این شبکه یک آرایه به طول ۱۰۰۰ است که بیانگر ۱۰۰۰ Category مربوط به دادگان است.

۲- نسخه های مختلف معماری و تفاوت آن‌ها: در مقاله داده شده فرمول بندی مسئله EfficirntNetB0 داده شده است. در حقیقت سایر نسخه های EfficientNetBx با تغییر پارامتر ϕ در فرمول بندی EfficientNetB0 به وجود آمده اند و در جدول های مقاله مقایسه نسخه های مختلف از نظر پیچیدگی (تعداد پارامتر ها) و دقت مقایسه شده اند.

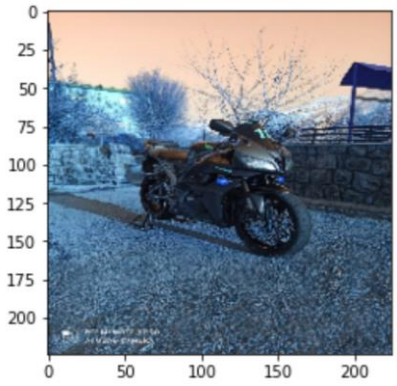
۳- پیش پردازش های اولیه برای تصویر ورودی: در پیش پردازش های مربوط به این شبکه، باید تصویر ابعاد (224, 224, 3) داشته باشد. به این منظور در پیاده سازی ها از کتابخانه cv2 استفاده می‌کنیم.

۴- مزایا نسبت به سایر مدل ها: مهم ترین ویژگی و مزیت این شبکه که در قسمت نتیجه گیری مقاله نیز به آن اشاره شده است، دقت شبکه برای دادگان مختلف نسبت به سایر مدل ها و استفاده در Trnasfer Learning است.

قسمت ب: پیاده سازی شبکه به کمک ایده Transfer Learning

در این قسمت می‌خواهیم عملکرد شبکه EfficientNetB0 را برای یک تصویر دلخواه بررسی کنیم. به این منظور به عنوان ورودی تصویر ۳-۱ را در نظر می‌گیریم و بعد از آماده کردن آن (از نظر سایز و ...) به شبکه می‌دهیم.

سپس ۳ دسته با بیشترین احتمال پیش‌بینی را مطابق با تصویر ۳-۲ نشان می‌دهیم. همانطور که مشاهده می‌شود با احتمال نسبتاً خوب، تصویر به خوبی طبقه بندی شده است. (تابع مربوط به این قسمت با نام Part_b() نوشته شده است.)



تصویر ۳-۱: تصویر انتخاب شده برای ورودی شبکه EfficientNetB0

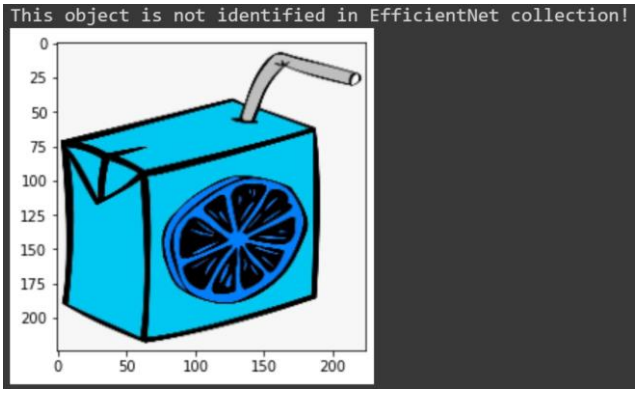
```
Label 1: moped  with Probability 0.2855878174304962
Label 2: motor_scooter  with Probability 0.23792120814323425
Label 3: crash_helmet  with Probability 0.03290800005197525
```

تصویر ۳-۲: سه دسته پیش‌بینی شده با بیشترین احتمال ها برای تصویر ۳-۱

قسمت ج: رفع یک مشکل خاص شبکه

در این قسمت می‌خواهیم یک ایده برای وقتی که شیء در تصویر ورودی جز مجموعه اشیا در Efficient نباشند را پیاده سازی کنیم. راحت ترین راه حل برای رفع این مشکل این است که یک آستانه تعیین کنیم که اگر بزرگترین احتمال ها از آن کوچکتر باشند، دسته ای تشخیص داده نشود و اطلاع داده شود که شیء در مجموعه وجود ندارد. این ایده در تابع Part_c() پیاده‌سازی شده است.

برای مثال در مجموعه EfficientNetB0 آبمیوه و ... وجود ندارد. مطابق با تصویر ۳-۳ یک تصویر آبمیوه به شبکه می‌دهیم و ایده را پیاده سازی می‌کنیم. همانطور که مشاهده می‌شود، ایده به درستی کار می‌کند.



تصویر ۳-۳: رفع مشکل نبودن شیء در مجموعه دادگان شبکه *EfficientNetB0*

ایده های دیگر: یک ایده دیگر این است که لایه آخر شبکه را حذف کرده و به جای آن یک لایه با خروجی به طول ۱ (Dense(1)) اضافه کنیم که 0 بودن آن بیانگر نبودن شیء در مجموعه اشیا باشد.

قسمت د: آموزش شبکه با مجموعه دادگان جدید

در این قسمت می‌خواهیم، با استفاده از یک مجموعه دادگان با دو کلاس، شبکه ای به کمک EfficientNetB0 طراحی کنیم که بتوان با کمک آن طبقه بندی بین دو کلاس را انجام داد.

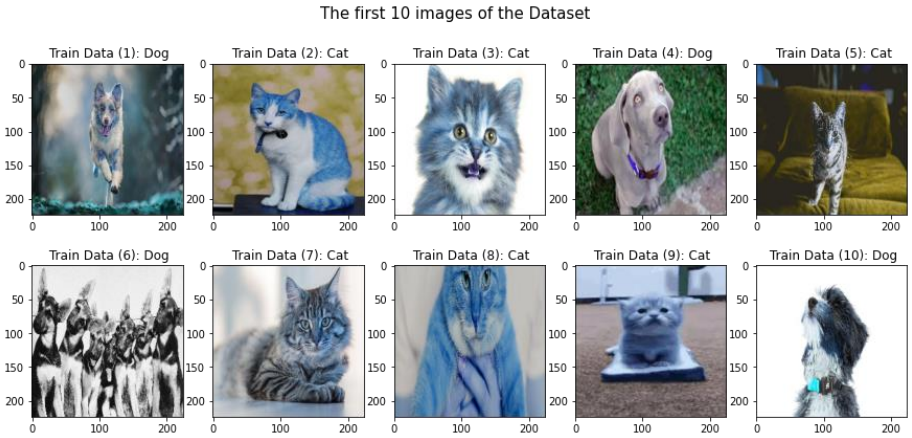
به این منظور دادگانی شامل دو کلاس گربه و سگ را انتخاب می‌کنیم. از مجموع ۶۹۲ تصویر، ۴۰۰ تصویر را به عنوان داده آموزشی و مابقی را به عنوان داده تست در نظر می‌گیریم.

Dataset Link: https://www.kaggle.com/datasets/samuelcortinhas/cats-and-dogs-image-classification

(تصویر dog_505.jpg سیاه و سفید بوده و تنسور نیست و به همین منظور آن‌را از مجموعه دادگان حذف می‌کنیم.)

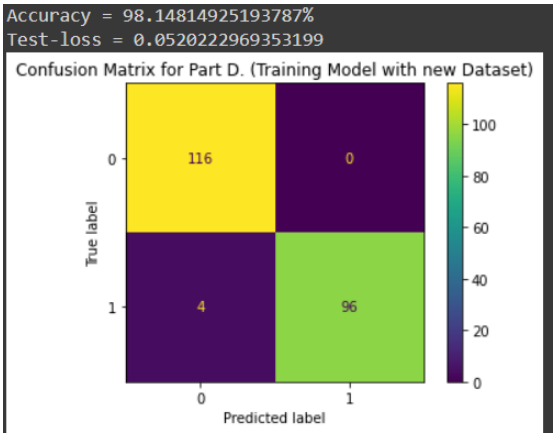
تصویر ۳-۴، ۱۰ داده اول مجموعه آموزشی را نشان می‌دهد.

حال برای اینکه بتوانیم طبقه بندی طراحی کنیم که این دو کلاس را از هم جدا کند، ابتدا لایه انتهایی شبکه EfficientNetB0 را حذف می‌کنیم و آنرا با یک لایه با سایز خروجی ۲ جایگزین می‌کنیم. دلیل اینکه لایه آخر خروجی با سایز ۲ داشته باشد بخاطر تعداد کلاس های مجموعه است. تصویر ۳-۵ مقادیر دقت و خطا روی داده های آموزش و در نهایت دقت و ماتریس آشفتگی (confusion matrix) را روی دادگان آزمون نشان می‌دهد.



تصویر ۳-۴: ۱۰ داده اول مجموعه دادگان گربه و سگ به همراه برچسب

```
Epoch 1/10
13/13 [==============================] - 11s 276ms/step - loss: 0.5426 - accuracy: 0.7200
Epoch 2/10
13/13 [==============================] - 4s 281ms/step - loss: 0.1853 - accuracy: 0.9525
Epoch 3/10
13/13 [==============================] - 4s 291ms/step - loss: 0.1040 - accuracy: 0.9650
Epoch 4/10
13/13 [==============================] - 4s 284ms/step - loss: 0.0593 - accuracy: 0.9875
Epoch 5/10
13/13 [==============================] - 4s 280ms/step - loss: 0.0295 - accuracy: 0.9975
Epoch 6/10
13/13 [==============================] - 4s 281ms/step - loss: 0.0260 - accuracy: 1.0000
Epoch 7/10
13/13 [==============================] - 4s 277ms/step - loss: 0.0170 - accuracy: 1.0000
Epoch 8/10
13/13 [==============================] - 4s 281ms/step - loss: 0.0119 - accuracy: 1.0000
Epoch 9/10
13/13 [==============================] - 4s 281ms/step - loss: 0.0115 - accuracy: 1.0000
Epoch 10/10
13/13 [==============================] - 4s 281ms/step - loss: 0.0102 - accuracy: 1.0000
```



تصویر ۳-۵: یادگیری شبکه و مقادیر دقت و خطا در هر دوره و دقت و ماتریس آشفتگی روی داد های تست

*** منابع استفاده شده در این بخش:

- https://stackabuse.com/courses/practical-deep-learning-for-computer-vision-with-python/lessons/image-classification-with-transfer-learning-creating-cutting-edge-cnn-models/
- https://deeplearning.cms.waikato.ac.nz/user-guide/class-maps/IMAGENET/
- https://keras.io/api/applications/efficientnet/
- https://paperswithcode.com/method/efficientnet