|  |  |  |  |
| --- | --- | --- | --- |
| **학습 일지** | | | |
| **작성자** | 이수진 | **학습일자** | 2020년 4월 22일 |
| **강의실** |  | **시간** | 09:00 ~ 18:00 |

* **학습내용**

|  |  |
| --- | --- |
| **시간** | **주요내용** |
| 09:00~09:50 | * **Loss nan 문제 🡪 loss 함수를 변경하여 해결**   **기존:**  **def lossless\_triplet\_loss(y\_true, y\_pred, N=3, beta=3, epsilon=1e-8):**  **"""**  **Implementation of the triplet loss function**    **Arguments:**  **y\_true -- true labels, required when you define a loss in Keras, you don't need it in this function.**  **y\_pred -- python list containing three objects:**  **anchor -- the encodings for the anchor data**  **positive -- the encodings for the positive data (similar to anchor)**  **negative -- the encodings for the negative data (different from anchor)**  **N -- The number of dimension**  **beta -- The scaling factor, N is recommended**  **epsilon -- The Epsilon value to prevent ln(0)**      **Returns:**  **loss -- real number, value of the loss**  **"""**  **anchor = tf.convert\_to\_tensor(y\_pred[:,0:N])**  **positive = tf.convert\_to\_tensor(y\_pred[:,N:N\*2])**  **negative = tf.convert\_to\_tensor(y\_pred[:,N\*2:N\*3])**    **# distance between the anchor and the positive**  **pos\_dist = tf.reduce\_sum(tf.square(tf.subtract(anchor,positive)),1)**  **# distance between the anchor and the negative**  **neg\_dist = tf.reduce\_sum(tf.square(tf.subtract(anchor,negative)),1)**    **#Non Linear Values**    **# -ln(-x/N+1)**  **pos\_dist = -tf.math.log(-tf.divide((pos\_dist),beta)+1+epsilon)**  **neg\_dist = -tf.math.log(-tf.divide((N-neg\_dist),beta)+1+epsilon)**    **# compute loss**  **loss = neg\_dist + pos\_dist**    **return loss**  **변경 후:**  **def triplet\_loss(y\_true, y\_pred, alpha = 0.2):**  **"""**  **Implementation of the triplet loss function**  **Arguments:**  **y\_true -- true labels, required when you define a loss in Keras, not used in this function.**  **y\_pred -- python list containing three objects:**  **anchor: the encodings for the anchor data**  **positive: the encodings for the positive data (similar to anchor)**  **negative: the encodings for the negative data (different from anchor)**  **Returns:**  **loss -- real number, value of the loss**  **"""**  **anchor, positive, negative = y\_pred[0], y\_pred[1], y\_pred[2]**  **# distance between the anchor and the positive**  **pos\_dist = tf.reduce\_sum(tf.square(tf.subtract(anchor,positive)))**  **# distance between the anchor and the negative**  **neg\_dist = tf.reduce\_sum(tf.square(tf.subtract(anchor,negative)))**  **# compute loss**  **basic\_loss = pos\_dist-neg\_dist+alpha**  **loss = tf.maximum(basic\_loss,0.0)**  **return loss**   * **Triplet sampling 함수 작성** * **모델 학습 및 효율 문제로 중단** |
| 10:00~10:50 |
| 11:00~11:50 |
| 12:00~12:50 | **중식** |
| 13:00~13:50 | * **AWS 환경설정** |
| 14:00~14:50 |
| 1500~15:50 |
| 16:00~16:50 |
| 17:00~17:50 |
| 18:00~18:50 |
| **기타**   * **Lossless\_triplet\_loss(기존)가 triplet\_loss(변경 후) 함수보다 더 업데이트된 버전의 loss 함수이기 때문에 모델 성능이 잘 안 나올 경우 Lossless\_triplet\_loss로 다시 변경해야 할 수 있음** | |