1. Czym się różnią mechanizmy atencji i self-atencji stosowane w architekturze Transfomers?
2. Podaj i wyjaśnij pokrótce przekształcenie stosowane w self-atencji.
3. Czemu służy element Positional Encoding w architekturze Transformers?

Oto odpowiedzi na Twoje pytania:

1. **Różnice między mechanizmem atencji a self-atencją w architekturze Transformers:**
   * **Mechanizm atencji:** W mechanizmie atencji uwaga jest skupiana na różnych częściach sekwencji danych wejściowych w kontekście innych sekwencji. To oznacza, że każdy element w sekwencji może być ważny dla każdego innego elementu.
   * **Self-atencja (atencja samoistna):** Jest to specjalny przypadek mechanizmu atencji, gdzie uwaga jest skupiana na różnych częściach tej samej sekwencji. Każdy element ma możliwość uwzględniania innych elementów w tej samej sekwencji, co pozwala na bardziej elastyczne analizowanie zależności wewnątrz sekwencji.
2. **Przekształcenie stosowane w self-atencji:**
   * W self-atencji stosowane jest przekształcenie liniowe dla każdego z trzech wektorów reprezentujących każdy token w sekwencji: wektoru zapytania (query), wektora klucza (key), i wektora wartości (value).
   * Dla każdego tokena, generowane są te trzy wektory poprzez przemnożenie wektora wejściowego przez odpowiednie macierze wag. Następnie, obliczana jest skalarowa suma iloczynów skalarnych pomiędzy wektorem zapytania, a wektorami kluczy, a wyniki są używane do obliczenia wag uwagi dla każdego tokenu.
   * Na koniec, wagi są używane do ważonego sumowania wektorów wartości, co daje ostateczne wyjście self-atencji dla danego tokenu.
3. **Rola elementu Positional Encoding w architekturze Transformers:**
   * W architekturze Transformers, model nie zachowuje informacji o kolejności tokenów w sekwencji, ponieważ operacja self-atencji jest permutacyjnie niezmiennicza.
   * Aby uwzględnić informacje o pozycji tokenów, używa się elementu Positional Encoding. Jest to dodawane do reprezentacji wejściowej przed przekazaniem jej do warstwy self-atencji.
   * Positional Encoding dostarcza modelowi informacji o pozycji tokenów w sekwencji, co jest istotne dla zrozumienia kontekstu i zachowania kolejności w danych wejściowych. Najczęściej używane są funkcje sinusoidalne i kosinusoidalne w celu kodowania pozycji.