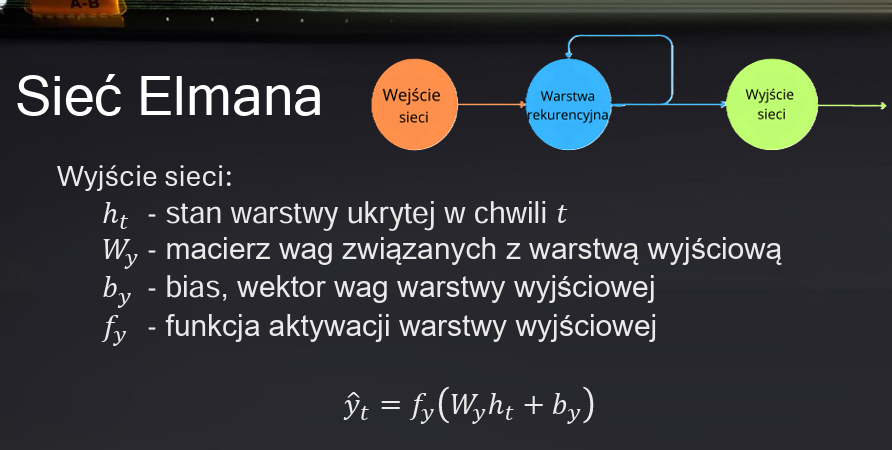
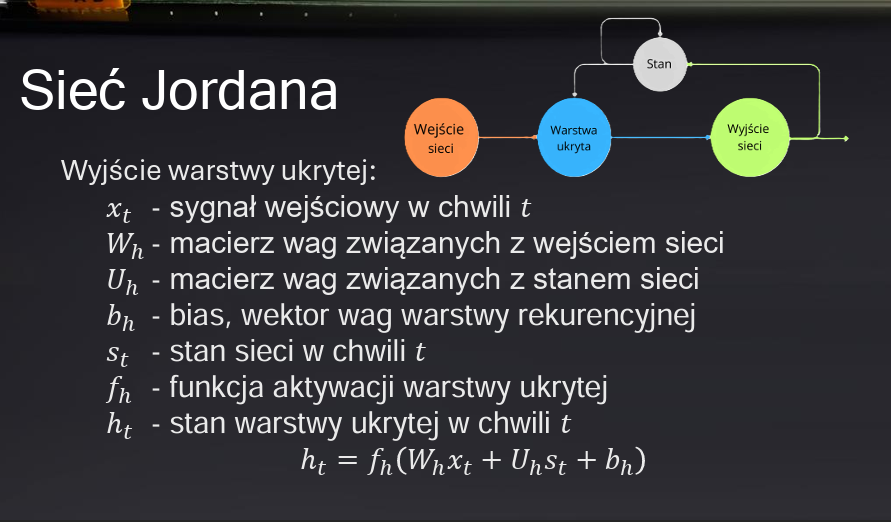
Sieci rekurencyjne wiadomo co to jest. Sieć posiada pewną pamięć potrafi zinterpretować daną w zależności od poprzedniej.

Wyróżniamy różne typy rodzaje:

Sieci Hopfielda, BAM (bidarectional assosiation memmory), Sieci Elmana , Siecie Jordana, LSTM, GRU, Echo State.



Siecie Elmana to podstawowe RNN-> sygnał wejściowy podawany jest na wejście sieci potem przemnażany jest przez wagi, następnie w warstwie rekurencyjnej dodawany jest poprzedni sygnał który ta warstwa wysłała połączenie wewnątrz warstwy ukrytej, mnożone przez jakaś macierz u dodawany bias i wszystko jest otoczone funkcją aktywacji. W rezultacie powstaje sygnał ht ukryty będzie on przesyłany do wyjścia sieci oraz propagowany z powrotem. Następnie mając sygnał ht wyjście jest obliczane przez wymnożenie go przez macierz i oddanie obciążenia- klasycznie



W sieciach Jordana podejście jest trochę inne rekurencja ma miejsce z wyjścia do warstwy ukrytej, wyróżnia pojęcia stan ukryty oraz stan sieci. Mamy wejście wagi i bias i na podstawie stanu obliczany jest sygnał ht. Wyjście sieci obliczane tak jak w każdej do tej pory sieci. Wyliczone wyjście jest kierowane też do komórki stan sieci pobierany jest poprzedni stan i to pozwala obliczyć odpowiednie wyjście.

Problemy z uczeniem-> zanikanie gradientu czyli gradient jest tak mały że nie pozwala modyfikować wag lub eksplodujący gradient staje się na tyle duży że sieć skacze ale nie może osiągnąć zbieżności.

Rozróżnia się też sieci ze względu na połączenia jakie są w sieci

Wiele do jednego-> podajemy na sieć sekwencje słów aby otrzymać informacje czy tekst jest spamem czy nie

K do K-> wprowadzam sekwencje przemówienia i każdy z elementu dostanie fragment opisu auto deskrypcja, analiza danych giełdowych

Wiele do wielu-> tłumaczenie

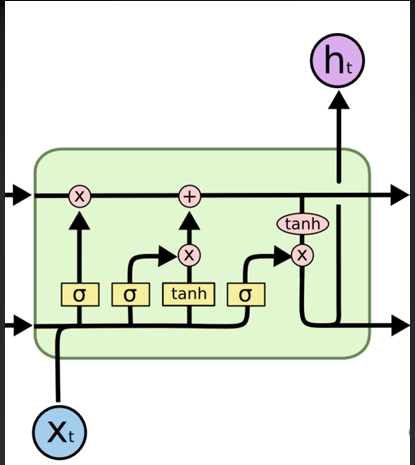
Jeden do wielu-> modele generatywne

Jak uczymy RNN-> Wsteczna propagacja przez czas trzeba zmodyfikować wsteczną propagacje błędów tak aby uwzględniała sekwencyjność.

Mamy zbiór uczący w postaci ciągu sygnałów wejściowych i że n jest duże [x1…,xn] możemy sobie to wyobrazić tak że y to będzie słowo które trzeba przewidzieć. Bierzemy pierwsze słowo, obliczamy wartość ukrytą i h1 jest wysyłane do wyjścia i obliczane jest yhat 1 i tak dalej z czego stany ukryte są przekazywane do kolejnych iteracji. Wagi nie ulegają zmianom co komplikuje liczenie gradientów. Tutaj coś wspomniane jeżeli w macierzy u pierwszy element jest większy od 1 to gradient możne eksplodować.

Clipping gradient-> zamykanie gradientu czyli ograniczamy ich maksymalną wartość.

Innym sposobem na rekuraryzacje uczenie jest przyjęcie określonej struktury prym wiedzie LSTM.



Po pierwsze mamy stan komórki i jest to jedne rodzaj pamięci który będzie biegł przez całą sieć drugi to stan ukryty łączony jest on z wejściem czyli sygnałem podawanym w chwili t. Czyli tworzymy połączony sygnał. Następnie ten połączony wektor wykorzystywany jest do ustalenia czy ten stan komórki należy zapomnieć czy nie. Żeby to zrobić obliczana jest funkcja sigmoidalna i po przez przemnożenie poprzedniego stanu z funkcją sigmoidalną czy stan ma być zapomniany czy nie-> ten fragment sieci nazywa się bramkom zapominania. Następnie ten sam sygnał próbuje wnieś coś od siebie do stanu komórki rozdzielany jest na dwie części jedna opatrzona funkcją sigmoidalną (czy warto dodać) a druga tangensem hiperbolicznym(ile warto dodać wzmocnienie lub osłabienie), wyniki tych działań są ze sobą przemnażane- tą część sieci nazywamy bramkom wejścia. Na koniec dalej ten sam sygnał otaczany jest funkcją sigmoidalną i połączone to będzie z wyjście (iloczyn stanu komórki z tangensa hiperbolicznego i sigmoida ze stanu ukrytego) -ta część to bramka wyjścia.