

P O L I T E C H N I K A R Z E S Z O W S K A

im. Ignacego Łukasiewicza

WYDZIAŁ MATEMATYKI I FIZYKI STOSOWANEJ

**Aldona Świrad**

Predykcja szeregów czasowych: metody klasyczne i elementy uczenia maszynowego

**Praca dyplomowa inżynierska**

Opiekun pracy:  
dr Ewa Rejwer-Kosińska

Rzeszów, 2024

Spis treści

[1. Wstęp 5](#_Toc183432950)

[*Cel pracy* 5](#_Toc183432951)

[*Zakres pracy* 5](#_Toc183432952)

[1. Wprowadzenie teoretyczne 5](#_Toc183432953)

[4. Metody uczenia maszynowego dla szeregów czasowych 8](#_Toc183432954)

[4.1 Model RNN 8](#_Toc183432955)

[4.2 Model LSTM 12](#_Toc183432956)

Wstęp

1. Wprowadzenie teoretyczne
   1. Definicja szeregów czasowych
   2. Model ARIMA
2. Analiza wybranych szeregów czasowych metodami klasycznymi
   1. Analiza – opis, wykresy, zakres zmienności
   2. Dobór modelu
3. Metody uczenia maszynowego dla szeregów czasowych

4.1 Model RNN

4.2 Model LSTM

1. Prognozowanie szeregów czasowych

5.1 Prognozowanie metodą ARIMA

5.2 Prognozowanie metodą LSTM

5.3 Porównanie

Podsumowanie

Bibliografia

Dodatek

# Wstęp

Tekst akapitu.

Tekst akapitu.

**Cel pracy**

Celem niniejszej pracy inżynierskiej jest zbadanie i porównanie skuteczności metod analizy szeregów czasowych: metod klasycznych oraz metod opartych na uczeniu maszynowym, ze szczególnym uwzględnieniem modeli LSTM. Głównym celem pracy jest zastosowanie tych modeli do prognozowania szeregów czasowych oraz ocena ich przydatności w różnych scenariuszach analizy danych czasowych.

**Zakres pracy**

W pierwszym rozdziale pracy podane zostaną najważniejsze pojęcia z zakresu modelowania i predykcji szeregów czasowych. Rozdział drugi poświęcony zostanie analizie własności i modelowaniu wybranych szeregów czasowych, z zastosowaniem metod klasycznych. W rozdziale trzecim podane zostaną podstawowe pojęcia i metody uczenia maszynowego (w tym modele RNN i LSTM), stosowane w analizie szeregów czasowych. Porównanie wybranych metod prognozowania szeregów czasowych pod kątem ich skuteczności, wraz z wnioskami dotyczącymi przydatności poszczególnych modeli w różnych scenariuszach analizy szeregów czasowych, będą zawarte w ostatnim rozdziale pracy. W podsumowaniu zawarte będą sugestie dotyczące dalszych badań oraz potencjalnych zastosowań praktycznych uzyskanych wyników.

Praca będzie opierać się głównie na analizie literatury naukowej oraz eksperymentach przeprowadzonych na rzeczywistych danych, co umożliwi kompleksowe zbadanie tematu oraz wyciągnięcie trafnych wniosków dotyczących analizy szeregów czasowych za pomocą metod klasycznych i uczenia maszynowego.

# 1. Wprowadzenie teoretyczne

Dodać: (trend, sezonowość, proces stochastyczny, biały szum, def. Parametry, szereg czasowy – parametry, wykresy i rysunki obrazujące szereg czasowy, poprawić wzory)

Wprowadzenie teoretyczne ma na celu przybliżenie podstawowych pojęć związanych z szeregami czasowymi oraz zaprezentowanie jednego z najczęściej wykorzystywanych modeli do ich analizy – modelu ARIMA. Zrozumienie tych pojęć oraz metodologii jest kluczowe do poprawnej analizy i prognozowania szeregów czasowych.

2.1 Definicja i własności szeregów czasowych

Szereg czasowy (ang. *time series*) to sekwencja danych gromadzonych w równych odstępach czasu, np. dziennie, miesięcznie, rocznie. Każda obserwacja w szeregu czasowym składa się z dwóch elementów: momentu w czasie, do którego się odnosi, oraz wartości zmiennej mierzonej w tym momencie. Przy czym, kolejność obserwacji ma istotne znaczenie. Przykładami szeregów czasowych mogą być kursy walut, wartości indeksów giełdowych, liczba sprzedanych produktów w danym okresie, czy zmiany temperatury w ciągu roku.

W praktyce, analiza szeregów czasowych ma na celu identyfikację struktury danych, wyodrębnienie trendów, sezonowości oraz innych wzorców, a także prognozowanie przyszłych wartości na podstawie dostępnych danych historycznych, czym będziemy się zajmować w tej pracy.

***Stacjonarność szeregu czasowego***

Szeregi czasowe mogą być stacjonarne lub niestacjonarne. Szereg stacjonarny charakteryzuje się stałą średnią, wariancją i autokorelacją w czasie, natomiast szereg niestacjonarny może zawierać zmieniające się średnie, wariancje lub inne nieliniowe zależności. Aby skutecznie analizować szeregi czasowe i budować modele prognostyczne, często stosuje się metody przekształcania niestacjonarnych szeregów na stacjonarne. **Proszę wymienić i bardziej szczegółowo omówić kilka z tych metod**

***Trend i sezonowość***

**Zamiast podrozdziałów w podrozdziale w taki sposób proszę odnosić się do innych charakterystyk szeregów czasowych, które będzie Pani opisywać (czyli biały szum, itp.)**

***Różnicowanie niestacjonarnego szeregu czasowego***

Różnicowanie służy do przekształcania niestacjonarnych szeregów czasowych   
w szeregi stacjonarne. Proces ten eliminuje trend i sezonowość, co ułatwia modelowanie fluktuacji w krótszych okresach.

Proces różnicowania polega na obliczeniu różnicy między kolejnymi obserwacjami:

**(Różnicowanie będzie Pani wykonywać podczas analizy szeregów czasowych, także być może tutaj będzie trzeba coś więcej dopisać)**

2.2 Model ARIMA

Model ARIMA (ang. *AutoRegressive Integrated Moving Average*) to jeden   
z najpopularniejszych modeli stosowanych w analizie szeregów czasowych. Jest to model liniowy, który łączy trzy kluczowe elementy: autoregresję (AR), różnicowanie (I) i średnią ruchomą (MA).

***Model autoregresyjny AR(p)***

Autoregresja polega na przewidywaniu wartości zmiennej w danym momencie na podstawie wcześniejszych obserwacji tej samej zmiennej. Model *AR(p)*, gdzie parametr *p* oznacza liczbę opóźnionych obserwacji (lagów), zakłada, że wartość zmiennej zależy od jej wcześniejszych wartości:

**(Proszę pisać równania w kreatorze równań, nie jako wstawione zdjęcia ! Ewentualne poprawki/zmiany będzie wtedy można wstawić bez problemu)**

gdzie:

- wartość zmiennej w czasie t,

– współczynniki modelu,

– błąd losowy w czasie t.

***Model średniej ruchomej MA(q)***

Średnia ruchoma zakłada, że bieżąca wartość zmiennej jest kombinacją poprzednich błędów losowych. Model *MA(q)*, gdzie *q* oznacza liczbę opóźnień błędów losowych, zakłada, że wartość zmiennej w danym momencie zależy od wcześniejszych błędów:

gdzie:

– błędy losowe,

– współczynniki modelu.

***Model ARIMA(p,d,q)***

Model *ARIMA* łączy w sobie: autoregresję (*p*), różnicowanie (*d*) oraz średnią ruchomą (*q*). Model ten opisuje szereg czasowy za pomocą kombinacji wcześniejszych wartości zmiennej oraz błędów losowych, przy jednoczesnym uwzględnieniu odpowiedniej liczby różnicowań, w celu zapewnienia stacjonarności. Ogólna postać modelu *ARIMA(p, d, q)* wygląda następująco:

gdzie:

– zróżnicowanie wartości zmiennej,

– współczynniki autoregresji,

– współczynniki średniej ruchomej,

– błąd losowy w czasie t.

Model ARIMA jest szeroko stosowany w prognozowaniu szeregów czasowych ze względu na swoją elastyczność oraz zdolność do modelowania różnych wzorców, takich jak trendy i sezonowość. Jednak, dobór odpowiednich wartości parametrów modelu wymaga analizy danych oraz testów diagnostycznych, takich jak autokorelacja reszt czy testy stacjonarności.

# Metody uczenia maszynowego dla szeregów czasowych

Uczenie maszynowe (*ang. machine learning*) to dziedzina informatyki, zajmująca się tworzeniem różnych algorytmów i modeli pozwalającym na samodzielne uczenie się z danych przez dany program lub maszynę. Właściwość ta pozwala na rozwiązywanie problemów zbyt trudnych i skomplikowanych, by można było je rozwiązać za pomocą tradycyjnego programowania lub innymi metodami klasycznym. Nawet problemy które są rozwiązywalne inaczej, mogą być rozwiązywane metodami uczenia maszynowego z racji na inne podejście do danego zagadnienia.

Samo uczenie maszynowe to tylko część sztucznej inteligencji (*ang. artificial intellligence*)*.* Natomiast w zbiorze metod uczenia maszynowego można wyróżnić jedne z najbardziej popularnych sieci neurolonych (*ang. neural networks*), a jeszcze dalej uczenie głębokie (*ang. deep learning*). Tam właśnie można wyróżnić metody, którymi będę zajmować się w tej pracy i które opiszę poniżej. RNN jest jedną z metod uczenia głębokiego, natomiast LSTM jest rodzajem metody RNN.

Obraz zawierający tekst, krąg

Opis wygenerowany automatycznie

Cechą charakterystyczną, która odróżnia klasyczne programowanie od uczenia maszynowego jest fakt, że w metodzie klasycznej dostarczamy dane wejściowe oraz samodzielnie tworzymy algorytm rozwiązujący dany problem, dzięki czemu dostajemy dane wyjściowe, będące np. rozwiązaniem. Stosując uczeni maszynowe dostarczamy jedynie dane – wejściowe i wyjściowe, które można podzielić na **dane treningowe i testowe**, gdzie dane treningowe służą do nauki modelu, natomiast dane testowe pozwalają sprawdzić jego skuteczność na nieznanych wcześniej przykładach. Można również wyróżnić **dane walidacyjne**, które służą do wstępnej oceny modelu podczas jego budowy. Dostarczone dane są przez program interpretowane za pomocą neuronów, które są w swej istocie różnymi równaniami matematycznymi. Następnie otrzymujemy algorytm, który może posłużyć do dalszej pracy już z danymi, które np. chcemy prognozować.

Obraz zawierający tekst, zrzut ekranu, diagram, design

Opis wygenerowany automatycznie

***Sieć neuronowa***

Sieć neuronowa nie jest wcale tak nowym wynalazkiem, jak może się zdawać. Pierwszą sieć neuronową opracował już w 1943 roku Warren McCulloch i Walter Pitts. Głównym konceptem sieci neuronowej jest inspiracja neuronem biologicznym. Neuron otrzymuje informacje, przekształca ją i zwraca ją do kolejnych neuronów. Działając tak tworzy się cała sieć neuronowa. Jej reprezentantem w świecie biologicznym jest mózg, który interpretuje informacje i zwraca odpowiedzi.

Obraz zawierający tekst, kwiat

Opis wygenerowany automatycznie

W świecie cyfrowym neuron jest swego rodzaju równaniem posiadającym parametry:

1. **Waga** (*ang.weight*) - określa siłę znaczenia (wagę) danego połączenia w neuronie.
2. **Obciążenie/przesunięcie** (*ang.bias*) *-* wartość dodawana do wyniku sumy ważonej w procesie obliczania wyjścia neuronu. Celem jest umożliwienie modelowi przesunięcia funkcji aktywacji, co pozwala lepiej dopasować model do danych.

odpowiednikiem mózgu jest komputer, który dostając dane wejściowe, zinterpretuje je i poda nam dane wyjściowe.

Taki neuron tworzy następnie warstwy neuronów. Możemy je podzielić w sposób podstawowy na warstwę wejścia i wyjścia. Jeśli warstw jest więcej pojawiają się warstwy ukryte.

Obraz zawierający tekst, diagram, linia, design

Opis wygenerowany automatycznie

***Funkcja aktywacji***

**Funkcja aktywacji** to kluczowy element w sztucznych sieciach neuronowych, który decyduje o tym, czy i w jaki sposób neuron "aktywuje się" (przekazuje sygnał) na podstawie swojej wejściowej sumy ważonej. Funkcja aktywacji wprowadza nieliniowość do modelu. Ponadto niektóre z tych funkcji dokonują normalizacji wyjścia, co oznacza przekształcenie wartości wyjściowej w określony zakres. Pomaga to w stabilnym trenowaniu sieci. Przykładami takich funkcji mogą być: sigmoid, czy tanh – tangens hiperboliczny.

Najpopularniejszą funkcją aktywacji jest funkcja **ReLU** (*ang. Rectified Linear Unit*).

W tej funkcji wartości ujemne zmieniane są na 0, a dodatnie pozostają bez zmian, jest ona szybka w obliczeniach, jednakże może prowadzić do sytuacji gdzie neurony nie będą się uczyć, gdyż wartość osiągną wartość 0. Wcześniejsze informacje będą zapominane. Wartość 0 zawsze jest problematyczna w uczeniu sieci neuronowych, gdyż może prowadzić do „zapominania”, utraty wcześniejszych informacji, przez co uczenie jest nieefektywne.

Obraz zawierający linia, diagram, Wykres, tekst

Opis wygenerowany automatycznie

Jeśli chcemy, aby nasz neuron podawał informacji interpretacji informacji w skali od 0 do 1 możemy skorzystać z funkcji **sigmoid.**

Jej wyniki można interpretować np. w ten sposób, że wyniki bliżej 0 oznaczają proporcjonalnie odpowiedź negatywną, a wyniki blisko 1 odpowiedź proporcjonalnie pozytywną. Należy uważać, gdy posiadamy skrajne wartości, ponieważ przez kształt funkcji mogą one zanikać.

Obraz zawierający linia, Wykres, diagram, Równolegle

Opis wygenerowany automatycznie

**Tanh**, czyli tangens hiperboliczny, swym kształtem przypomina funkcję sigmoid, lecz zbiór wartości na którym operuje wynosi (-1,1).

Jest ona lepsza dla wartości wycentrowanych wokół zera dlatego np. w metodzie LSTM, będzie to funkcja przekształcająca dane po działaniu funkcji sigmoid. Podobnie jak on ma również problem z zanikającymi wartościami skrajnymi.

Obraz zawierający linia, Wykres, diagram, Równolegle

Opis wygenerowany automatycznie

***Propagacja wsteczna***

**Propagacja wsteczna** (*ang.backpropagation*) to kluczowy algorytm stosowany w trenowaniu sztucznych sieci neuronowych. Pozwala modelowi uczyć się poprzez dostosowywanie wag i biasów w celu minimalizacji błędu predykcji. Działa poprzez obliczanie gradientów **funkcji kosztu** względem parametrów modelu i aktualizowanie ich w kierunku zmniejszania tego błędu.

**Funkcja kosztu** (*ang. cost function*), zwana również funkcją straty (*ang.* *loss function*), mierzy, jak dobrze model przewiduje oczekiwane wyniki. Określa różnicę między przewidywaniami modelu, a rzeczywistymi wartościami z danych treningowych. Jej minimalizacja jest kluczowym celem podczas trenowania modelu. To ona wskazuje zazwyczaj czy model uczy się dobrze, czy zachodzą jakieś trudności. Jedną z najpopularniejszych funkcji kosztu jest MSE - błąd średniokwadratowy (*ang. mean squared error*).

– wartości rzeczywiste,

– wartości przewidywane,

– ilość elementów

Można zauważyć, że ponieważ podnosimy różnice do kwadratu, to większe wartości błędów będą bardziej zauważalne niż mniejsze. Jesteśmy w stanie zredukować to zjawisko korzystając z MAE – średniego błędu bezwzględnego (*ang. mean absolute error*).

Istnieją oczywiście wiele więcej innych funkcji straty.

Etapami działania propagacji wstecznej są:

1. Propagacja do przodu - Dane wejściowe przechodzą przez sieć, generując wyjścia na podstawie wag, obciążeń i funkcji aktywacji, a następnie obliczana jest wartość funkcji kosztu.
2. Propagacja wsteczna - Gradient funkcji kosztu jest obliczany w odwrotnym kierunku, od warstwy wyjściowej do wejściowej, używając reguły łańcuchowej pochodnych.

Gradienty są obliczane dla każdego parametru (wag i biasów) w każdej warstwie, co pozwala określić, w jakim stopniu dany parametr wpływa na błąd. Wagi i obciążenia są aktualizowane zgodnie z regułą gradientu prostego (*ang.gradient descent*), zmniejszając błąd modelu poprzez przesunięcie parametrów w kierunku przeciwnym do gradientu. Propagacja wsteczna jest efektywna, uniwersalna i działa w różnych typach sieci neuronowych, takich jak sieci konwolucyjne (CNN) czy rekurencyjne (RNN). Algorytm może cierpieć na problem zanikającego gradientu w bardzo głębokich sieciach oraz ryzyko przeuczenia modelu, jeśli jest zbyt złożony lub źle regulowany.

**BPTT (Backpropagation Through Time):** Jest to szczególny przypadek propagacji wstecznej, który jest stosowany w sieciach rekurencyjnych (RNN), gdzie model ma pamięć i przetwarza dane sekwencyjne. BPTT rozszerza propagację wsteczną na dane, które mają strukturę czasową, umożliwiając propagowanie błędów przez wiele kroków czasowych. W sieciach rekurencyjnych, BPTT oblicza gradienty nie tylko dla wag, ale również dla "ukrytych stanów" w każdym kroku czasowym. Błąd jest propagowany zarówno wstecz przez warstwy, jak i przez czas, co oznacza, że gradienty muszą być obliczane na podstawie całej sekwencji danych. Ten rodzaj propagacji wstecznej będzie wykorzystywany w mojej pracy, w części praktycznej – predykcji za pomocą LSTM.

## 4.1 Model RNN

Rekurencyjne sieci neuronowe (RNN, ang. Recurrent Neural Networks) to sieci neuronowe zaprojektowane w ten sposób, aby przetwarzać dane sekwencyjne. Ich charakterystycznym elementem są komórki pamięci. Odróżnia je to od tradycyjnych sieci neuronowych. RNN potrafią „pamiętać” wcześniejsze stany, co jest kluczowe w dalszym podejmowaniu decyzji, np.: jak w przetwarzaniu języka naturalnego, rozpoznawaniu mowy, analizie wideo, czy analizie szeregów czasowych.

Struktura RNN umożliwia rekurencję, co oznacza, że wynik jednej operacji jest wykorzystywany jak wejście dla kolejnej.

Każdy krok czasowy w RNN można opisać jako:

W tym wzorze:

– macierz zawierająca dane wyjściowe w takcie t dla poszczególnego elementu, gdzie m to ilość próbek, a n liczba neuronów

- macierz zawierająca dane wejściowe dla wszystkich przykładów, gdzie n to liczba wejść

– macierz wag dla wejść w danym t

- wag dla danych wyjść z poprzedniego taktu

– wektor o wymiarze będący obciążeniem (*ang. bias*) dla każdego neuronu

W takcie t=0 nie istnieją żadne dane wyjściowe, więc zakłada się, że ich wartość równa jest 0.

Neuron RNN zachowuje pamięć o wartości stanu w poszczególnych taktach w **komórkach pamięci**. Jeden neuron lub warstwa RNN zachowuje w pamięci tylko krótkie sekwencje. Funkcja (*ang. hidden*) określa dane wejściowe w aktualnym kroku czasowym oraz stan z poprzedniego, t.zn.:

Obraz zawierający tekst, diagram, Czcionka, szkic

Opis wygenerowany automatycznie

Neuron można wyobrazić sobie jako zapętlony sam w sobie, czyli działający rekurencyjnie. Można to też przedstawić w postaci rozwijającego się neuronu na kolejne neurony. Ta wizualizacja jest bardziej przejrzysta dla dalszych rozumowań.

Obraz zawierający diagram, tekst, szkic, linia

Opis wygenerowany automatycznie

***Rodzaje sieci RNN***

Modele RNN możemy podzielić na różne rodzaje sieci:

1. Sieć sekwencyjna (*ang. sequence-to-sequence network*)

Dostaje informację i dokonuje predykcji na ciąg kolejnych taktów, np.: podając cenę produktu z N ostatnich dni przewiduje jakie ceny będzie miał w kolejnych dniach (od N-1 do jutra).

1. Sieć sekwencyjno-wektorowa (*ang. sequence-to-vector network*)

Otrzymuje dane i zbierając ich informacje może dac końcowy wynik, np.: zbierając sekwencję zdań można określić , czy są one pozytywne czy negatywne.

1. Sieć wektorowo sekwencyjna (*ang. vector-to-sequence network*)

Stale jest jej przekazywana ta sama informacja na postawie której otrzymujemy wynik, np.: przedstawiamy sygnał dźwiękowy i dostajemy jego opis.

1. Koder-dekoder

Bardziej rozbudowana sieć przetwarzająca część danych na podstawie poprzednich. Stosowana między innymi do przetwarzania języka naturalnego.

Tutaj można wstawić wizualizację opisanych sieci.

Dla modelu RNN, charakterystyczne jest stosowanie BPTT (*ang. backpropagation through time*) – propagacji wstecznej w czasie, zamiast jej klasycznego odpowiednika.

Najpierw dokonywany jest przebieg do przodu poprzez sieć. Następnie sekwencja jest oceniana poprzez funkcję straty. Czy mogę tak zapisać ten wzór?

gdzie:

– wartości znane z zestawu danych,

– wartości przewidywane,

– maksymalny takt.

Po zakończeniu BPTT przeprowadza etap gradientu prostego w celu zaktualizowania parametrów.

***Problem pamięci krótkotrwałej***

Dokonując pierwszych obliczeń przy dłuższych sekwencjach można zauważyć, że im dłuższa sekwencja tym mniej dokładne wyniki, co wydaje się sprzeczne z ideą sieci neuronowych, gdyż teoretycznie ilość danych do nauki zwiększa się. Powiązane może to być z problemem zanikającego gradientu (*ang. vanishing gradient*)lub wybuchającego gradientu (*ang. exploding gradient*). Problem ten pojawia się na etapie propagacji wstecznej BPTT.

***Zanikający gradient***

Gdy nasza wartość w będzie z zakresu (-1,1), będzie dochodzić do wielokrotnego jej mnożenia, co przyczyni się do zaniku pamięci wcześniejszych informacji w sekwencji i będą zapamiętywane tylko ostatnie elementy. Im dłuższa będzie sekwencja tym pamięć będzie bardziej „ulotna”, gdyż wartość będzie dążyć do zera, co jest z tożsame z „zapominaniem” informacji. Taką sytuację możemy zwizualizować na poniższym przykładzie.

Obraz zawierający diagram, szkic, tekst, rysowanie

Opis wygenerowany automatycznie

Jak można zauważyć pierwsza wartość wejściowa pomnożona wielokrotnie przez wartość w dużej ilości sekwencji praktycznie zniknie i analogicznie każda następna w proporcjonalnie mniejszym stopniu.

***Wybuchający gradient***

Odwrotne zachowanie wykaże model, gdy będzie z zakresu   
. Będzie wtedy dochodzić do wielokrotnego jej mnożenia, co przyczyni się do niestabilności pamięci. Wartość wag stanie się bardzo duża co będzie prowadzić do błędów numerycznych. Funkcja kosztu zacznie gwałtownie rosnąć co uniemożliwi dalsze uczenie, a czym dłuższa będzie sekwencja tym bardziej wyniki będą odbiegały od prawdy, gdyż wartość będzie dążyć do nieskończoności. Taką sytuację możemy zwizualizować na poniższym przykładzie.

Obraz zawierający diagram, tekst, Czcionka, linia

Opis wygenerowany automatycznie

Jak można zauważyć pierwsza wartość wejściowa pomnożona wielokrotnie przez wartość w dużej ilości sekwencji da dalece błędny wynik i analogicznie każda następna w proporcjonalnie mniejszym stopniu. Wszystkie wartości będą odbiegać od rzeczywistości, a wraz z coraz większą liczbą obliczeń predykcja będzie coraz bardziej błędna.

Ze względu na wyżej wymienione ograniczenia, RNN nie jest odpowiednim modelem do zadań wymagających przetwarzania skomplikowanych lub długich zależności czasowych. Dlatego w mojej pracy zdecydowałam się na wykorzystanie modelu LSTM (Long Short-Term Memory), który stanowi ulepszoną wersję RNN. LSTM, dzięki swojej architekturze opierającej się na komórkach pamięci i mechanizmie bramek, skutecznie radzi sobie zarówno z problemem zanikającego, jak i wybuchającego gradientu, co pozwala mu modelować długoterminowe zależności w danych.

LSTM nie tylko eliminuje główne wady klasycznych RNN, ale także cechuje się większą stabilnością i skutecznością w nauce. Dlatego model LSTM będzie kluczowym narzędziem w realizacji celów opisanych w mojej pracy.

Jeśli pojawi się coś nowego z RNN dopisać tutaj!

## 4.2 Model LSTM

Jednym z bardziej rozbudowanych rekurencyjnych sieci neuronowych jest model LSTM (*ang. long-short term memory*). Jest on odpowiedzią na problem zanikającego i wybuchającego gradientu, zaprojektowany tak, aby efektywnie gospodarować „pamięcią”. Struktura LSTM jest o wiele bardziej skomplikowana, ale dzięki temu model ten może swobodnie działać na długich sekwencjach danych.

***Jednostka LSTM – struktura***

Podstawowym założeniem jednostki LSTM jest utrzymanie informacji o stanach wcześniejszych. Do tego celu struktura modelu posiada dwa wektory przekazywania informacji. Na rysunku X.X są one kolejno zaznaczone kolorami: zielony: stan ukryty (*ang.hidden*) - długotrwały i czerwony – stan komórkowy (*ang.cell*)– krótkotrwały.

Stan został pozbawiony niewygodnego mnożenia wag, dzięki czemu problem zanikającego i wybuchającego gradientu został usunięty. Stan posiada wagi dla każdej kolejnej bramki czemu w wygodny sposób będzie je modyfikował, nie tracąc informacji.

Śledząc drogę jaką pokonuje informacja można zauważyć, że stan napotyka **bramkę zapominającą.** *(ang. forget gate).* Oczywiście jej rola zapominania jest wykonywana, lecz co ciekawe ma ona też wpływ na zapamiętywanie poprzez dodanie nowych „wspomnień” za pomocą operacji sumowania, a następnie mnożenia. Funkcja aktywacji, która jest w niej zazwyczaj używana, sigmoid, pozwala na utrwalenie ile pamięci zostanie zachowane, np. jeśli wartości niewiele się wahają to zmiany będą widoczne, gdyż będą pomnożone przez ułamek, jeśli wynik będzie skrajny to albo usunie wartości w pamięci będący bliski zeru, lub zostawi wynik prawie bez zmian gdyż dokona się mnożenie przez 1.

Obraz zawierający tekst, zrzut ekranu, Czcionka, diagram

Opis wygenerowany automatycznie

**Bramka wyjściowa** *(ang. input gate)* decyduje, na podstawie bieżącego wejścia i poprzedniego stanu ukrytego, jak dużo informacji z komórki pamięci ma zostać ujawnione na wyjściu*.* Cześć odpowiadająca za zapamiętanie w jakim procencie nowy potencjalny stan długotrwały ma zostać zachowany, korzysta z funkcji aktywacji - sigmoid, lecz cześć odpowiadająca za zapamiętanie w jaka wartość nowego potencjalnego stanu długotrwałego ma zostać zachowany, korzysta z funkcji aktywacji tangensa hiperbolicznego. Funkcja ta może przekazać wartości z zakresu [-1, 1]. W ten sposób otrzymujemy nowy stan . Korzystanie z innej funkcji, niż sigmoid może również wskazywać na inny charakter tego elementu. Można to interpretować, że jest to główna warstwa generująca pamięć długotrwałą, a więc tą która w większym stopniu wpływa na wynik końcowy i jest kluczowa w strukturze całego modelu. Pozostałe warstwy nazywamy **kontrolerami bramek** *(ang. gate controllers)* i wpływają one na to ile danej wartości w pamięci zostanie zachowane.

Obraz zawierający tekst, zrzut ekranu, diagram, Równolegle

Opis wygenerowany automatycznie

Uzyskany nowy stan długotrwały wykorzystamy ponownie w **bramce wyjściowej** *(ang. output gate)*. Nowe informację przechodzą przez funkcję tangensa hiperbolicznego, po czym zachodzi decyzja jaką wartość nowego, potencjalnego stanu krótkotrwałego zapamiętać. Stary stan krótkotrwały natomiast będzie jednym z czynników, który z pomocą funkcji sigmoid wyznaczy jaki procent nowego, potencjalnego stanu krótkotrwałego należy zapamiętać.

Obraz zawierający tekst, zrzut ekranu, diagram, róż

Opis wygenerowany automatycznie

Obraz zawierający tekst, diagram, zrzut ekranu, Plan

Opis wygenerowany automatycznieW ten sposób otrzymujemy kolejne stany długotrwałe i krótkotrwałe. Wszystkie elementy opisane powyżej zilustrowane są na poniższym rysunku w celu dokładniejszego zrozumienia zachodzących procesów. Komórki LSTM są niezwykle skuteczne i wychwytują długoterminowe wzorce.

Każdą z dróg którą podąża informacja można interpretować jako wektor/równanie. Dzięki temu teoretyczne rozważania można przekuć w realne matematyczne obliczenia. Na rysunku zaznaczone zostały wagi i obciążenia zgodnie z prezentowanymi poniżej równaniami.

- macierze wag dla wszystkich czterech warstw dla ich połączenia   
z – wektorem wejściowym.

- macierze wag dla wszystkich czterech warstw dla ich połączenia

z – stanem krótkotrwałym.

- obciążenia dla wszystkich czterech warstw.

|  |  |
| --- | --- |
| POLITECHNIKA RZESZOWSKA im. I. Łukasiewicza | Rzeszów, Rok |
| Wydział Elektrotechniki i Informatyki |  |
|  |  |

**STRESZCZENIE (wybierz rodzaj pracy)**

**TYTUŁ PRACY**

Autor: Imię Nazwisko, nr albumu: (wybierz symbol studiów)-123456 Opiekun: (tytuł naukowy przed) Imię Nazwisko(tytuł naukowy po)

Słowa kluczowe: (max. 5 słów kluczowych w 2 wierszach, oddzielanych przecinkami)

(tekst streszczenia - max. 10 wierszy)

RZESZOW UNIVERSITY OF TECHNOLOGY Rzeszow, Rok

Faculty of Electrical and Computer Engineering

**DIPLOMA THESIS (wybierz rodzaj pracy) ABSTRACT**

**TYTUŁ PRACY W WERSJI ANGIELSKIEJ**

Author: Imię Nazwisko, code: (wybierz symbol studiów) -123456 Supervisor: (tytuł naukowy przed) Imię Nazwisko (tytuł naukowy po)

Key words: (max. 5 słów kluczowych w 2 wierszach, oddzielanych przecinkami)

(tekst streszczenia w jęz. angielskim - max. 10 wierszy)