Politechnika
Śląska

PROJEKT INŻYNIERSKI

Modelowanie i analiza procesów związanych z działaniem systemu centralnego ogrzewania oraz przygotowania wody użytkowej w budynku mieszkalnym z wykorzystaniem uczenia maszynowego

Jakub KULA
Nr albumu: 296849

Kierunek: Automatyka i Robotyka
Specjalność: Technologie Informacyjne

PROWADZĄCY PRACĘ
dr inż. Szymon Ogonowski, prof. PŚ
KATEDRA Katedry Pomiarów i Systemów Sterowania
Wydział Automatyki, Elektroniki i Informatyki

Gliwice 2024

Tytuł pracy

Modelowanie i analiza procesów związanych z działaniem systemu centralnego ogrzewania oraz przygotowania wody użytkowej w budynku mieszkalnym z wykorzystaniem uczenia maszynowego

Streszczenie

(Streszczenie pracy – odpowiednie pole w systemie APD powinno zawierać kopię tego streszczenia.)

Słowa kluczowe

(2-5 slow (fraz) kluczowych, oddzielonych przecinkami)

Thesis title

Modeling and analysis of the heating and hot water preparation processes in residential building using machine learning

Abstract

(Thesis abstract – to be copied into an appropriate field during an electronic submission – in English.)

Key words

(2-5 keywords, separated by commas)

Spis treści

Rozdział 1

Wstęp

1.1 Cel i zakre pracy

wprowadzenie w problem/zagadnienie

1.2 Aktualny stan wiedzy

1.2.1 Data-Collected models

Modele oparte na zebranych danych są szeroką kategorią modeli statystycznych i analitycznych, które wykorzystują dane zebrane z różnych źródeł do formułowania wniosków, Edwards et al.[11] stworzyli roczne profile poboru wody, na podstawie danych pomiarowych z 73 domostw w Québec(Kanada). Utworzone profile poboru ciepłej wody użytkowej mają trzy kluczowe elementy: Wysoka rozdzielczość czasowa. Cztery poziomy zużycia - odpowidały one średniej, medianie, a także procentylom. Trzy wzorce czasowe - koncentowały sie one na porannym, wieczornym lub równomiernym zużyciu wody przez cały dzień, aby uwzględnić różne style życia i zwyczaje mieszkańców.

Modele szeregów czasowych są specjalną kategorią modeli statystycznych przeznaczonych do analizowania i prognozowania danych, które są zebrane lub obserwowane w regularnych odstępach czasu. Gelazanskas i Gamage[6] zauważywszy silną dzienną i tygodniową korealcje. Zastosowali kobinacje metode dekompozycji sezonowej oraz model ARIMA. Po porównianiu zaproponowanego modelu wraz z wybranymi modelami referencyjnymi, stwierdzono, że rozdzielenie szeregów czasowych na kilka składowych odgrywa znaczącą rolę w dokładności prognoz - uwzględnienie sezonowych wzorców zużycia może znacząco poprawić jakość prognoz.

1.2.2 Model Behawioralny

Modele behawioralne, są to modele oparte na zachowaniu ludzi w określonych sytuacjach, biorą one pod uwagę różne czynniki takie jak motywację, preferencje czy wpływ społeczny. Są one zazwyczaj stosowane w dziedzinach naukowych takich jak psychologia czy ekonomia, ale również w inżynierii. Model behawioralne stworzony przez N.D. Pflugradt[10] symulował indywidualne zachowania w celu przewidywania zużycia energii i wody w gospodarstwie domowym. Model został stworzony w celu uniknięcia polegania na statystycznych rozkładach prawdopodobieństwa. Autor stwierdził, że przewidywania modelu są bardzo dobre dla gospodarstw zamieszkanych przez jedną osobę, gdyż największym problem modelu jest jego skalowalność, ze względu na konieczność dokonania dużej ilość założeń i trudności w gromadzeniu danych o zachowaniach wielu mieszkańców.

1.3 Charakterystyka rozdziałów

(...)

W rozdziale drugim, zatytułowanym "Zastosowane Narzędzia w Pracy", zostanie szczegółowo omówione wymagania dotyczących języka programowania oraz głównej biblioteki używanej do tworzenia sieci neuronowych. Najpierw zostaną rozpatrzone dwa główne języki, które są popularne w dziedzinie uczenia maszynowego i zostanie dokonany wybór, który lepiej odpowiada wymaganiom. Następnie koncentracja zostanie położona na wybór biblioteki do tworzenia sieci neuronowe. Ponownie zostaną przedstawione dwie popularne biblioteki, które zostaną przeanalizowane pod kątem funkcjonalności, wydajności i wsparcia społeczności. Następnie zostanie wybrana jedna, która najbardziej sprzyja realizacji założeń. W dalszej części zostaną przedstawione biblioteki pomocnicze, które są niezbędne do przetwarzania i analizy danych. Zostaną opisane najważniejsze funkcje i zastosowania tych narzędzi. Zostanie także poruszony wątek kompatybilności tych bibliotek.

W kolejnym rozdziale pracy zostanie przedstawiona metodologia projektowania modelu sieci neuronowej. Szczegółowo omówione zostaną hiperparametry oraz zasady działania sieci neuronowych, aby zapewnić solidne podstawy teoretyczne dla dalszej części badań. Następnie zostaną omówione dane wejściowe, które są niezbędne do stworzenia modelu prognozującego dzienne zużycie wody. Szczególna uwaga zostanie poświęcona sposobom przetwarzania tych danych, aby były one adekwatne do zastosowania w sieciach neuronowych. W dalszej części rozdziału zaprezentujemy cztery różne modele sieci neuronowych, które będą się różnić ilością wejść. Dokładnie zostaną omówione wyniki generowane przez każdy z modeli dla trzech różnych zestawów danych. Na zakończenie rozdziału zostanie zaproponowany sposób ulepszenia wybranego modelu sieci neuronowej. Ulepszenie to będzie polegało na wprowadzeniu dodatkowego parametru wykorzystującego dane historyczne.

W ramach rozdziału "Pogoda"niniejszej pracy dyplomowej dokonano prezentacji i analizy danych meteorologicznych, które posłużyły jako fundament dla procesu nauki dwóch modeli sieci neuronowych o zróżnicowanej złożoności architektonicznej. Następnie, w dalszej części rozdziału, skoncentrowano się na szczegółowym badaniu i ocenie ważności poszczególnych wejść dla obu modeli. Do tego celu wykorzystano trzy zaawansowane metody analityczne: Permutacyjną Ważność Cech, analizę wag wejściowych modelu oraz metodę LIME(Local Interpretable Model-agnostic Explanations). Każda z tych technik dostarcza istotnych informacji o znaczeniu poszczególnych zmiennych wejściowych w kontekście ich wpływu na wydajność i dokładność modeli.

Rozdział 2

Zastosowane narzędzia w pracy

2.1 Python

Wybór głównego języka programowania zastosowanego w projekcie, wiązał się z postawieniem pewnych wymagań. Pierwszym z tych wymagań była dostępność dedykowanej biblioteki do uczenia maszynowego, która posiada narzędzia do efektywnej pracy nad modelami czy ich testowanie. Użycie biblioteki która jest dobrze utrzymana zapewnia ogromne wsparcie społeczności, które może okazać się nieocenione w procesie nauki czy rozwiązywania problemów.
Kolejnym wymaganiem jest aby wybrana technologia była ciągle wspierana i aktualizowana. Machine learning jest aktualnie jedną z najszybciej rozwijających się dziedzin programowania, co wiąże się z szybkimi zmianami. Po dokładnej analizie postawionych wymagań, zdecydowano o wyborze dwóch głównych języków programowania, które najlepiej odpowiadają potrzebom projektu

- R
- Python

R jest językiem skoncentrowanym na analizie danych i statystyce. Posiada on bardzo bogaty ekosystem jednak może stanowić to przyczynę wielu konfliktów pomiędzy pakietami. Największą wadą tego jest języka jest problem ze skalowalnością. Praca z dużą ilością danych skutkuje zużyciem ogromnej ilości pamięci RAM.
Skorzystanie z Pythona będzie lepiej spełniać wymogi projektu. Jest on językiem bardziej wszechstronny oraz posiada obszerną biblioteke standardową jak i bardzo liczne zewnętrzne bibloteki. Największą wadą Pythona jest jego wydajność. Gdyż jest językiem interpretowanym, więc nie jest on kompilowany do kodu maszynowego przed jego uruchomieniem.

2.2 Tensorflow

Decyzja o wyborze odpowiedniej biblioteki do tworzenia sieci neuronowych jest kluczowym elementem każdego projektu związanego z uczeniem maszynowym i głębokim uczeniem. TensorFlow, będący jedną z dwóch głównych otwartych bibliotek do uczenia maszynowego w Pythonie, wyróżnia się swoją wszechstronnością i wsparciem od Google. Jego głównym konkurentem jest PyTorch, rozwijany przez Facebook, który zyskał popularność ze względu na swoją elastyczność i przyjazność dla użytkownika.

- Rozbudowana Dokumentacja: Jest niezbędne, aby biblioteka była wyposażona we wszechstronną i precyzyjnie zorganizowaną dokumentację, która wspiera proces edukacyjny oraz efektywne rozwiązywanie problemów technicznych. W kontekście tego wymogu TensorFlow wyróżnia się, dostarczając szczegółowe instrukcje i obszerną kolekcję przykładowych zastosowań, co stanowi bezcenny zasób zarówno dla osób rozpoczynających pracę z tą technologią, jak i dla programistów o zaawansowanym poziomie doświadczenia.

- Integracja z Innymi Narzędziami i Bibliotekami: Python oferuje szeroką gamę pomocniczych bibliotek, które dostarczają zróżnicowanych typów danych, co jest istotnym aspektem jego funkcjonalności, więc ważne aby biblioteka była z nimi kompatybilna. TensorFlow oferuje szerokie możliwości integracji, zwłaszcza z narzędziami Google Cloud[4], podczas gdy PyTorch jest bardziej ukierunkowany na modułowość i elastyczność w integracji z różnorodnymi ekosystemami.

- Łatwość Użycia: Intuicyjność i prostota użytkowania są niezbędne dla szybkiego prototypowania i eksperymentowania. PyTorch, ze swoim dynamicznym grafem obliczeń, jest często postrzegany jako bardziej dostępny dla nowych użytkowników oraz bardziej elastyczny w badaniach.

- Wydajność i Optymalizacja: Optymalna wydajność i możliwości skalowania są kluczowe w aplikacjach przemysłowych. TensorFlow, z lepszym wsparciem dla TPU[3] i rozbudowanymi opcjami treningu rozproszonego, często przewyższa PyTorch w dużych projektach i zastosowaniach o dużym obciążeniu.

- Obsługa Wersjonowania Modeli: Zdolność do zarządzania różnymi wersjami modeli jest istotna w długoterminowej pracy nad projektem. TensorFlow i PyTorch oferują różne podejścia do zarządzania i wersjonowania modeli, które należy rozważyć w kontekście specyficznych wymagań projektu.

Po dokładnej analizie i uwzględnieniu wszystkich kluczowych wymagań zdecydowano, że najlepszym wyborem dla niniejszego projektu jest TensorFlow. Jest on idealnym wyborem zapewniającym wysoką wydajność, ale również elastyczność i skalowalność

2.3 Narzędzia pomocnicze

Python odgrywa kluczową rolę w dziedzinie uczenia maszynowego i analizy danych, oferując zróżnicowany zestaw narzędzi, które ułatwiają i usprawniają kluczowe procesy, takie jak przetwarzanie danych, ich testowanie oraz wizualizacja. Korzystając z nich, możliwe jest znaczne przyspieszenie procesów związanych z uczeniem maszynowym, od przygotowania danych aż po analizę wyników, co podkreśla ich niezastąpioną wartość w projektach naukowych i przemysłowych.

2.3.1 Pandas i Numpy

NumPy, znany również jako Numerical Python, jest fundamentalną biblioteką do obliczeń naukowych w Pythonie. Oferuje ona wsparcie dla wielowymiarowych, jednorodnych tablic, które w znaczący sposób przewyższają standardowe listy Pythona pod względem wydajności, szczególnie przy obróbce dużych zbiorów danych. NumPy znacząco upraszcza operacje na danych dzięki wektoryzacji, umożliwiając wykonanie operacji takich jak dodawanie, mnożenie czy transpozycja tablic bez konieczności stosowania pętli i iteracji

Biblioteka Pandas jest jedną z najbardziej popularnych i użytecznych narzędzi w języku Python przeznaczonych do analizy danych. Kluczowym elementem, który Pandas oferuje swoim użytkownikom, jest struktura danych znana jako DataFrame. Jest to dwuwymiarowa, etykietowana tablica, umożliwiająca efektywną manipulację i analizę złożonych zbiorów danych. Dodatkowo biblioteka Pandas wyróżnia się również zoptymalizowaną wydajnością. Jest szczególnie skuteczna w obsłudze dużych zestawów danych, co stanowi kluczowe znaczenie w pracach wymagających szybkiego i efektywnego przetwarzania znacznych ilości informacji. Pandas wyposażony jest w bogaty zestaw narzędzi przeznaczonych do przetwarzania i analizowania danych, co znacznie ułatwia pracę z nimi. Istotną cechą tej biblioteki jest również jej zdolność do łatwego wczytywania i zapisywania danych z i do różnych formatów, takich jak CSV czy Excel.

NumPy i Pandas to dwie biblioteki, które idealnie współdziałają w ekosystemie Pythona, szczególnie w kontekście analizy danych i uczenia maszynowego. Pandas, będąc biblioteką wysokiego poziomu do manipulacji i analizy danych, jest zbudowany na fundamencie NumPy, większość funkcji w niej dostępnych, w tym te dotyczące operacji matematycznych i statystycznych, bazuje na mechanizmach i strukturach NumPy. Ponadto, zarówno NumPy, jak i Pandas wykazują doskonałą kompatybilność z bibliotekami używanymi w uczeniu maszynowym, takimi jak TensorFlow.

2.3.2 Matlibplot

Matplotlib jest obecnie uznawana za najpopularniejszą bibliotekę do tworzenia wykresów i wizualizacji danych w języku Python. Ta wszechstronna biblioteka umożliwia tworzenie różnorodnych rodzajów wykresów, w tym wykresów liniowych, słupkowych, kołowych, a także map ciepła, które będą odgrywały kluczową rolę w jednym z badań przedstawionych w tej pracy. Matplotlib wyróżnia się także szerokimi możliwościami personalizacji wykresów, umożliwiając użytkownikom pełną kontrolę nad aspektami wizualnymi takimi jak style linii, kolory, a także możliwość dodawania adnotacji i etykiet, co znacznie zwiększa czytelność i efektywność prezentacji danych. Jedną z najważniejszych cech Matplotlib jest jej integracja z Pandas i NumPy, które są dwiema głównymi bibliotekami do przetwarzania i analizy danych używanymi w tej pracy.

2.3.3 Sckit-learn, ELI5 i LIME

Scikit-learn jest biblioteka zawierająca bogaty zbiór algorytmów uczenia maszynowego, nie tylko oferuje szeroki zakres technik uczenia, ale także zapewnia dostęp do rozmaitych narzędzi przetwarzania danych. Dodatkowo scikit-learn oferuje różnorodne metryki oceny ELI5 i Lime udostępniają nam narzędzia umożliwiające głębsze zrozumienie mechanizów decyzyjnych modelu. Dzięki nim jesteśmy w łatwy sposób zbadać każdą predykcję modelu jak i wpływ każdej zmiennej z osobna.

2.3.4 CUDA i Conda

CUDA, będąca narzędziem opracowanym przez firmę NVIDIA, stanowi kluczową technologię umożliwiającą wykorzystanie mocy obliczeniowej procesorów graficznych do przyspieszenia działania aplikacji. CUDA umożliwia znaczne przyspieszenie obliczeń w porównaniu do tradycyjnego przetwarzania na CPU, dzięki tej technologii czas potrzebny do trenowania zaawansowanych modeli uczenia maszynowego zostanie znacznie skrócony.

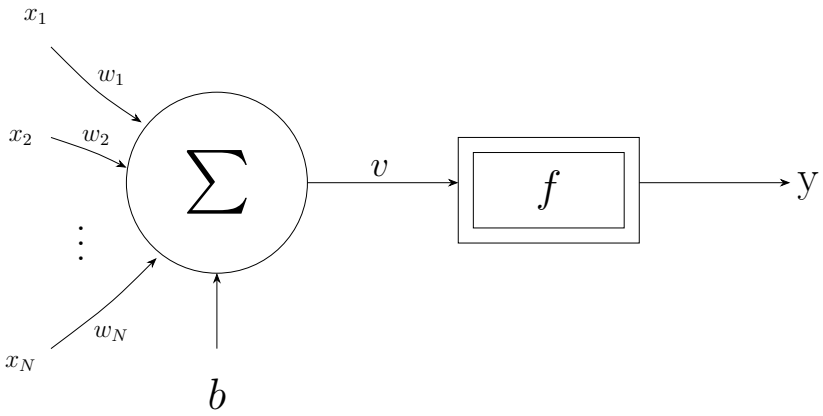
Conda jest środowiskiem wirtualnym i systemem do zarządzania pakietami. Pozwala ona na tworzenie odseparowanych środowisk dla różnych projektów, co jest bardzo ważne w momencie gdy różne projekty potrzebują różnych wersji tej samej biblioteki. Jest to szczególnie istotne w kontekście naszej pracy, gdzie narzędzia opisane wcześniej - mimo że idealnie współpracują ze sobą - wymagają specyficznych, niekoniecznie najnowszych wersji.

Rozdział 3

Modelowanie sieci neuronowej

3.1 Metodologia projektowania modelu sieci neuronowej

W początkowych etapach rozwoju sieci neuronowych, neuron, będący ich najmniejszym elementem był wzorowany na rzeczywistych neuronach ludzkiego układu nerwowego. Każdy taki neuron w sieci neuronowej funkcjonuje jako układ MISO, oznaczający wiele wejść i jedno wyjście. W praktyce oznacza to, że sygnał wyjściowy neuronu jest przekazywany do każdego neuronu w kolejnej warstwie sieci, jednak ponieważ jest to ta sama wartość, traktujemy to jako pojedyncze wyjście.



Rysunek 3.1: Model neuronu

Neuron w sieci neuronowej jest sumatorem. Sumuje on wartości wszystkich wejść, przy czym każde wejście jest mnożone przez określoną wagę oraz dodawana jest do nich wartość stała. Po zsumowaniu tych wartości sygnał wyjściowy neuronu jest przekształcany przez funkcję aktywacji, która decyduje o ostatecznej wartości wyjściowej tego neuronu.[5]

Wyjście pojedynczego neuronu można opisać za pomocą wzoru:

$$y = f\left(\sum_{j=1}^{N} w_j x_j + b\right) \tag{3.1}$$

gdzie:

- y - sygnał wyjściowy neuronu,
- x_j - sygnał wejściowy neuronu dla j-tego wejścia,
- w_j - waga przypisana do j-tego sygnału wejściowego neuronu,
- b - składnik stały,
- $f(\cdot)$ - funkcja aktywacji neuronu,
- N - liczba wejść neuronu.

Neurony organizują się w sieci, tworząc struktury znane jako sieci neuronowe. Każda sieć neuronowa składa się z wielu neuronów, które są wzajemnie połączone i mogą być zorganizowane w różne warstwy. Charakterystyczną cechą neuronów należących do tej samej warstwy jest to, że otrzymują one wspólny sygnał wejściowy.

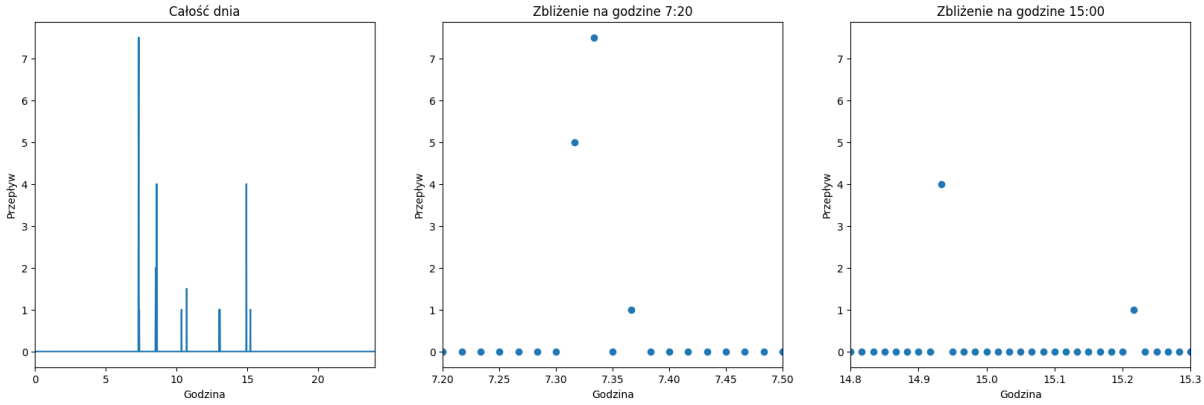
Proces tworzenia architektury sieci neuronowej jest procesem wyboru hiperparametrów, czyli wartości konfiguracyjnych, które określają sposób działania i uczenia się modelu. Hiperparametry w sieciach neuronowych można podzielić na dwie kategorie:

1. Hiperparametry Architektury Sieci:
   (a) Liczba warstw
   (b) Liczba neuronów
   (c) Typy warstw
2. Hiperparametry Procesu Uczenia:
   (a) Szybkość uczenia się: Wielkość kroku, o jaki wagi sa aktualizowane podczas uczenia.
   (b) Liczba epok: Ilośc jaką zestaw uczący jest przetwarzany przez sieć.
   (c) Rozmiar partii: Liczba próbek danych, które są przetwarzane jednocześnie w jednym kroku uczenia.
   (d) Optymalizator: Algorytm służący do aktualizacji wag w sieci.
   (e) Funkcja straty: Metryka oceniająca, jak dobrze model radzi sobie z zadaniem.

Aby sieć neuronowa mogła efektywnie przewidywać wyniki, kluczowe jest jej odpowiednie nauczenie. Proces ten rozpoczyna się od inicjalizacji wag sieci, co stanowi punkt wyjściowy dla dalszego uczenia. Następnie, w trakcie procesu uczenia, wagi te są ciągle modyfikowane. Odbywa się to poprzez przesyłanie danych przez sieć i obliczanie błędu, który jest miarą różnicy między przewidywaniami sieci a rzeczywistymi wynikami. Kolejnym etapem jest obliczenie gradientu funkcji straty dla każdej wagi w sieci. Gradient ten wskazuje kierunek, w którym należy zmodyfikować wagi, aby zminimalizować błąd. Aktualizacja wag odbywa się przy pomocą wybranego algorytmu optymalizacyjnego zwanego optymalizatorem. Cały ten proces jest powtarzany przez ustaloną liczbę iteracji i epok, co pozwala sieci na stopniowe 'naukę' i poprawę swojej zdolności do przewidywania wyników.[9]

3.2 Dane wejściowe i proces ich przetwarzania

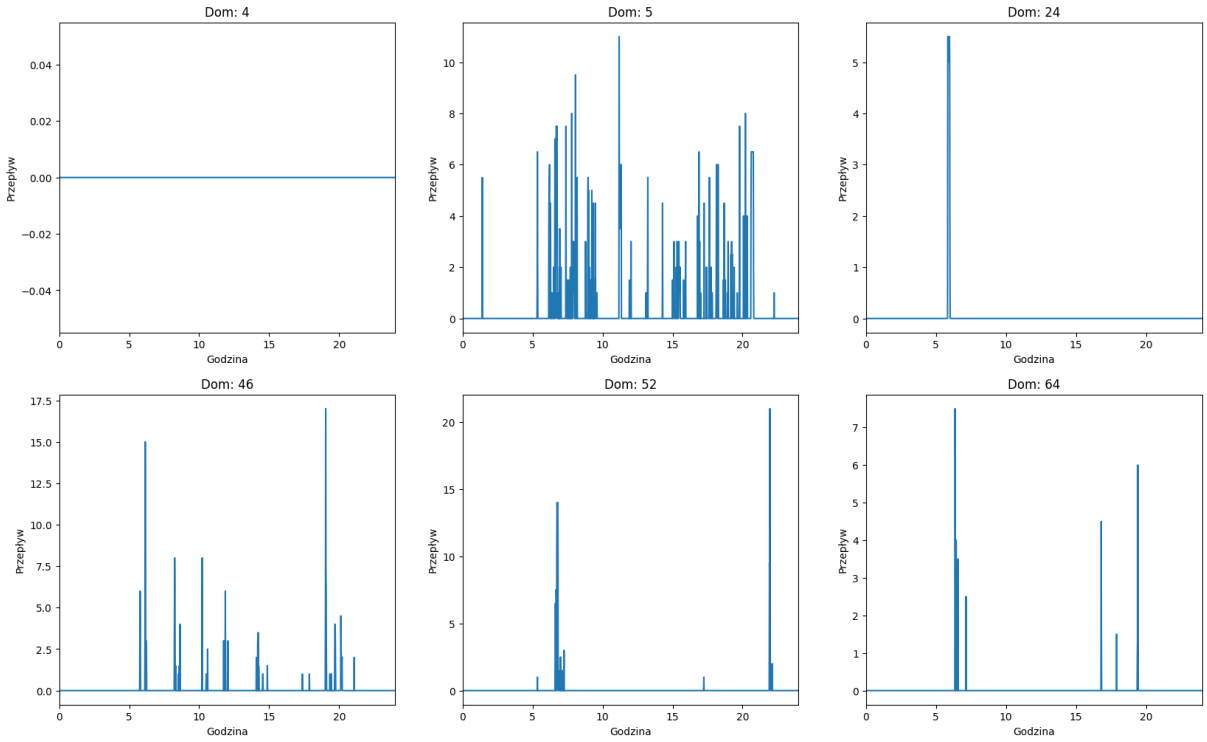
Projekt wykorzystuje dane zebrane przez M.J. Ritchie et al.[8] w 2018 roku. Informacje o zużyciu wody zostały pozyskane z 77 domostwo znajdujących się na terenie Południowej Afryki, zamieszkiwanym przez dwoje osób dorosłych oraz dwójkę dzieci, oraz posiadających podstawowe urządzenia AGD takie jak pralka czy zmywarka. Dane były zbierane przez 16 tygodni, w cyklach trwających 4 tygodnie, aby równomiernie obejmować każdą porę roku. Próbkowanie danych odbywało sie co minute, nieprzerwanie przez całą dobę. Całość danych zostało podzielone na 73 oddzielne pliki w formacie .csv. W każdym pliku znajdował się timestamp w formacie "RR-MM-DD HH:MM:SS", który określał dzień i godzinę pomiaru danych. Oprócz tego zawierał informacje o zużyciu wody, oraz temperaturę otoczenia. Dodatkowo dane były podzielone na poszczególne poru roku.



Rysunek 3.2: Podpis rysunku zawsze pod rysunkiem.

Pierwszy wykres przestawia dane zebrane z domu nr. 13 zebrane w dniu 05/02/2018. Oś X oznaczająca godzinę, począwszy od północy. Oś Y reprezentuje przepływ wody w danym momencie dnia. Wykres przedstwia nieregularne piki o nierównomiernym rozkładzie. Okresami o zwiększonym przepływie są godziny 7-13 oraz 14-15. Okresy mniejszej aktywności możemy zaobserwować w godzinach późno popołudniowych oraz nocnych. Drugi i trzeci wykres przedstawiają zbliżenie na godzinę 7 oraz 15. Dzięki zwężeniu analizowanego zakresu czasu możliwe było dokładniejsze zbadanie struktury występujących pików. Ta metoda wizualizacji ujawniła, że poszczególne piki, które na ogólnym wykresie dobowym mogły sprawiać wrażenie pojedynczych punktów, w rzeczywistości są złożone z wielu pojedynczych zdarzeń. To odkrycie jest istotne, ponieważ wskazuje na bardziej złożoną dynamikę przepływu w określonych momentach doby, co na pierwszy rzut oka mogło umknąć uwadze.

W celu lepszego wstępnego zrozumienia charakterystyki analizowanego zestawu danych, niezbędne jest także szczegółowe przyjrzenie się kilku losowo wybranym domostwom.



Rysunek 3.3: Porównanie przepływów dla przykładowych domów w dniu 05/02/2018

Analiza przedstawionych histogramów przepływów dla poszczególnych losowo wybranych domostw, wykonana na podstawie danych z dnia 05/02/2018, ukazuje wyraźne różnice w charakterystyce przepływów dla każdego z nich. Zgodnie z Rysunkiem 3.2, można stwierdzić, że każdy dom prezentuje unikalny wzór aktywności, co może odzwierciedlać różnorodność nawyków, planów dnia lub specyficznych potrzeb mieszkańców. Na przykład, dla domu nr 5 zużycie wody jest rozłożone przez większą część dnia, z obserwowaną aktywnością w rozmaitych godzinach. Jednakże zarejestrowano również pojedyncze zużycie w nocy. Z kolei Dom 52 charakteryzuje się wyraźnym szczytem wieczornym, co stanowi kontrast w stosunku do pozostałych domów.
W przypadku Domu nr 4, nie odnotowano żadnego przepływu w analizowanym dniu. Brak danych może wynikać z co najmniej dwóch potencjalnych przyczyn. Pierwszą z nich jest zastosowany czas próbkowania, który wynosił jedną minutę. Taki interwał może nie być wystarczająco krótki, aby zarejestrować sporadyczne lub krótkotrwałe zdarzenia przepływu. Drugą możliwością, która może wyjaśniać brak zarejestrowanej aktywności, jest potencjalna nieobecność mieszkańców w domu w danym dniu.

Pomimo iż zgromadzone dane charakteryzowały się wysoką jakością, ich specyficzny format wymagał przygotowania skryptu celem ich przetwarzania i ekstrakcję istotnych informacji.

```
for f in csv_files:
    dataset = pd.read_csv(f)
    for i in range(len(dataset)):
        dt = dataset.loc[i, "Summer_Timestamps"]
        data, time = dt.split(' ')
        year, month, day = (int(x) for x in data.split('-'))
        ans = datetime.date(year, month, day)
        dzied_tygodnia = dni_tygodnia_mapa[ans.strftime("%A")]

        hours, minutes, null = time.split(":")
        time = (int(hours)*60+int(minutes))/(60*24)

        przeplyw = dataset.loc[i, "Summer_Water_Consumption"]

        dane.loc[len(dane)] = {'Pora_roku': 2,
                               'Dzien_tygodnia': dzied_tygodnia,
                               'Czas_dnia': time,
                               'Przeplyw': przeplyw}
```

Rysunek 3.4: Fragment skryptu przetwarzającego dane.

W ramach procesu dostowania formatu aby przystosować go do wymagań tensorflow dzień tygodnia został zamieniona na etykietę liczbową, która przyjmuje wartość od 1 do 7, co odpowiada kolejnym dniom tygodnia. Podobny proces został zastosowany do etykietowania pór roku. Każda została zakodowana jako etykieta w zakresie od 1 do 4 co prezentuje kolejno wiosnę, lato, jesień i zimę. Dodatkowo czas dnia został zmieniony na procent dnia w skali od 0 do 1.

3.3 Projektowanie i ocena modeli

W ramach realizacji badań nad optymalizacją architektury sieci neuronowej oraz doborem hiperparametrów zdecydowano się na podział danych uczących na trzy zbiory. Pierwszy z nich to zestaw który zawiera dane pochodzące z 12 losowo wybranych domostw, co ma na celu zapewnienie reprezentatywności i różnorodności w ramach próby badawczej. Drugi zestaw stanowi podzbiór zawierający dane z pojedynczego gospodarstwa domowego, co pozwala na szczegółową analizę wydajności modelu w warunkach bardziej jednorodnych danych. Dodatkowo, utworzony został trzeci zestaw danych, który obejmował informacje z wszystkich 77 domów biorących udział w badaniu.

Podział zbioru danych na trzy zestawy okazał się kluczowy dla efektywnego doboru hiperparametrów modelu, szczególnie biorąc pod uwagę, że cały zbiór danych zawierał aż 12,5 miliona wierszy. Zestaw wybranych domostw, zawierający blisko 2 miliony wierszy, oraz pojedyncze domostwo z 161 tysiącami wierszy, umożliwiły przeprowadzenie dokładniejszych i bardziej zróżnicowanych testów.

Czas uczenia sieci był znacząco różny dla poszczególnych zestawów danych. Przykładowo, dla całego zbioru danych proces uczenia trwający 10 epok przy rozmiarze partii równym 64 zajmował około 46 minut. Tymczasem dla wybranych domostw czas ten skracał się do 9 minut, a dla pojedynczego domostwa uczenie trwało zaledwie 40 sekund. W ramach badań podjęto próby wykorzystania Google Colab, będącego popularnym narzędziem służącym do programowania i przetwarzania danych w chmurze. Po odpowiednim skonfigurowaniu środowiska napotkano na pierwszy znaczący problem - czas trwania uploadu pliku. Zaskakująco, przesyłanie pliku o rozmiarze 300 MB, zawierającego 12,5 miliona wierszy, zajęło znacznie więcej czasu, niż można było przewidywać. Kolejnym krokiem było przeprowadzenie procesu uczenia maszynowego na danych, zaplanowanego na 10 epok. Niestety, cały proces trwał ponad 100 minut, co wskazuje na ograniczenia wersji darmowej Google Colab. W związku z tym stwierdzono, że bez inwestycji w wersję płatną, Google Colab nie zapewnia oczekiwanej redukcji czasu niezbędnego do nauki modelu

Początkowo hiperparametry były testowane na najmniejszym zbiorze, co pozwalało na szybką i efektywną ocenę różnych konfiguracji. Po uzyskaniu zadowalających wyników na zbiorze pojedynczego domostwa testy były rozszerzane kolejno na zbiór wybranych domostw, a następnie na pełny zbiór danych. Taka strategia pozwoliła na stopniowe i metodyczne dostosowywanie hiperparametrów, minimalizując przy tym czas i zasoby potrzebne do przeprowadzenia eksperymentów, a jednocześnie maksymalizując ogólną skuteczność modelu. Na podstawie serii przeprowadzonych testów dotyczących dostrajania modelu wybrano następujące hiperparametry

Tabela 3.1: Hiperparametry Sieci Neuronowej

Optymalizator	Funkcja strat	Początkowy współczynnik uczenia	Rozmiar partii
Adam	mse	0.0001	64

Po przeprowadzeniu serii eksperymentów, w procesie selekcji optymalnej architektury sieci neuronowej, najbardziej efektywną konfiguracją okazała się struktura składająca się z sześciu warstw, z których cztery pełniły funkcję warstw ukrytych. W procesie iteracyjnego dostosowywania i ewaluacji różnych architektur sieci, model o takiej budowie wykazał najlepsze wyniki w zakresie dokładności i generalizacji na testowanych zbiorach danych. Architektura ta charakteryzowała się kolejno malejącą liczbą neuronów w poszczególnych warstwach: pierwsza warstwa zawierała 512 neuronów, druga 256, trzecia

128, czwarta 64, piąta 32, a szósta, będąca warstwą wyjściową, miała 1 neuron. Wszystkie warstwy, z wyjątkiem ostatniej, wykorzystywały funkcję aktywacji ReLU. Natomiast ostatnia warstwa, pełniąca rolę warstwy wyjściowej, zastosowała funkcję aktywacji typu 'linear'

W ramach opracowanego modelu sieci neuronowej zastosowano dynamicznie zmieniający się współczynnik uczenia, oparty na metodzie wykładniczego spadku, opisanego wzorem:

$$\text{Współczynik uczenia}(epoka) = \begin{cases} \text{Początkowy współczynik uczenia} & \text{jeżeli } epoka < 5 \\ \text{Współczynik uczenia}(epoka - 1) \times e^{-0.1} & \text{jeżeli } epoka \geq 5 \end{cases} \tag{3.2}$$

Użycie tej motody pozwoliło na zmniejszanie wartości współczynnika uczenia w trakcie procesu trenowania, co zwiększyło zdolności adaptacyjne sieci. Został on zastosowany, gdyż częstym zjawiskiem było generowanie przez sieć stałej wartości wyjściowej, niezależnie od różnych danych wejściowych.

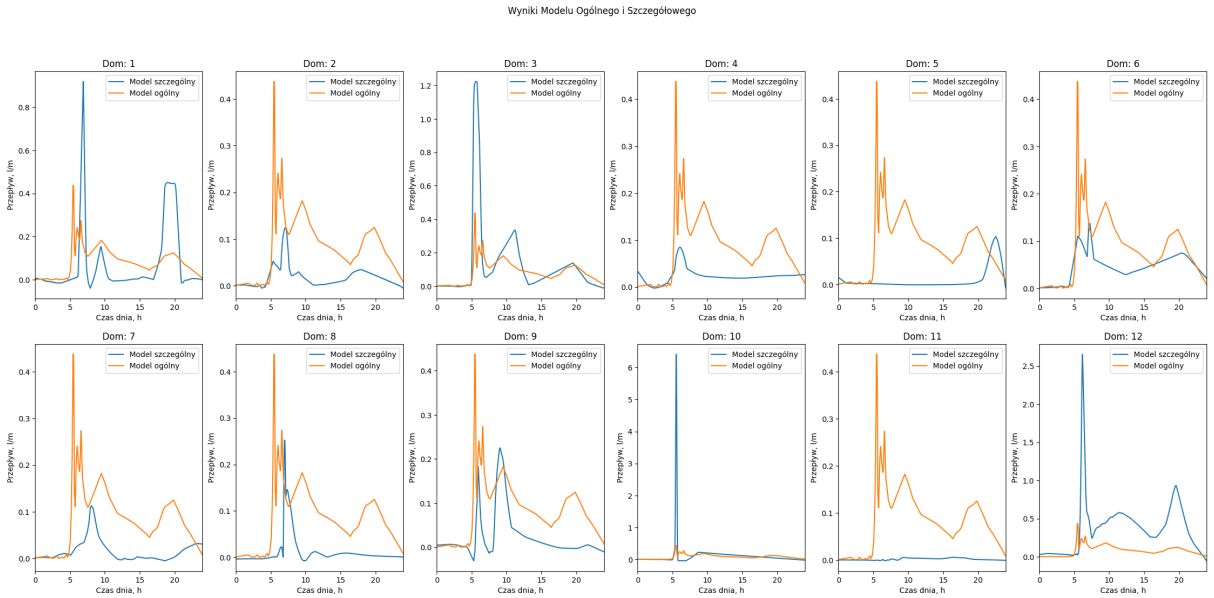
W ramach procesu testowania różnych konfiguracji sieci neuronowej zaproponowano eksplorację wydajności modeli przy różnorodnych kombinacjach wejść. Celem tego podejścia było zbadanie, jak zmiana danych wejściowych wpłynie na zdolność modelu do nauki i generalizacji przewidywania przepływu. Poniżej przedstawiono zestawienie modeli, które zostały uwzględnione w analizie:

1. Model A
   (a) Wejścia: Dzień tygodnia, pora dnia
   (b) Wyjście: Przepływ
2. Model B
   (a) Wejścia: pora dnia
   (b) Wyjście: Przepływ
3. Model C
   (a) Wejścia: pora roku, dzień tygodnia, pora dnia
   (b) Wyjście: Przepływ
4. Model D
   (a) Wejścia: pora roku, pora dnia
   (b) Wyjście: Przepływ

Wyniki te dostarczą wglądu w to, które wejścia są najbardziej wartościowe dla modelowania przepływu oraz czy dodanie dodatkowych informacji kontekstowych przyczynia się do znaczącej poprawy wyników predykcyjnych.

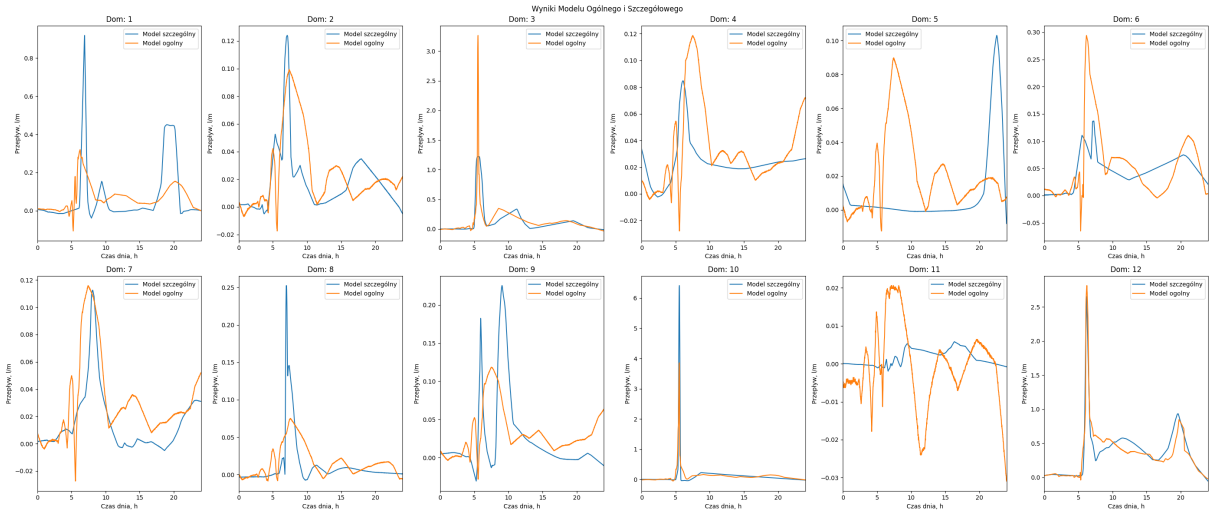
3.4 Walidacja i próby dostrajania (?)

W celu weryfikacji poprawności i efektywności opracowanego modelu sieci neuronowej przeprowadzono porównanie modelu nauczonych na danych ze wszystkich 12 domostw z modelami utworzonymi dla każdego z tych domów osobno. Taki eksperyment miał na celu ocenę zdolności generalizacji modelu nauczonych na zbiorze 12 domostw w porównaniu z modelami specyficznymi dla poszczególnych domów.



Rysunek 3.5: Porównianie modelu ogólnego z modelami szczególnymi

W celu dalszego zwiększenia skuteczności modelu sieci neuronowej zaproponowano wprowadzenie dodatkowego wejścia do systemu – tygodniowego zużycia. Implementacja tego rozwiązania została przeprowadzona w specyficzny sposób, mający na celu uniknięcie przekształcenia tego parametru w niezamierzony label identyfikujący poszczególne domy. W fazie uczenia modelu, do każdego tygodnia przypisywano sumę zużycia zarejestrowanego w tym okresie. Natomiast w fazie testowania, model otrzymywał średnią wartość tygodniowego zużycia. Celem tej strategii było umożliwienie modelowi korzystania z danych historycznych zużycia w sposób, który poprawiałby jego zdolność do przewidywania, jednocześnie zachowując elastyczność i możliwość generalizacji wyników na różne domostwa.



Rysunek 3.6: Porównianie modelu ogólnego z modelami szczególnymi po dodaniu kolejnego wejścia do sieci

Analizując charakterystykę modelu ogólnego i szczególnego dla każdego z domostw, przedstawioną na Rysunku 3.5, można zauważyć, że dla domostwa nr 10 i 12 model ogólny wykazał się wysoką zgodnością z modelem szczegółowym, co świadczy o jego zdolności do precyzyjnego odwzorowania charakterystyki przewidywania przepływu w ciągu dnia. W przypadku tych dwóch domów przebieg przewidywań dla obu modeli jest podobny, co wskazuje na to, że model ogólny efektywnie uchwycił dynamikę zużycia charakterystyczną dla tych konkretnych domostw. Natomiast w kontekście Domu nr 5, wyniki ukazują, że model ogólny miał znaczne trudności z dopasowaniem się do wzorców przepływu.

W celu dokładniejszej oceny i porównania efektywności modelu ogólnego, nauczonego na danych z 12 domostw, z modelami szczegółowymi, nauczonymi dla poszczególnych domów, zastosowano wskaźnik błędu średniokwadratowego. MSE, obliczany jako średnia kwadratów różnic między wartościami przewidywanymi przez model a rzeczywistymi danymi, posłużył jako miara odchylenia modelu ogólnego od wyników modeli szczegółowych. W tym kontekście niższa wartość MSE wskazywała na lepszą zgodność modelu ogólnego z wynikami modeli szczegółowych, sugerując, że model ogólny skuteczniej generalizuje dane, zbliżając się do precyzji modeli trenowanych na danych z pojedynczych domostw.

Tabela 3.2: Porównianie wartości MSE dla każdego obu modelów

MSE	DOMOSTWO											
	1	2	3	4	5	6	7	8	9	10	11	12
Model 1	0.022	0.0074	0.037	0.0067	0.011	0.0039	0.0084	0.0095	0.0072	0.22	0.010	0.19
Model 2	0.019	0.00053	0.034	0.00085	0.0013	0.0025	0.00055	0.00069	0.0027	0.075	0.00013	0.019

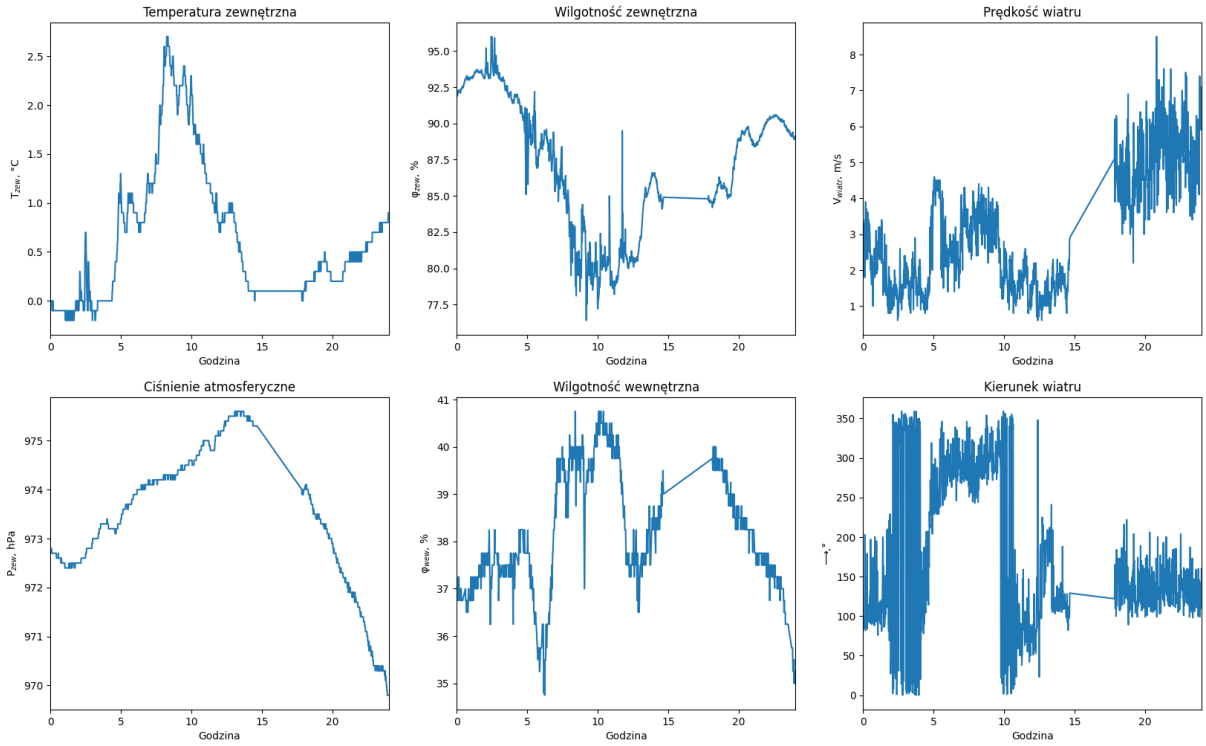
Analizując przedstawione dane w tabeli, można zauważyć, że po dodaniu dodatkowego wejścia do systemu, czyli tygodniowego zużycia, model 2 (Model z dodatkowym wejściem) osiągnął znacznie lepsze wyniki w porównaniu z modelem bazowym. Wartości błędu średniokwadratowego dla modelu 2 są niższe w porównaniu do modelu 1 dla każ-

dego z domostw, co wskazuje na poprawę dokładności predykcji. Na przykład, dla Domu nr 2, MSE zmniejszyło się z 0.0074 w modelu 1 do 0.00053 w modelu 2, co jest znaczącą poprawą. Podobne znaczące redukcje można zaobserwować w przypadku Domu nr 12, gdzie MSE spadło z 0.19 do 0.019

Rozdział 4

Pogoda

NAPISAĆ TUTAJ COŚ!



Rysunek 4.1: Porównanie warunkuch atmosferycznych na przestrzeni dnia

Rysunek przedstawia zestaw sześciu wykresów, które porównują warunki atmosferyczne w ciągu dnia. Każdy wykres zawiera oś X, która reprezentuje czas, oraz oś Y, która przedstawia różne zmienne meteorologiczne. W pierwszym wykresie prezentowane są zmiany temperatury zewnętrznej na przestrzeni dnia. Temperatura oscyluje w zakresie od około -0.5°C do 2.5°C, z maksymalnymi wartościami obserwowanymi w okolicach godziny ósmej. Charakterystyka tej zmienności temperatury sugeruje, że dane mogły być zbierane w okresie zimowym. Następny wykres ilustruje zmiany wilgotności powietrza

na zewnątrz budynku, wyrażone w procentach. Wysoki poziom wilgotności, utrzymujący się przez cały dzień, może wskazywać na sezon zimowy lub jesienny. Prędkość wiatru, prezentowana na kolejnym wykresie, wykazuje znaczący wzrost w ciągu dnia, osiągając maksymalne wartości przekraczające 8 $\frac{m}{s}$, co może być charakterystyczne dla warunków zimowych, kiedy wiatry są silniejsze. Wykres ciśnienia atmosferycznego pokazuje wzrost wartości w pierwszej połowie dnia, po czym następuje stopniowy spadek. Wilgotność wewnętrzna, prezentowana na piątym wykresie, wykazuje wahania z ogólnym trendem spadkowym w drugiej połowie dnia. Szósty wykres przedstawia kierunek wiatru, który jest zmienny w ciągu dnia, z wyraźnymi zmianami szczególnie w środkowej części dnia, co może świadczyć o niestabilnych warunkach atmosferycznych.

W celu zbadania wpływu wielkości modelu na wyniki oraz sprawdzenia ważności różnych wejść, zaproponowano w badaniu dwa modele o różnej złożoności architektury. Oba modele korzystały z tej samej funkcji harmonogramowania tempa uczenia, która redukowała szybkość uczenia po czwartej epoce, oraz z tych samych parametrów kompilacji, w tym optymalizatora Adam z początkową szybkością uczenia 0.001, funkcji straty MSE Pierwszy model składał się z mniejszej liczby warstw i neuronów: warstwa normalizująca, trzy warstwy gęste z odpowiednio 64, 32 i 1 neuronami, używając funkcji aktywacji 'relu' dla pierwszych dwóch warstw i 'linear' dla warstwy wyjściowej. Drugi model był znacznie większy, zawierając więcej warstw i neuronów: warstwa normalizująca, sześć warstw gęstych o zwiększającej się liczbie neuronów: 1024, 512, 256, 128, 64, 32, zakończonych warstwą wyjściową z 1 neuronem, używając funkcji aktywacji 'relu' dla warstw ukrytych i 'linear' dla warstwy wyjściowej. Oba modele były trenowane przez 50 epok z rozmiarem partii równym 32. Celem porównania tych dwóch modeli było ustalenie, czy zwiększenie liczby warstw i neuronów w modelu wpłynie na jego

W ramach procesu weryfikacji skuteczności zastosowanych modeli sieci neuronowych została zastosowana walidacja krzyżowa, metodą holdout. Dane zostały podzielone w proporcji 80% do 20% na zestaw treningowy i testowy. Ta strategia podziału danych miała na celu zapewnienie solidnej bazy do nauki modeli oraz efektywnej oceny ich wydajności. Model numer 1 wykazał się niższą skutecznością w porównaniu do modelu numer 2. Świadczy o tym wartość błędu średniokwadratowego, która dla modelu pierwszego wyniosła 0.0668, natomiast dla modelu drugiego było to znacząco niższe, a mianowicie 0.0237. Ta różnica w wartościach MSE wskazuje na wyższą precyzję i efektywność modelu numer 2 w procesie uczenia i oceny na podstawie dostępnych danych.

4.1 Permutacyjna Ważność Cech

Permutacyjna Ważność Cech, to technika stosowana w uczeniu maszynowym do oceny znaczenia poszczególnych wejść dla modelu predykcyjnego. Metoda ta jest stosowana zarówno dla modeli klasyfikacyjnych, jak i regresyjnych. Proces ten rozpoczyna się od trenowania modelu na oryginalnym zestawie danych, co pozwala na ustalenie bazowej wydajności modelu. Następnie przeprowadza się permutację każdej cechy z osobna w zbiorze testowym, losowo mieszając jej wartości, podczas gdy wszystkie inne cechy pozostają niezmienione. Po dokonaniu permutacji model jest ponownie oceniany na zmodyfikowanym zbiorze danych. Wyniki tej oceny są następnie porównywane z wynikami uzyskanymi na oryginalnym, niezmodyfikowanym zbiorze. Różnica w wydajności modelu, taka jak spadek dokładności w klasyfikacji lub wzrost błędu średniokwadratowego w regresji, jest wykorzystywana do oceny ważności danej cechy. Im większy spadek wydajności, tym większa uważana jest ważność tej cechy dla modelu.

W kontekście wykorzystania PFI do oceny ważności cech w modelu istotne jest zwrócenie uwagi na kwestię korelacji między danymi. PFI opiera się na permutacji pojedynczych cech, co oznacza zmianę wartości jednej zmiennej niezależnie od pozostałych. W przypadku, gdy dane są silnie skorelowane, taka metoda permutacji może prowadzić do błędnych wniosków

W celu implementacji techniki PFI w pracy został wykorzystany obiekt PermutationImportance z biblioteki sklearn. Zastosowanie tego dedykowanego narzędzia umożliwiło nie tylko dokładną, ale i wydajną realizację tej metody, co pozwoliło na jej skuteczną integrację z procesem badawczym.

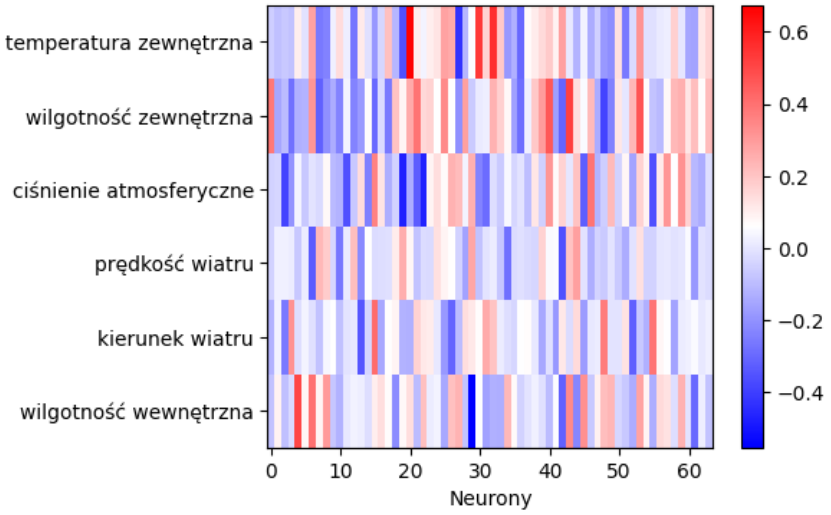
Tabela 4.1: Porównianie ważności wejść dla metody PFI

	Temperatura zewnętrzna	Wilgotność zewnętrzna	Ciśnienie atmosferyczne	Prędkość wiatru	Kierunek wiatru	Wilgotność wewnętrzna
Model krótki	0.39	1.00	0.66	0.15	0.01	0.44
Model długi	0.64	1.	0.54	0.04	0.19	0.81

Tabela 4.1 porównuje znormalizowane wartości wejść dla dwóch wariantów modelu dla metody PFI, w modelu uproszczonym, najwyższą ważność ma wilgotność zewnętrzna, co sugeruje, że jest to główny predyktor w tym modelu. Temperatura zewnętrzna i wilgotność wewnętrzna również odgrywają znaczące role. Wilgotność zewnętrzna mają umiarkowaną ważność, podczas gdy kierunek i prędkość wiatru ma stosunkowo niższą ważność. W Modelu rozszerzonym obserwujemy wzrost ważności ciśnienia atmosferycznego. Interesująco, wilgotność wewnętrzna otrzymuje maksymalną ważność, co czyni ją najważniejszą cechą dla tego modelu, z kolei prędkość i kierunek wiatru mają większy wpływ w porównaniu do Modelu krótkiego. Wilgotność zewnętrzna doznaje znaczącego obniżenia ważności, zostajac cechą o najniższej wartości.

4.2 Badanie wag wejściowych pierwszej warsty

W analizie modeli uczenia maszynowego, interpretacja wag pierwszej warstwy może służyć jako metoda określania ważności cech wejściowych. Ponieważ wagi w pierwszej warstwie sieci neuronowej są bezpośrednio połączone z cechami wejściowymi, wartości tych wag mogą dostarczać informacji o znaczeniu poszczególnych cech dla predykcji modelu. Wysoka wartość wagowa sugeruje, że zmiana wartości tej cechy wejściowej może mieć istotny wpływ na wynik modelu. Jednakże, ta metoda interpretacji może być mniej efektywna w przypadku bardziej złożonych, głębokich sieci neuronowych. W takich modelach, liczne warstwy i skomplikowane struktury, w tym nieliniowe aktywacje i interakcje między neuronami, mogą skutkować tym, że bezpośredni wpływ pojedynczych wag jest trudniejszy do zrozumienia.



Rysunek 4.2: Mapa ciepła wag pierwszej warstwy dla modelu o uproszczonej architekturze

Mapa ciepła wag pierwszej warstwy sieci neuronowej stanowi cenne narzędzie analityczne, pozwalające na wizualną interpretację i zrozumienie wpływu dużej liczby cech na proces uczenia. Jest to instrument szczególnie użyteczny w kontekście wysokowymiarowych zbiorów danych, gdzie tradycyjne metody analizy mogą okazać się niewystarczające. W przedstawionym przypadku, analiza mapy ciepła nie ujawnia istotnych anomalii w rozkładzie wag. Obserwuje się jedynie sporadyczne wartości, które odstają od średnich wag, lecz nie osiągają one poziomu znacząco wpływającego na wyniki modelu.

W celu dokładnej analizy ważności poszczególnych wag wejściowych sieci neuronowej przeprowadzono obliczenie średniej wartości wagi dla każdego z wejść. Procedura ta umożliwiła identyfikację względnej ważności cech poprzez porównanie ich przeciętnego wpływu na aktywację neuronów w modelu. Następnie, aby umożliwić porównywalność wyników niezależnie od ich pierwotnej skali, dokonano normalizacji obliczonych średnich wag.

Tabela 4.2: Porównianie wag wejściowych dla pierwszej warstwy uproszczonego modelu

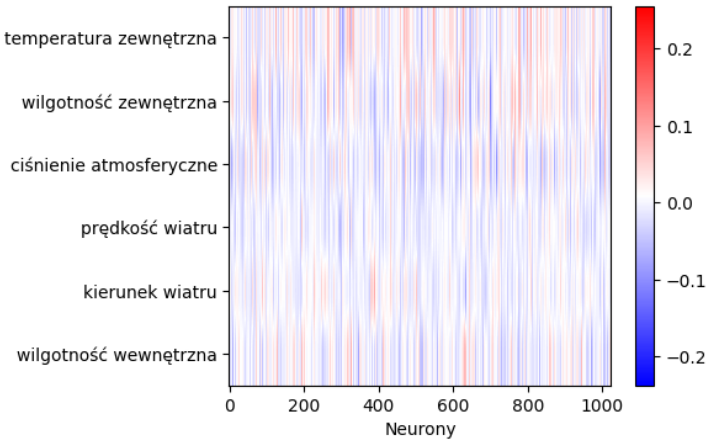
Temperatura zewnętrzna	Wilgotność zewnętrzna	Ciśnienie atmosferyczne	Prędkość wiatru	Kierunek wiatru	Wilgotność wewnętrzna
0.63	1.00	0.31	0.13	0.22	0.72

Analiza ważności wejść pierwszej warstwy sieci neuronowej, przeprowadzona za pomocą metody badania wag wejściowych, ujawnia istotne różnice w porównaniu z wynikami uzyskanymi metodą Permutation Feature Importance dla modelu o mniej rozbudowanej architekturze. W przeprowadzonych badaniach największą ważność przyznano ciśnieniu atmosferycznemu, co stanowi odchylenie od wyników w wynikach PFI.

Dodatkowo, warto zauważyć, że w badaniu wag wejściowych pierwszej warstwy, temperatura zewnętrzna otrzymała ważność równą 0.92, co jest wynikiem świadczącym o jej istotnym wpływie na model. Jest to wartość znacznie przewyższająca inne cechy, co sugeruje, że temperatura zewnętrznai ciśnienie atmosferyczne pełnią dominującą rolę w przewidywaniach modelu. W kontraście do nich, wilgotność zewnętrzna, prędkość wiatru oraz kierunek wiatru uzyskały ważności w okolicach 0.5, co wskazuje na ich umiarkowany, lecz w miarę zrównoważony wkład w działanie modelu.

Z kolei kierunek wiatru został oceniony jako cecha o najniższej ważności w badaniu wag wejściowych, jednakże nawet ta najniższa wartość (0.4) jest znacznie wyższa niż najniższe ważności zarejestrowane w metodzie PFI, gdzie dla wilgotności wewnętrznej odnotowano wartość minimalną na poziomie 0.03. Jest to szczególnie godne uwagi, gdyż wskazuje to na różnice w interpretacji znaczenia poszczególnych cech w zależności od zastosowanej metody analizy ważności wejść.

Analizy bardziej złożonych modeli sieci neuronowych, metoda interpretacji wag pierwszej warstwy może okazać się nieefektywna. Ze względu na zwiększoną głębokość i złożoność architektury, wagi w pierwszej warstwie tracą bezpośrednią i jednoznaczną interpretowalność, która jest charakterystyczna dla prostszych modeli. W modelach rozbudowanych, cechy wejściowe przechodzą przez wiele warstw transformacji, co skutkuje utratą bezpośredniego powiązania między wagami pierwszej warstwy a wynikowymi decyzjami modelu. W efekcie, interpretacja tych wag może nie odzwierciedlać faktycznego wpływu poszczególnych cech na decyzje modelu, co jest spowodowane nakładaniem się, transformacją i połączeniem informacji w kolejnych warstwach sieci.



Rysunek 4.3: Mapa ciepła wag pierwszej warstwy dla modelu o rozbudowanej architekturze

Na przedstawionej mapie cieplnej, ilustrującej wagi przypisane poszczególnym neuronom w warstwie wejściowej sieci neuronowej o rozmiarze 1024 jednostek, zaobserwować można istotne wzorce, choć sama mapa staje się trudna do interpretacji ze względu na dużą gęstość neuronów. Mimo tej wizualnej złożoności, zauważyć można, że temperatura zewnętrzna jest reprezentowana przez znacząco większą liczbę wag o dodatnich wartościach. W przeciwieństwie, ciśnienie atmosferyczne wykazuje tendencję do posiadania większej ilości wag o wartościach ujemnych. Wagi odpowiadające kierunkowi wiatru wydają się charakteryzować przewagą wartości bliskich zeru, co sugeruje mniejszy wpływ tej cechy na model.

Tabela 4.3: Porównianie wag wejściowych dla pierwszej warstwy rozbudowanego modelu

Temperatura zewnętrzna	Wilgotność zewnętrzna	Ciśnienie atmosferyczne	Prędkość wiatru	Kierunek wiatru	Wilgotność wewnętrzna
1.00	0.46	0.63	0.41	0.20	0.00

Analiza przeprowadzona z wykorzystaniem mapy cieplnej okazała się w tym przypadku być wyjątkowo efektywnym narzędziem, pozwalającym na wizualne wyodrębnienie cech o dużej wadze dla modelu. Obserwacje dotyczące tempetaru zewnętrznej i ciśnienia atmosferycznego pokrywają sie z wynikami otrzymanymi z analizy wag wejść do pierwszej warstywy. Interesującym zaskoczeniem okazała się natomiast rola kierunku wiatru, który początkowo oszacowany jako cecha mało znacząca, okazał sie znacznie znaczącym.

4.3 LIME

W kontekście zrozumienia modeli uczenia maszynowego, metoda Local Interpretable Model-agnostic Explanations jest istotnym narzędziem, które umożliwia interpretację decyzji modelu na poziomie lokalnym. Metoda ta wyróżnia się spośród innych, takich jak PFI, dzięki swojemu unikalnemu podejściu skoncentrowanemu na pojedynczych instancjach danych. Podczas gdy PFI dąży do zapewnienia ogólnego zrozumienia wpływu cech na model przez zbieranie i analizowanie informacji z różnych instancji, LIME skupia się na wyjaśnieniu, w jaki sposób model dokonuje przewidywań dla konkretnej, wybranej próbki danych.

Podstawowym elementem metody LIME jest modyfikacja danych i tworzenie na ich podstawie uproszczonego modelu, który ma za zadanie odwzorować zachowanie oryginalnego, skomplikowanego modelu, ale tylko w ograniczonym, lokalnym obszarze wokół analizowanej instancji. Proces ten rozpoczyna się od stworzenia zbioru danych przez modyfikację wybranej instancji, co skutkuje powstaniem podobnych, ale nieidentycznych przykładów. Następnie, na tych zmodyfikowanych danych, oblicza się przewidywania za pomocą oryginalnego modelu. Kluczowym krokiem jest trenowanie prostego modelu, takiego jak regresja liniowa, na podstawie tych przewidywań. Model ten służy do aproksymacji wpływu zmian w danych na przewidywania modelu. W ten sposób, analizując współczynniki modelu liniowego, można zrozumieć, które cechy miały największy wpływ na przewidywania dla danej instancji.

W ramach pracy, w celu dokładnej oceny skuteczności metody LIME, zdecydowano się na zastosowanie jej na losowo wybranych próbkach ze zbioru testowego.

Tabela 4.4: Porównanie ważności wejść dla metody LIME

	Temperatura zewnętrzna	Wilgotność zewnętrzna	Ciśnienie atmosferyczne	Prędkość wiatru	Kierunek wiatru	Wilgotność wewnętrzna
Model 1	1.00	0.08	0.68	0.39	0.04	0.81
	1.00	0.37	0.58	0.28	0.22	0.15
	0.82	0.14	1.00	0.34	0.05	0.62
Model 2	1.00	0.79	0.41	0.20	0.25	0.08
	1.00	0.16	0.14	0.11	0.09	0.76
	0.60	1.00	0.20	0.30	0.70	0.50

Tabela przedstawia wyniki sześciu losowo wybranych prób oceny ważności wejść, z zastosowaniem zarówno modelu uproszczonego, jak i rozszerzonego. W celu ułatwienia porównania, wyniki te zostały znormalizowane i zaokrąglone do dwóch miejsc po przecinku. Analiza danych wykazała, że w dwóch na trzy przypadki, zarówno w modelu uproszczonym, jak i rozszerzonym, najważniejszym wejściem okazała się być temperatura zewnętrzna. Ponadto, w pięciu z sześciu analizowanych przypadków, kierunek wiatru został oceniony jako najmniej znaczący.

Szczególnie interesująca jest obserwacja z pierwszej próby, gdzie temperatura zewnętrzna otrzymała znacznie niższą wagę, podczas gdy ciśnienie atmosferyczne, które zazwyczaj charakteryzowało się niższymi wagami, w tej próbie uzyskało najwyższą. Z kolei prędkość wiatru, niezależnie od zastosowanego modelu, konsekwentnie otrzymywała niską wagę w każdej z prób.

Mimo iż LIME jest metodą zasadniczo skoncentrowaną na dostarczaniu interpretacji lokalnych, istnieje możliwość adaptacji jej do generowania wniosków o charakterze bardziej ogólnym. Można to osiągnąć poprzez zastosowanie metody LIME wielokrotnie na różnych próbkach danych, a nastepnie wyciąganięcie średniej. Zaproponowano wybranie czterech zestawów ze zbioru testowego, zawierających odpowiednio 10, 50, 100, i 1000 próbek.

Tabela 4.5: Porównanie uśrednionych ważności wejść dla metody LIME dla wielu prób

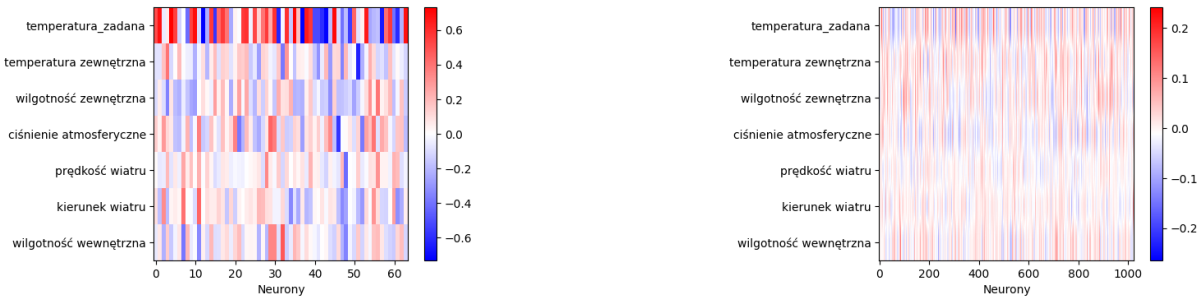
	Ilość próbek	Temperatura zewnętrzna	Wilgotność zewnętrzna	Ciśnienie atmosferyczne	Prędkość wiatru	Kierunek wiatru	Wilgotność wewnętrzna
Model 1	10	0.70	1.00	0.12	0.08	0.42	0.39
	50	1.00	0.24	0.27	0.00	0.08	0.09
	100	1.00	0.43	0.46	0.09	0.09	0.07
	1000	1.00	0.37	0.44	0.12	0.13	0.07
Model 2	10	1.00	0.53	0.52	0.15	0.19	0.00
	50	1.00	0.39	0.01	0.10	0.06	0.00
	100	1.00	0.25	0.26	0.19	0.12	0.03
	1000	1.00	0.61	0.26	0.19	0.08	0.00

Zarówno w Modelu 1, jak i Modelu 2, temperatura zewnętrzna jest systematycznie uznawana za cechę o największym wpływie na przewidywania modelu, co wskazuje na jej przewodnią rolę w analizowanych danych meteorologicznych. Niezmiennie wysokie wartości ważności tej zmiennej świadczą o jej dominującej pozycji w obu modelach, niezależnie od wielkości próby. Dla obu modelów, wilgotność jak i ciśnienie wykazują stabilne wartości, tylko w pojedyńczych próbach ciśnienie staja sie dominującą zmienną. Prędkość wiatru, która generalnie otrzymuje niższe ważności, ujawnia znaczący wzrost w przypadku 1000 próbek w Modelu 2. To może wskazywać na to, że dla bardziej złożonych analiz, gdy dostępna jest większa liczba danych, prędkość wiatru zaczyna wykazywać silniejsze korelacje z wynikami modelu bardzo złożonego.Kierunek wiatru dla modelu pierwszego pozostaje cechą o niskiej ważności, lecz dla modelu rozbudowanego zystkujena znaczniu. Wilgotność wewnętrzna konsekwentnie otrzymuje najniższe wartości ważności, co wskazuje na jej marginalną rolę w procesie decyzyjnym obu modeli. Wilgotność wewnętrzna i zewnętrzna zostały zidentyfikowane jako wejścia o najmniejszym wpływie na model.

4.3.1 Dodanie nowego parametru

Tabela 4.6: TODO

	Bez dodatkowego parametru	Z dodatkowego parametru
Model 1	0.0706	0.0691
Model 2	0.0262	0.0230



(a) TODO (b) TODO

Rysunek 4.4: TODO

Tabela 4.7: TODO

	Metoda	Temperatura SET	Temperatura zewnętrzna	Wilgotność zewnętrzna	Ciśnienie atmosferyczne	Prędkość wiatru	Kierunek wiatru	Wilgotność wewnętrzna
Model 1	PFI	1 00	0.50	0.00	0.48	0.86	0.87	0.46
	Wagi wejściowe	1.00	0.19	0.28	0.37	0.41	0.03	0.00
	$\text{LIME}_{śr(100)}$	1.00	0.11	0.40	0.34	0.09	0.05	0.01
Model 2	PFI	1 00	0.62	0.00	0.13	0.58	0.73	0.11
	Wagi wejściowe	0.80	0.76	0.28	1.00	0.70	0.24	0.05
	$\text{LIME}_{śr(100)}$	1.00	0.17	0.11	0.07	0.02	0.00	0.01

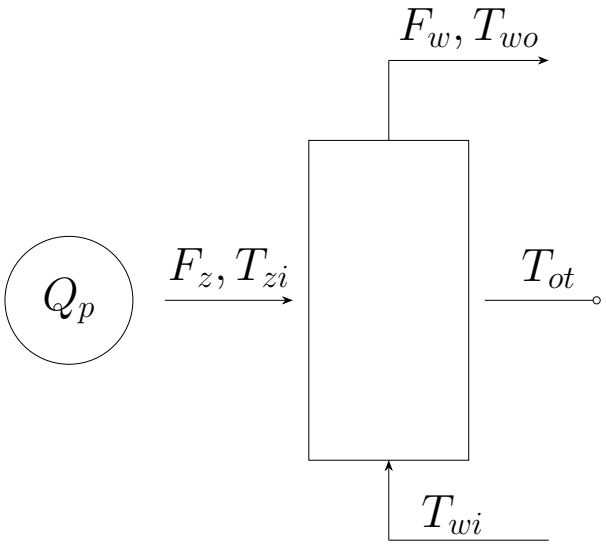
Rozdział 5

Modelowanie zbiornika CWU

5.1 Metodologia

linearyzacja [2]
inne podejścia[7, 1]

5.1.1 Model jednowarstwowy



Rysunek 5.1: Model zasobnika - pojedyńcza warstwa

$$\rho C_w V \frac{dT_{wo}}{dt} = Q_p - \rho C_w F_w (T_{wo} - T_{wi}) - \frac{\lambda A}{d}(T_{wo} - T_{ot}) \tag{5.1}$$

gdzie:

A Powierzchnia wymiany ciepła DO POPRAWY

d powierzchnia wymiany ciepła DO POPRAWY

V Objętość

C_w Ciepło właściwe

λ Współczynnik wymiany ciepła

Q_p Ciepło systemu grzewczego

T_{ot} Temperatura otoczenia

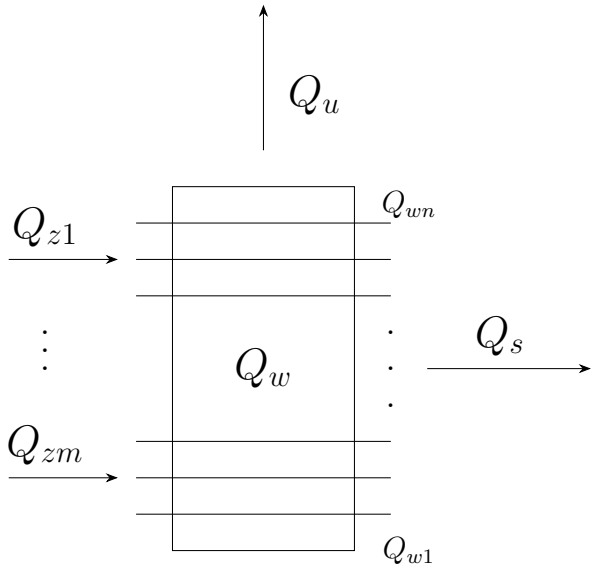
T_{wo} Temepratura wyjściowa

T_{wi} Temperatura wejściowa

F_w TODO

F_z TODO

5.1.2 Model wielowarstwowy



Rysunek 5.2: Model zasobnika - wiele warstwa

$$Q_w = \sum_{m=1}^{m} Q_m - Q_u - Q_s \tag{5.2}$$

$$Q_{wn} = \sum_{n=1}^{m(n)} Q_{m(n)} - Q_{un} - Q_{sn} \tag{5.3}$$

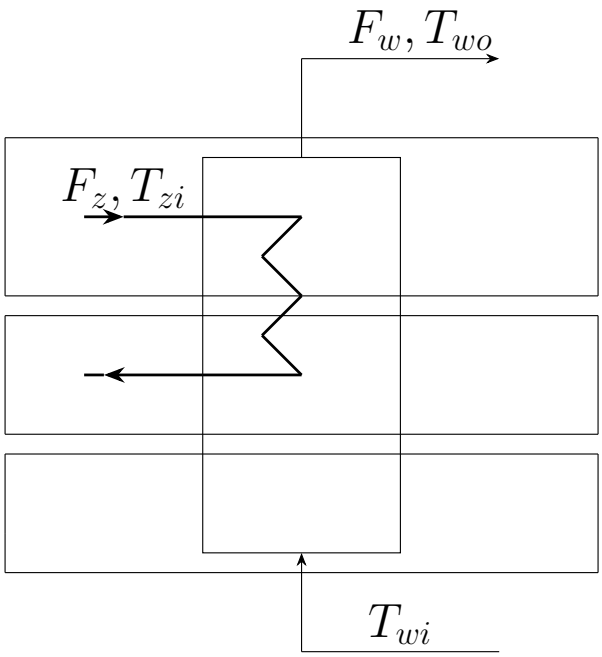
Q_m

Q_w Ciepło całego zbiornika

Q_{wm} Ciepło warstwy

Q_u Ciepło upuszczające zbiornik DO POPRAWY

Q_s Straty ciepła



Rysunek 5.3: Trójwarstwowy model zasobnika

$$\frac{dT^3_{wo}}{dt} = b^3_1 F_z(T_{zi} - T^3_{wo}) - b^3_2 F_w(T^3_{wo} - T^2_{wo}) - b^4_3(T^3_{wo} - T_{ot}) \tag{5.4}$$

$$\frac{dT_{zi}}{dt} = p_1 Q_g - p_2 F_z(T_{zi} - T^3_{wo}) - p_3(T_{zi} - T_{ot}) \tag{5.5}$$

$$\frac{dT^2_{wo}}{dt} = b^2_1 F_z(T_{zi} - T^2_{wo}) - b^2_2 F_w(T^2_{wo} - T^1_{wo}) - b^2_3(T^2_{wo} - T_{ot}) - b^2_4(T^2_{wo} - T^1_{wo}) + b^2_5(T^3_{wo} - T^2_{wo}) \tag{5.6}$$

$$\frac{dT^1_{wo}}{dt} = -b^1_2 F_w(T^1_{wo} - T_{wi}) - b^1_3(T^1_{wo} - T_{ot}) + b^1_5(T^2_{wo} - T^1_{wo}) \tag{5.7}$$

Przedstawienie modelu warstwowego, równań stanu, pokazanie wyników symulacji modelu

5.2 Wyniki symulacji

Rozdział 6

Optymalizacja

6.1 Funkcja kosztów

$$G = \int p_1 Q_g \, dt \tag{6.1}$$

6.2 Funkcja komfortu

$$J = \int (T_{wo} - T_{wym})^2 \left| \frac{\text{sign}(T_{wo} - T_{wym} - \delta) + \text{sign}(T_{wo} - T_{wym} + \delta)}{2} \right| dt \tag{6.2}$$

Rozdział 7

Podsumowanie i wnioski

Bibliografia

[1] S. Alizadeh. „An Experimental and Numerical Study of Thermal Stratification in a Horizontal Cylindrical Solar Storage". W: *Solar Energy* (1999).

[2] Carriere Claquin i Rocaries. *Modelling and application of adaptive control to a gas heater*. Glasgow, UK: Proceedings of IEEE International Conference on Control i Applications, 1994.

[3] Google Cloud. *Introduction to Cloud TPU*. 2023. URL: `https://cloud.google.com/tpu/docs/intro-to-tpu` (term. wiz. 08. 01. 2024).

[4] Google Colab. *Welcome to Colaboratory*. 2023. URL: `https://cloud.google.com/tpu/docs/intro-to-tpu` (term. wiz. 08. 01. 2024).

[5] R.G. Palmer J. Hertz A. Krogh. *Wstęp do teorii obliczeń neuronowych*. Warszawa: WNT, 1993.

[6] K. Gamage L. Gelažanskas. „Forecasting hot water consumption in residential houses". W: *Energies 8* (2015), s. 12702 –1271.

[7] J. Streicher M. Y. Haller. „Comparative Analysis of Thermal Energy Storage Stratification Efficiency". W: *Thermal Energy Storage for Efficiency and Sustainability* (2009).

[8] M.J. Booysen M.J. Ritchie J.A.A. Engelbrecht. „A probabilistic hot water usage model and simulator for use in residential energy management". W: *Energy & Buildings* 8 (2021).

[9] S. Osowski. *Sieci neuronowe w ujęciu algorytmicznym*. Warszawa: WNT, 1996.

[10] N.D. Pflugradt. *Modeling water and energy consumption in households*. 2016. URL: `https://nbn-resolving.org/urn:nbn:de:bsz:ch1-qucosa-209036` (term. wiz. 08. 01. 2024).

[11] A. Laperrièr S. Edwards I. Beausoleil-Morrison. „Representative hot water draw profiles at high temporal resolution for simulating the performance of solar thermal systems". W: *Solar Energy* 111 (2015), s. 43–52.

Dodatki

Spis skrótów i symboli

ASHRAE	American Society of Heating, Refrigerating and Air-Conditioning Enginee
ARIMA	Autoregressive integrated moving average
MISO	Multiple Input, Single Output
CUDA	Compute Unified Device Architecture
GPU	Graphics processing unit
CPU	Central processing Unit
LIME	Local Interpretable Model-agnostic Explanations
PFI	Permutation Feature Importance
$f(\cdot)$	Funkcja aktywacji neuronu
y	sygnał wyjściowy neuronu
x	sygnał wejsciowy neuronu
v	pobudzenie neuronu
w_j	Waga neuronu
b	Składnik stały
$T_{zi}^{n,m}$	Temperatura systemu grzewczego
T_{wo}^{n}	Temepratura warsty
T_{ot}	Temperatura otoczenia
F_w	Przepływ wody w zbiorniku
F_z	Przepływ wody systemu grzewczego
Q_p	Ciepło systemu grzewczego

Q_{zm} Cieplo dostarczane do zasobnika

Q_m

Q_w Ciepło całego zbiornika

Q_{wm} Ciepło warstwy

Q_u Ciepło upuszczające zbiornik DO POPRAWY

Q_s Straty ciepła

A Powierzchnia wymiany ciepła DO POPRAWY

d powierzchnia wymiany ciepła DO POPRAWY

V Objętość

C_w Ciepło właściwe

λ Współczynnik wymiany ciepła

n numer warstwy

m numer źródła ciepła

ϕ_{zew} wilgotność zewnętrzna

ϕ_{wew} wilgotność wewnętrzna

$\longrightarrow$ Kierunek wiatru

V_{wia} Prędkość wiatru

T_{zew} Temperatura zewnętrzna

P_{zew} Ciśnienie zewnętrzne

Lista dodatkowych plików, uzupełniających tekst pracy

1. Dane
   (a) Dane od ASHRAE i ich przetworzona wersja
   (b) Dane potrzebne do oceny wazności wejść
2. Skrypty
   (a) Skrypt 1.py -robi coś
   (b) Skrypt 1.py -robi coś
3. Archiwum modeli opisanych w pracy

Spis rysunków

Spis tabel