Politechnika Wrocławska
Wydział Informatyki i Telekomunikacji

Kierunek: **Inżynieria Systemów (INS)**
Specjalność: **Inżynieria Danych (IND)**

PRACA DYPLOMOWA
INŻYNIERSKA

Moduł samouczący do systemu automatyki domowej

Przemysław Barcicki

Opiekun pracy
dr. inż. Patryk Schauer

Słowa kluczowe: automatyzacja domowa, uczenie maszynowe, smart-home

WROCŁAW 2023

STRESZCZENIE

Tutaj sobie na razie zostawie lorem ipsum, streszczenie napisze pod koniec/później w trakcie pisania.

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

ABSTRACT

Same as above c:

Nam dui ligula, fringilla a, euismod sodales, sollicitudin vel, wisi. Morbi auctor lorem non justo. Nam lacus libero, pretium at, lobortis vitae, ultricies et, tellus. Donec aliquet, tortor sed accumsan bibendum, erat ligula aliquet magna, vitae ornare odio metus a mi. Morbi ac orci et nisl hendrerit mollis. Suspendisse ut massa. Cras nec ante. Pellentesque a nulla. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Aliquam tincidunt urna. Nulla ullamcorper vestibulum turpis. Pellentesque cursus luctus mauris.

SPIS TREŚCI

1. WSTĘP

1.1. WPROWADZENIE DO PROBLEMATYKI

Automatyzacja to określenie na metody mające na celu ograniczenie ludzkiej interakcji do minimum w różnych procesach. Stosuje się ją w wielu dziedzinach począwszy na przemyśle, kończąc na procesach informatycznych. Pewnym jednym konkretnym obszarem zyskującym w ostatnim czasie dużo zainteresowania jest pojęcie inteligentnego domu (smart-home). Automatyka domowa to konkretne określenie na zastosowanie automatyzacji wśród urządzeń smart-home skupiających się na obszarze zastosowań gospodarstwa domowego. Celem takiego zastosowania jest kontrola pracy urządzeń znajdujących się w domu do osiągnięcia konkretnego ich stanu w sposób minimalizujący ludzką interakcję. W praktyce takie systemy to bardzo dobre rozwiązanie które niesie ze sobą wiele zalet i uproszczeń w codziennym życiu [8].

Zasada czy reguła w automatyce to w dokładny sposób określenie jakie akcje muszą zostać wykonane pod warunkiem pewnego stanu systemu. Warunkiem początkowym może być każdy stan dowolnego urządzenia w systemie, zmiana stanu dowolnego urządzania lub całkowicie zewnętrznego bodźca. W przypadku takiej automatyzacji często też korzysta się z kombinacyjnego połączenia wielu źródeł i informacji w celu stworzenia stanu początkowego, który jeśli wystąpi, jest przesłanką do wysłania przez system zarządzający urządzeniami polecenia do wykonania pojedynczej lub ciągu akcji. Każde dostępne na rynku dedykowane oprogramowanie obsługujące automatykę domową dostarcza różne narzędzia do szybkiego i intuicyjnego sporządzania takich zasad ale także dostarcza sposoby na komunikację z tymi urządzeniami. Mamy zatem pełny system, który agreguje wiele systemów, protokołów i sposobów wymiany danych w celu monitorowania i sterowania urządzeniami w ramach (lokalnej) sieci.

1.2. OPIS PROBLEMU

Mimo tego, że analiza i projektowanie systemów inteligentnej automatyki domów zaczęła się kilka dekad temu, istnieje wiele nierozwiązanych problemów, które muszą zostać rozwiązane, aby taka forma stała się popularna. Problemami są nieprawidłowe algorytmy oraz niskie celności wynikowe [6].

Mimo wygody w używaniu i łatwości w sporządzaniu zasad automatyki domowych zastosowań pojawiają się pewne trudności. Jednym z nich jest problem ze zmiennością

ludzkich nawyków. Sporządzenie zasad sprawia, że dane czynności są wykonywane wtedy i tylko wtedy gdy zostanie spełniony pewien konkretny stan określony przez nas podczas sporządzania danej reguły. Z logicznego punktu widzenia jest to odpowiednia odpowiedź systemu, natomiast z tego praktycznego już nie, ponieważ dana reguła może zostać wykonana mimo tego, że nie chcieliśmy aby się wykonała. Dodaje to pewien wymóg dodatkowego komplikowania danej zasady poprzez dodatkowe warunki w systemie lub ciągłe jej edytowanie aby odpowiadała naszym zmiennym nawykom.

Bez względu na to jakie narzędzia dostarcza nam dany system, sporządzanie dużej ilości skomplikowanych reguł może być problematyczne. Same środowiska zarządzania takimi systemami mimo wspierania tworzenia na tyle skomplikowanych zasad mogą nie być do tego przystosowane. Często zdarza się, że system automatyki udostępnia dodatkowe, jako wtyczki lub dodatki, moduły do tworzenia reguł za pomocą interpretowanych języków programowania [1], [9], [3]. Tworzy to natomiast zestaw innych problemów, gdzie użytkownik chcący stworzyć takie reguły, musi w przynajmniej podstawowy sposób znać języki języki programowania oraz dodatkowo w jaki sposób sporządzać te skrypty aby mogły być wykonywane przez oprogramowanie nadzorujące.

1.3. CEL PRACY

Celem niniejszej pracy jest opracowanie modułu do systemu automatyki domowej (smart-home) wspierającego tworzenie reguł decyzyjnych w oparciu o mechanizmy uczące się zachowań użytkownika.

1.4. ZAKRES PRACY

Zakres pracy obejmuje przegląd literatury w zakresie działania i budowy systemów wspomagania podejmowania decyzji podejmowania decyzji opartych o uczenie maszynowe.

2. PRZEGLĄD ISTNIEJĄCYCH ROZWIĄZAŃ

W tym rozdziale omówiono literaturę i istniejące rozwiązania problemu rozwiązywanego w dalszej części tej pracy. Pod uwage brano prace, które korzystają z klasycznych metod uczenia maszynowego oraz metod zawierających elementy sieci neuronowych. Analizie zostaną poddane użyte algorytmy jak i podejścia do osiągnięcia celu wspomagania decyzji automatyki domowej.

2.1. SYSTEMY KORZYSTAJĄCE Z ALGORYTMÓW WYKRYWANIA EPIZODÓW

Episode Discovery (wykrywanie epizodów) to metoda data mining'u (kopania danych) wykorzystująca istniejący ciąg występujących po sobie wydarzeń do wykrywania w nich powtarzalnych znaczących epizodów. Wśród epizodów można wyróżnić, tak zwane, epizody znaczące, które według zależnych od algorytmu charakterystyk, określają dany epizod jako często występujący.

Czesto można spotkać się pewną interpretacją ciągu zdarzeń w systemie gdzie każda występująca po sobie w pewnym oknie czasowym akcja jest reprezentowana jako odpowiedni znak ze zbioru wybranych możliwych zdarzeń. W przypadku opisu stanu systemu za pomocą reprezentacji znakowej, korzysta się z metodyki, gdzie duża litera oznacza przejście stanu danego urządzenia w załączone, a mała litera oznacza wyłączenie. W przykładowych ciągach zdarzeń, `baDgb`, `abD`, `Dagb`. Można zauważyć, że zawsze po wystąpieniu wyłączenia urządzenia `a`, następuje wyłączenie urządzenia `b`. W rzeczywistości algorytmy są bardziej zaawansowane i wykrywają epizody znaczące które występują często, ale nie zawsze, ale także operują na dużo dłuższych łańcuchach zdarzeń.

Wykorzystywanie algorytmów wykrywania epizodów na zapisanych już strumieniach wydarzeń pozwala na znalezienie pewnych nawyków i zależności z codziennego korzystania z domowych urządzeń. Znalezione i wyodrębnione epizody mogą zostać użyte z innymi algorytmami w celu podniesienia ich celności. Tak przetworzone dane wejściowe z dodatkowym użyciem innego algorytmu dają zdecydowanie lepsze wyniki niż w wypadku użycia samych sieci neuronowych bądź samego wykrywania epizodów [4], [6].

Ważnym elementem wykorzystania technik wykrywania epizodów jest prawidłowy wybór algorytmu (SPADE, SPEED, WINEPI, ...) jak i jego hiperparametrów, ponieważ inne wartości parametrów wybierających epizod znaczący może mocno wpływać na końcowy wynik [6]. O ile w przypadku algorytmu, nieprawidłowy wybór może ograniczać się do

wydłużonego czasu poszukiwania epizodów, a co za tym idzie, większego zużycia energii przez system, tak w przypadku niewłaściwych parametrów, algorytm może proponować za dużo akcji i nawyków, które źle będą wpływały na końcowy wynik, co w końcu sprawi, że komfort korzystania z takiego systemu będzie mały.

Podejście wykrywania epizodów, jest metodą skupiającą się na pewnych ciągach zdarzeń, które nie biorą pod uwagę żadnych innych zewnetrznych parametrów. Sprawia to, że system uczy się ciągów wydarzeń bez względu na aktualny stan systemu, czyli na przykład porę dnia, temperaturę w pomieszczeniu, dzień tygodnia czy też pogodę. Połączenie klasycznych algorytmów episode discovery z dodatkowymi algorytmami które, biorą pod uwagę inne parametry systemu, rozwiązuje ten problem, ale dodaje dużo skomplikowania w implementacji.

Korzystanie z metody dużych i małych liter do oznaczania wyłączeń i włączeń urządzeń tworzy dodatkowo pewne ograniczenia w postaci braku lub bardzo skomplikowanej obsługi urządzeń o niebinarnych stanach. O ile w przypadku urządzeń gdzie stanów jest kilka, jak na przykład systemy HVAC, można każdy z trybów pracy zinterpretować jako inną czynność, tak w przypadku urządzeń gdzie istnieje teoretycznie nieskończenie wiele stanów pośrednich, tak jak na przykład w ściemniaczach żarówkowych, nie jest możliwe reprezentowanie każdego z nich.

2.2. SYSTEMY KORZYSTAJĄCE Z PROCESÓW MARKOWA

Podczas modelowania zmian stanu systemu automatyki domowej można skorzystać z podejścia gdzie próbujemy opisać ciąg zdarzeń za pomocą prawdopodobieństwa przejść między stanami. Bardzo pomocne w takim podejściu okazuje się korzystanie z modeli Markowa. Istnieje pewne rozszerzenie modeli, które okazuje się bardziej pomocne w wypadku modelowania ludzkiej interakcji i zachowań z powodu uwzględnienia niezależnych i nieznanych przez system stanów, nazywanym ukrytym modelem Markowa. Czyste podejście z prawdopodobieństwem odpowiada na pytanie, jaka jest szansa na wykonanie akcji `A` pod warunkiem tego, że poprzednio wykonana akcja to `B`, dodatkowe ukryte informację pomagają w dokładniejszym określeniu przewidywanego stanu systemu.

Podobnie jak w przypadku wykrywania epizodów (2.1) do wytworzenia reprezentacji zmian systemu wykorzystywany jest literowy zapis, co sprawia, że te podejścia mają takie same ograniczenia. Jedna z prac [11], próbuje rozwiązać problem związany z rzeczywistą (niebinarną) naturą pewnych stanów poprzez kwantyzację w konkretne stany. Konkretne wartości temperaturowe zamieniane są na arbitralny identyfikator wskazujący na daną temperaturę, co używane jest w modelu jako jeden z parametrów.

Podejście znalezione w [11], próbuje także rozwiązać za pomocą autorskiej metody IHMM (Improved Hidden Markov Models) problem powiązania pewnych konkretnych nawyków i ciągów wydarzeń z pewną temporalną zmienną, co sprawia, że system podpo-

wiada konkretne akcje dokładniej o konkretnych porach dnia, lecz dalej nie rozwiązuje problemu rozpoznania np. dni tygodnia czy pogody na dworze.

Inna implementacja [2], korzysta z innego, również autorskiego rozwinięcia modeli markowa TMM (Task-based Markov Model), w której skupia się na zidentyfikowaniu wysokopoziomowych zadań, które to dalej są wykorzystywane do stworzenia pomniejszych modeli reprezentujących ciąg zdarzeń w konkretnym zadaniu. Zadania są identyfikowane na podstawie przerwy pomiędzy kolejnymi wydarzeniami oraz na podstawie dodatkowej informacji o lokalizacji danego urządzenia. Zadania wraz z informacjami o ilości, długości i położenia w przestrzeni używane są przez algorytm k-średnich do pogrupowania zadań i stworzenia zbioru konkretnych ciągów.

2.3. SIECI NEURONOWE

Często spotyka się w parze z innym algorytmem sieci neuronowe, ponieważ są w stanie poprawić zdecydowanie celność predykcji niskim kosztem [4]. Istnieją pewne rozwiązania, gdzie sieci neuronowe są głównym sposobem na przeprowadzania predykcji. Ich szerokie zastosowanie i wiele istniejących rozwiązań sprawia, że są w stanie występować na prawie każdej platformie obliczeniowej. Dodatkowo w ciągu ostatniej dekady widać było zdecydowany rozrost zastosowań i nowych rozwiązań korzystających w pewien sposób z głębokich sieci neuronowych.

Różne istniejące zastosowania i modele głębokich sieci neuronowych daje duże pole do ich zastosowań. W przypadku pracy autorstwa S. Umamageswari [10], zastosowano jedno z popularnych w ostatnim czasie rozwiązań LSTM (long short-term memory). Rekurencyjne sieci neuronowe korzystające z warstw LSTM są w stanie zauważać występujące w dużej odległości danych uczących po sobie pewne zależności. Jest to dobre zastosowanie dla sieci wykorzystujących dużą ilość danych wejściowych zawierające pewne informacje temporalne. Takie modele matematyczne są w stanie "nauczyć" się ciagu konkretnych zadań wykonywanych o konkretnych porach dnia, prezentując systemowi przykładowe historyczne wystąpienia tych zadań [7].

Ciekawym podejściem okazuje się wykorzystanie konwolucyjnych sieci neuronowych w przypadku wprowadzania dodatkowych zmiennych do systemu [5]. System w celu określenia następnej akcji którą ma podjąć bierze pod uwagę wyraz twarzy użytkownika. Obraz twarzy, przekształcany jest przez sieci konwolucyjne do zestawu informacji mówiącym o samopoczuciu użytkownika.

2.4. PODSUMOWANIE I WNIOSKI

Biorąc pod uwagę przeprowadzoną analizę, widać, że wstępne przetwarzanie danych i odpowiednie ich spreparowanie jest bardzo kluczowe. Prawidłowa reprezentacja stanu w

systemie jest kluczowa, ponieważ to na podstawie tego system będzie przeprowadzał swoje predykcje. W przypadku tradycyjnych algorytmów uczenia maszynowe widać ich zdecydowane ograniczenia, których rozwiązanie wymaga zdecydowanego komplikowania problemu poprzez wprowadzanie kwantyzowanych informacji czy zmiennych reprezentujących wartości temporalne. Metodą dającą największą elastyczność okazuje się wykorzystanie głębokich sieci neuronowych, ze względu na ich możliwość, przy wcześniejszym odpowiednim przekształceniu wejścia lub wyjścia, wprowadzania danych o różnych formatach, czy sterowania szerokim zakresem różnych urządzeń.

3. WYMAGANIA PROJEKTOWE

Celem tego projektu jest wytworzenie systemu zdolnego do wspomagania użytkownika w eksploatacji inteligentnych urządzeń IoT znajdujących się w jego domowym środowisku poprzez wykonywanie pewnych czynności za użytkownika.

Automatyzacja nie powinna wykonywać się wprost, tj. bez ludzkiej interakcji. Użytkownik korzystający z tego systemu będzie spełniał kluczową rolę, ponieważ wymaga się, aby system dokańczał powtarzalne ciągi zadań wykonywanych przez użytkownika.

Wymagane jest aby rozpoczęcie używania tego systemu wymagała od użytkownika minimum konfiguracji, system powinien pobierać dane z istniejącego już systemu automatyki domowej i używać ich w celu wytworzenia reguł. Dodatkowo, sam system powinien być niezależny od architektury komputera na którym się znajduje. Zaleca się, aby moduł współpracował z gotowymi systemy automatyki domowej. W wypadku utraty przez komputer obsługujący system prądu, system powinien wznowić swoją pracę bez ponownego procesu tworzenia reguł decyzyjnych.

Wymaga się, aby system nie wymagał dużej ilości prac w celu dodania nowych typów źródeł i typów ujść danych, tak aby ewentualne dodawanie obsługi urządzeń dla których nie istnieje gotowe wsparcie było najprostsze.

W wypadku innej odpowiedzi systemu niż użytkownik oczekuje, użytkownik powinien mieć łatwy sposób na cofnięcie wykonania błędnej akcji do poprzedniego stanu.

Biorąc powyższe pod uwagę, można określić listę wymagań funkcjonalnych i niefunkcjonalnych tego systemu. Wymagania funkcjonalne:

— wspomaganie codziennej eksploatacji urządzeń w domu,
— cofanie akcji,
— odczyt oraz zapis do pamięci stałej,
— wsparcie istniejącego systemu automatyki,

Wymagania niefunkcjonalne:

— minimum konfiguracji,
— działanie niezależne od architektury komputera,
— łatwość w tworzeniu rozszerzeń,

4. IMPLEMENTACJA

4.1. ARCHITEKTURA

Na tym stadium pracy bardzo ważne jest dokładne zbadanie wymogów znajdujących się w rozdziale 3, w celu odpowiedniego przygotowania architektury systemu. Błędny wybór, może skutkować ograniczeniami co do zastosowań tego systemu.

4.1.1. HomeAssistant – HA

Jako system automatyki domowej, kontrolujący wymianę informacji naszego modułu ze światem rzeczywistym wybrano platformę HomeAssistant. Jest to jeden z największych niekomercyjnych systemów tego typu. Jego przewagą w porównaniu do niektórych innych dostępnych na rynku systemów są rzesze fanów i majsterkowiczów, którzy oferują świetne wsparcie i pomagają w rozwiązywaniu problemów. HA skupia się także na prywatności. Programiści open-source jak i użytkownicy systemu zalecają utrzymywanie systemu na swojej własnej infrastrukturze w domu. Dzięki wiernym fanom i samej architekturze projektu system posiada bardzo bogatą bibliotekę integracji z różnymi systemami, od systemów typu Google Assistant w celu dodawania funkcjonalności sterowania głosem, przez własnościowe systemy zarządzania oświetleniem w domu aż po wsparcie dodatkowych bezprzewodowych protokołów komunikacyjnych przeznaczonych do zastosowań domowych. Bardzo ważnym atutem poza szerokim polem różnych integracji jest także system rozszerzeń, gdzie użytkownik może dodać pewną kompletnie nieistniejącą w systemie funkcjonalność do poprawy działania systemu, czy automatyzacji innych rzeczy niezwiązanych z domem.

4.1.2. AppDaemon – AD

System pozwalający stworzenie modułu samouczącego powinien dawać maksimum możliwości i niezależności, zatem wybrano AppDaemon. AppDaemon to środowisko wykonawczego Pythona, w wielowątkowej architekturze piaskownicy, służące do pisania aplikacji automatyzacji dla projektów automatyki domowej (i nie tylko), których wymogiem jest solidna architektura sterowana zdarzeniami. AD jest od razu gotowe do współpracy z systemem HomeAssistant, co sprawia, że integracja systemów HA/AD jest bezproblemowa. System pozwala natychmiastowo reagować na zmiany stanów urządzeń znajdujących się w domowym środowisku, poprzez korzystanie z asynchronicznej architektury callback.

4.1.3. Docker

Ze względu na wymóg niezależności od platformy na jakiej będzie znajdować się ten system, zdecydowano o wyborze jako jednej z głównych technologii, systemu Docker w celu zapewnienia aplikacjom pracującym pod nim odpowiednich warunków niezależnie od systemu operacyjnego na jakim się znajduje. Dodatkowym atutem, który sprawił że wybrano

4.1.4. Tensorflow

Wszystkie moje ziomki korzystają z tf.

4.1.5. Python

Wykorzystanie struktury HA/AD ogranicza nas do wyboru Python jako przewodniego języka programowania w tym projekcie.

PODSUMOWANIE

Curabitur tellus magna, porttitor a, commodo a, commodo in, tortor. Donec interdum. Praesent scelerisque. Maecenas posuere sodales odio. Vivamus metus lacus, varius quis, imperdiet quis, rhoncus a, turpis. Etiam ligula arcu, elementum a, venenatis quis, sollicitudin sed, metus. Donec nunc pede, tincidunt in, venenatis vitae, faucibus vel, nibh. Pellentesque wisi. Nullam malesuada. Morbi ut tellus ut pede tincidunt porta. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Etiam congue neque id dolor.

BIBLIOGRAFIA

[1] Cockburn, A., *Dokumentacja appdaemon*.

[2] Cook, D.J., Youngblood, M., Heierman, E.O., Gopalratnam, K., Rao, S., Litvin, A., Khawaja, F., *Mavhome: An agent-based smart home*, w: *Proceedings of the First IEEE International Conference on Pervasive Computing and Communications, 2003.(PerCom 2003).* (IEEE, 2003), str. 521–524.

[3] openHAB Community openHAB Foundation, *Dokumentacja pisania skryptów*.

[4] Heierman, E., Cook, D., *Improving home automation by discovering regularly occurring device usage patterns*, w: *Third IEEE International Conference on Data Mining* (2003), str. 537–540.

[5] Majeed, R., Abdullah, N.A., Ashraf, I., Zikria, Y.B., Mushtaq, M.F., Umer, M., *An intelligent, secure, and smart home automation system*, Scientific Programming. 2020, tom 2020, str. 4579291.

[6] Marufuzzaman, M., Tumbraegel, T., Rahman, L.F., Sidek, L.M., *A machine learning approach to predict the activity of smart home inhabitant*, J. Ambient Intell. Smart Environ. 2021, tom 13, 4, str. 271–283.

[7] Pang, B., Nijkamp, E., Wu, Y.N., *Deep learning with tensorflow: A review*, Journal of Educational and Behavioral Statistics. 2020, tom 45, 2, str. 227–248.

[8] Szabłowski, S., *Projektowanie dydaktycznych systemów automatyki domowej*, Dydaktyka informatyki. 2019, , 14, str. 137–146.

[9] Team, D., *Dokumentacja pisania skryptów domoticz*.

[10] Umamageswari, S.S.M., Kannan, M.K.J., *Tensorflow-based smart home using semi-supervised deep learning with context-awareness*, Journal of Mathematical and Computational Science. 2022.

[11] Wu, E., Zhang, P., Lu, T., Gu, H., Gu, N., *Behavior prediction using an improved hidden markov model to support people with disabilities in smart homes*, w: *2016 IEEE 20th International Conference on Computer Supported Cooperative Work in Design (CSCWD)* (2016), str. 560–565.

SPIS RYSUNKÓW

SPIS LISTINGÓW

SPIS TABEL

Dodatki

A. DODATEK 1

Nulla ac nisl. Nullam urna nulla, ullamcorper in, interdum sit amet, gravida ut, risus. Aenean ac enim. In luctus. Phasellus eu quam vitae turpis viverra pellentesque. Duis feugiat felis ut enim. Phasellus pharetra, sem id porttitor sodales, magna nunc aliquet nibh, nec blandit nisl mauris at pede. Suspendisse risus risus, lobortis eget, semper at, imperdiet sit amet, quam. Quisque scelerisque dapibus nibh. Nam enim. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Nunc ut metus. Ut metus justo, auctor at, ultrices eu, sagittis ut, purus. Aliquam aliquam.