



FAKULTA APLIKOVANÝCH VĚD
ZÁPADOČESKÉ UNIVERZITY
V PLZNI

KATEDRA
KYBERNETIKY

BAKALÁŘSKÁ PRÁCE

Automatické ovládání žaluzií s využitím strojového učení

Autor:
Vojtěch Breník

Vedoucí práce:
Ing. Martin Bulín, M.Sc.

20. května 2022

Prohlášení

Předkládám tímto k posouzení a obhajobě bakalářskou práci zpracovanou na závěr studia na Fakultě aplikovaných věd Západočeské univerzity v Plzni.

Prohlašuji, že jsem bakalářskou práci vypracoval samostatně a výhradně s použitím odborné literatury a pramenů, jejichž úplný seznam je její součástí.

V Plzni dne 23. května 2022

ZÁPADOČESKÁ UNIVERZITA

Fakulta aplikovaných věd

Katedra kybernetiky

Abstrakt

Automatické ovládání žaluzií s využitím strojového učení

Vojtěch Breník

Automatic blinds control using machine learning

Poděkování

Rád bych poděkoval svému vedoucímu bakalářské práce Ing. Martinovi Bu-línovi, MSc. za odborné vedení, cenné rady a pomoc při vypracování této práce. Dále děkuji Petru Vildovi ze Stanice mladých techniků za poskytnutí prostoru a vybavení pro výrobu plošných spojů a IoT Labu při Centru informatizace a výpočetní techniky ZČU a Miloši Mulačovi za poskytnutí vybavení, materiálu a za pomoc při tisku krabičky na měřicí zařízení.

Obsah

Abstrakt	ii
1 Úvod	1
1.1 Internet věcí (IoT) a domácí automatizace	2
1.2 Umělé neuronové sítě (NS)	2
1.2.1 Umělý neuron	2
1.2.2 Aktivační funkce	3
1.2.3 Dopředná neuronová síť	5
1.2.4 Rekurentní neuronová síť, LSTM	6
1.3 Venkovní žaluzie s pohonem	7
2 Definice úlohy	9
2.1 Vstupy systému	9
2.2 Výstupy systému	11
3 Hardwarové komponenty	12
3.1 Zařízení pro měření teploty vzduchu a intenzity osvětlení	12
3.1.1 Zapojení	13
3.1.2 Deska plošných spojů (DPS)	14
3.1.3 Modul ESP-12E s MCU a WiFi	16
3.1.4 Teplotní senzor DS18B20	16
3.1.5 Senzor intenzity osvětlení TSL2591	16
4 Sběr dat a komunikace komponent	17
4.1 Použití komunikačního protokolu MQTT	18
4.2 Komunikace s pohonem žaluzie	18
4.3 Využití služby OpenWeather	20
4.4 Databáze	20
4.5 Webový server pro komunikaci s GUI	21
5 Automatické ovládání žaluzií	23
5.1 Příprava dat pro strojové učení	23
5.2 Regresor založený na pravidlech (If-else)	24
5.3 Dopředná neuronová síť jako regresor (FFNN)	26
5.4 Rekurentní neuronová síť jako regresor (LSTM)	26
5.5 Přetrénování neuronových sítí (retraining)	27
6 Grafické uživatelské rozhraní	28
6.1 Stránka Data	28
6.2 Stránka Simulator	29
6.3 Stránka Control	30
6.4 Stránka Live	31

7	Vyhodnocení	33
7.1	Vyhodnocení sběru dat a spolehlivosti	33
7.2	Volba struktury NS a parametrů učení	34
7.3	Vliv jednotlivých příznaků na predikci	36
7.4	Porovnání regresorů	39
7.5	Vyhodnocení vlivu retrainingu	40
8	Diskuze	42
8.1	Možnosti reálného nasazení systému	43
9	Závěr	44
	Bibliografie	45
	A1 Prohledávání parametrů konstrukce a trénování NS	47

Seznam obrázků

1.1	Grafy aktivačních funkcí	4
1.2	Dopředná neuronová síť	6
1.3	LSTM jednotka	7
1.4	Venkovní žaluzie	8
3.1	Skutečné umístění měřicího zařízení	13
3.2	Schéma zapojení měřicích zařízení	14
3.3	Návrh DPS měřicích zařízení	14
3.4	Výroba DPS	15
4.1	Schéma komunikace komponent	17
4.2	Schéma endpointů a přenášených struktur	22
6.1	Ovládací panel v záhlaví GUI	28
6.2	Stránka Data v GUI	29
6.3	Stránka Simulator v GUI	29
6.4	Vizualizace výstupu regresoru	30
6.5	Stránka Control v GUI	30
6.6	Stránka Live v GUI	31
7.1	Graf sesbíraných dat (příklad)	33
7.2	Graf druhu sesbíraných vzorků a výpadků	34
7.3	Boxplot FFNN	35
7.4	Boxplot LSTM	35
7.5	Permutační důležitost příznaků FFNN	37
7.6	Permutační důležitost příznaků LSTM	37
7.7	Důležitost příznaků FFNN (nulování)	38
7.8	Důležitost příznaků LSTM (nulování)	38
7.9	Porovnání predikce regresorů	39
7.10	Porovnání predikce FFNN (retraining)	41
7.11	Porovnání predikce LSTM (retraining)	41

Seznam tabulek

2.1	Příznaky pro automatické ovládání žaluzie	11
4.1	Příznaky a jejich MQTT téma	18
5.1	Prohledávání parametry dopředné neuronové sítě	26
5.2	Prohledávání parametry rekurentní neuronové sítě	27
7.1	Výpadky	34
7.2	Porovnání regresorů	39
7.3	Retraining FFNN	40
7.4	Retraining LSTM	40
A1.1	Výsledky prohledávání parametrů LSTM	47
A1.2	Výsledky prohledávání parametrů FFNN	48

Seznam zkratек

API Application Programming Interface (rozhraní pro programování aplikací). 1, 18, 33, 42

DPS Deska plošných spojů. iv, vi, 12–14

FFNN Feedforward Neural Network (dopředná neuronová síť). vi, 27, 35

GUI Graphical User Interface (grafické uživatelské rozhraní). iv, 12, 19–21, 27, 28, 30, 31

IoT Internet of Things (internet věcí). iv, 2, 7

LSTM Long Short-Term Memory (druh rekurentní neuronové sítě). vi, 6, 7, 26, 27, 35

MCU Microcontroller Unit (jednočipový počítač). iv, 13, 14, 16, 42

MSE Mean Squared Error (střední kvadratická odchylka). 23, 26, 34, 36, 39, 40, 42, 43

NS Neuronová síť. iv, v, 2, 9, 21, 23, 24, 26, 27, 34, 36, 38, 42–44, 47, 48

ReLU Aktivační funkce Rectified Linear Unit. 3, 4

URL Uniform Resource Locator. 19

1 Úvod

Stínění oken v pozemních stavbách určených k bydlení se využívá k regulaci teploty a zajištění soukromí obyvatel. Jednou z možností stínění jsou venkovní žaluzie osazené pohonem na dálkové ovládání. Jejich uživatel může měnit výšku vytažení žaluzie a sklon lamel a ovlivňovat tím množství záření, které skrz okno prochází. Pomocí vzdáleného ovládání pak lze žaluzie řídit automaticky (Lubinová, 2013).

Komplexní systém může generovat akční zásahy pro řízení žaluzií na základě různých vstupních informací (datum a čas, měření veličin pomocí senzorů) bez účasti uživatele za účelem zajištění jeho komfortu a úspory prostředků vynaložených na regulaci teploty v interiéru (vytápění a chlazení). Sběr informací na základě kterých se má rozhodovat o nastavení žaluzie i samotné rozhodování musí probíhat v reálném čase.

Tato práce se zabývá návrhem systému pro automatické ovládání žaluzií včetně návrhu a realizace měřicích zařízení použitelných pro měření veličin užitečných při rozhodování systému a možnostmi využití strojového učení k ovládání žaluzií v reálném čase. Zkoumá časový vývoj použitých algoritmů při opakovaném učení na postupně sbíraných datech s ohledem na měnící se požadavky uživatele a měnící se podmínky v průběhu roku. Cíle této práce jsou následující:

1. Určit vhodné vstupní veličiny (příznaky) a výstupní veličiny pro automatické rozhodování o stavu žaluzie.
2. Zkonstruovat měřicí zařízení pro měřitelné příznaky a odesílat jejich hodnoty na server.
3. Zajistit komunikaci s vybranými žaluziemi pomocí poskytovaného API¹.
4. Zajistit komunikaci se zdroji všech příznaků a data ukládat pro pozdější použití.
5. Navrhnout základní rozhodovací systém založený na pravidlech („baseline“).
6. Navrhnout model neuronové sítě a porovnat jeho použitelnost s baseline systémem.

V práci se využívá skutečná venkovní žaluzie s hliníkovými lamelami a pohonem od firmy Somfy. Vzdálenou komunikaci s ním zajišťuje řídicí jednotka Tahoma připojená k internetu, která komunikuje se servery výrobce, jejichž prostřednictvím je možné žaluzii ovládat a zjistit její aktuální stav. Žaluzie se nachází před oknem pokoje autora této práce v rodinném domě v obci vzdálené asi 12 km severovýchodně od Plzně. Uvnitř tohoto pokoje a na střeše daného domu také probíhají všechna měření.

¹Application Programming Interface (rozhraní pro programování aplikací)

1.1 Internet věcí (IoT) a domácí automatizace

Internet věcí je síť fyzických objektů, jako jsou různá zařízení, vozidla, nástroje a budovy se zabudovanými obvody (zejména elektronickými), řídicím softwarem, senzory, aktuatory a síťovým rozhraním, které umožňuje vzájemnou výměnu informací mezi objekty. Objekty tak mohou být vzdáleně monitorovány nebo řízeny, což vytváří příležitosti pro zapojení věcí z reálného světa do počítačových systémů a jejich vzájemnou spolupráci (Gokhale, Bhat a Bhat, 2018).

Výrobci objektů, které lze zapojit do IoT, často neumožňují vzájemnou komunikaci se všemi zařízeními (zejména se zařízeními jiných výrobců) a také využívají proprietárních komunikačních protokolů. To omezuje spolupráci mezi objekty a jejich efektivní řízení (Jaihar et al., 2020).

Systém domácí automatizace zajišťuje ovládání některých zařízení uvnitř nebo v okolí domácnosti bez účasti jejich obyvatel. Může řídit a monitorovat osvětlení, zabezpečení, vytápění, větrání nebo také právě stínění, což může obyvatelům zjednodušit život v domácnosti, snížit spotřebu některých zdrojů a obyvatele včas upozornit na vzniklé anomálie. V takovém systému lze s výhodou využít princip IoT a informací a služeb, které jednotlivá zapojená zařízení poskytují (Popa et al., 2018; Jaihar et al., 2020).

Tradiční systémy domácí automatizace umožňují ruční nastavení jejich chování, kdy na základě různých událostí (změna hodnoty senzoru, časový okařík, akce uživatele) vykonávají akce (Apple Inc., 2022; Home Assistant, 2022). Nevýhodou tohoto přístupu je právě ruční nastavování a časová neměnnost systému, který se tak nemůže přizpůsobovat novým okolnostem (Popa et al., 2018).

Obě zmíněné nevýhody tradičního postupu může řešit přístup založený na datech, který se při jejich souvislém sběru může přizpůsobovat a ruční nastavení časových i jiných závislostí vzejdou z dat samy. K tomuto účelu lze efektivně využít metody strojového učení (Popa et al., 2018). V této práci je prozkoumáno možné využití umělých neuronových sítí v aplikaci domácí automatizace pro automatické řízení venkovních žaluzií.

1.2 Umělé neuronové sítě (NS)

Umělé neuronové sítě jsou stežejní součástí navrhovaného systému, ve kterém se používají pro odhad správného nastavení žaluzie. Skládají se ze vzájemně propojených umělých neuronů obvykle uspořádaných do vrstev. Samotné neurony se také někdy považují za nejjednodušší neuronové sítě. Každý z neuronů má množinu parametrů, které se pomocí optimalizačních algoritmů nastavují tak, aby podle zvoleného kritéria výstup neuronové sítě co nejlépe odpovídal předloženým trénovacím datům.

1.2.1 Umělý neuron

Bishop ve své knize (1995) shrnuje vývoj od matematického modelu neuronu McCullocha a Pittse (1943) až po perceptron (Rosenblatt, 1958), který je v některých neuronových sítích využíván dodnes. V obou zmíněných případech

se jedná o matematickou funkci, která zobrazuje vstupy x_1, x_2, \dots, x_N , kde N je počet vstupů neuronu, na výstup y .

McCullochův a Pittsův model neuronu pracuje v diskrétních úrovních vstupů a výstupu: $x_i, y \in \{0, 1\}$. Vstupy x_i jsou postupně sečteny. Přesáhne-li součet nastavený práh Θ , neuron je aktivován a jeho výstup $y = 1$. V opačném případě je $y = 0$. Výstup takového neuronu tedy můžeme vyjádřit jako

$$y = g\left(\sum_{i=1}^N x_i\right),$$

kde $g(a)$ je v tomto případě Heavisidova funkce (sekce 1.2.2). Funkce $g(a)$ se obecně nazývá *aktivační* a v jiných modelech neuronu se využívají různé i jinak definované funkce (Bishop, 1995; příklady jsou uvedené v sekci 1.2.2).

Perceptron (Rosenblatt, 1958) je v určitém slova smyslu zobecněním McCullochova a Pittsova modelu neuronu. Jako vstupy x_i uvažuje hodnoty z \mathbb{R} a kromě prahu Θ má navíc váhy vstupů $w_i \in \mathbb{R}$. Těmi jsou postupně vynásobeny příslušné vstupy x_i a výsledky jsou sečteny. Přesáhne-li tento součet nastavený práh Θ , perceptron je aktivován a jeho výstup $y = 1$. V opačném případě je $y = 0$. Označíme $\mathbf{x} = [x_0, x_1, x_2, \dots, x_N]^T$ a $\mathbf{w} = [w_0, w_1, w_2, \dots, w_N]^T$, kde $x_0 = 1$ je přidaný myšlený vstup, který nabývá vždy hodnoty 1, a $w_0 = \Theta$ je jeho odpovídající váha. Součet se tím zvýší o práh Θ a pro výstup y perceptronu pak platí:

$$y = g(\mathbf{w}^T \mathbf{x}),$$

kde $g(a)$ je Heavisidova funkce. Parametr w_0 se nazývá *bias* a reprezentuje práh Θ pro aktivaci perceptronu (Bishop, 1995).

Rosenblatt také navrhl algoritmus učení perceptronu s učitelem pro klasifikaci, které postupnou úpravou parametrů zajistil u lineárně separabilních množin správnou klasifikaci trénovacích dat (Rosenblatt, 1958).

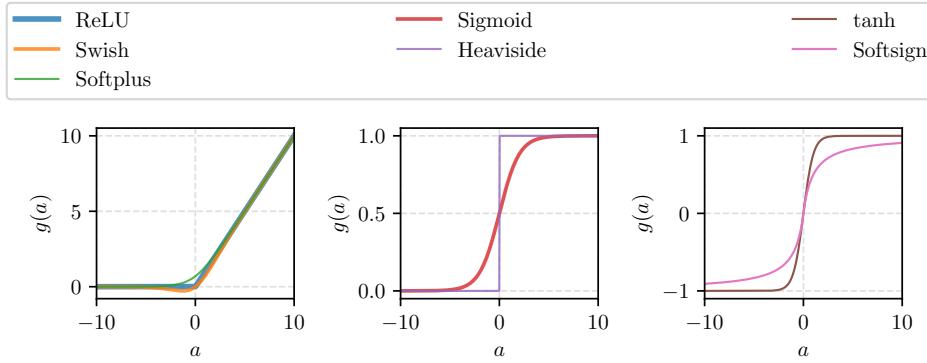
Pokud se jako funkce $g(a)$ místo Heavisidovy funkce použije jiná aktivační funkce (sekce 1.2.2), která je diferencovatelná vzhledem k vektoru vah a biasu \mathbf{w} , lze pro učení perceptronu použít algoritmus *backpropagation* (Linnainmaa, 1970) založený na optimalizační metodě *gradient descent* (Bishop, 1995). Ta byla původně navržena jako obecná metoda pro řešení soustav rovnic (Cauchy, 1847).

1.2.2 Aktivační funkce

Aktivační funkce se v neuronových sítích využívají k vyhodnocení výstupu (aktivace) y na základě váženého součtu vstupů (včetně biasu). Podle svého průběhu mohou mít využití v různých úlohách a mají také vliv na průběh učení. (Nwankpa et al., 2018) Níže jsou uvedené některé aktivační funkce, jejich definice a vlastnosti. Jejich průběhy $g(a)$ pro $a \in \langle -10, 10 \rangle$ jsou v grafech na obrázku 1.1.

Aktivační funkce Rectified Linear Unit (ReLU)

Funkce ReLU (Nair a Hinton, 2010) je široce používána v aplikacích hlubokého učení a poskytuje výsledky na špičkové úrovni. Ve srovnání se sigmoidou



OBRÁZEK 1.1: Grafy vybraných aktivačních funkcí

a hyperbolickým tangensem dává v hlubokém učení lepší výsledky a zajišťuje větší míru zobecnění. (Nwankpa et al., 2018) Definována je takto:

$$g(a) = \max(0, a) = \begin{cases} a, & \text{jestliže } a \geq 0 \\ 0, & \text{jestliže } a < 0. \end{cases}$$

ReLU se obvykle používá ve skrytých vrstvách neuronových sítí a je doplněna jinou aktivační funkcí ve výstupní vrstvě. Velkou výhodou je malá výpočetní náročnost výpočtu funkční hodnoty, což má pozitivní vliv na rychlosť trénování. (Nwankpa et al., 2018)

Aktivační funkce Swish

Funkce Swish (Ramachandran, Zoph a Le, 2017) v některých ohledech překonává ReLU. Průběhem se výrazněji liší jen v okolí 0, kde má ostré lokální minimum. Narození od ReLU je hladká a nemonotonní. Nemonotónnost ji také odlišuje od většiny aktivačních funkcí. Stejně jako ReLU je omezená zdola a neomezená shora. Je definována jako:

$$g(a) = a \cdot \sigma(\beta a),$$

kde $\sigma(z) = \frac{1}{1+e^{-z}}$ je sigmoida a β je volitelný nebo trénovatelný parametr.

Ramachandran, Zoph a Le ukázali, že na naprosté většině zkoušených úloh je alespoň tak dobrá jako původní aktivační funkce (ve většině případů dokonce lepší). Její podobnost s ReLU umožňuje snadné nahrazení v již existujících modelech

Aktivační funkce softplus

Funkce softplus (Dugas et al., 2000) je hladkou approximací ReLU s nenulovými gradienty. Je definována jako:

$$g(a) = \ln(1 + e^a).$$

Je hladká, ostře monotonné a zdola omezená. Při jejím použití modely konvergují rychleji než s využitím ReLU a sigmoidy.

Aktivační funkce sigmoida

Sigmoida je speciálním případem logistické funkce, jedná se o hladkou nelineární aktivační funkci, omezenou shora i zdola. Její definice je:

$$g(a) = \sigma(a) = \frac{1}{1 + e^{-a}}.$$

Využívá se ve výstupních vrstvách neuronových sítí, mimo jiné v případě výstupů založených na pravděpodobnosti. (Nwankpa et al., 2018)

Heavisidova aktivační funkce

Heavisidova funkce se využívala v původních modelech neuronu pro svou podobnost s biologickou předlohou. (Bishop, 1995; McCulloch a Pitts, 1943) Je definována takto:

$$g(a) = \begin{cases} 0 & \text{jestliže } a < 0; \\ 1 & \text{jestliže } a \geq 1. \end{cases}$$

V současných aplikacích, které se trénují pomocí gradientních metod ji ale nelze použít, protože není diferencovatelná. Je limitním případem logistické funkce, je tedy omezená. Její obor hodnot je dvouprvková množina $\{0, 1\}$.

Aktivační funkce hyperbolický tangens

Hyperbolický tangens je hladká, monotónní, omezená funkce definovaná jako:

$$g(a) = \frac{e^a - e^{-a}}{e^a + e^{-a}}.$$

Její střední hodnota leží v 0. Využívá se v některých rekurentních neuronových sítích v oblastech jako zpracování přirozeného jazyka a rozpoznávání řeči. (Nwankpa et al., 2018)

Aktivační funkce softsign

Softsign je další z aktivačních funkcí. Je podobná jako hyperbolický tangens, ale konverguje pomaleji. Definuje se jako:

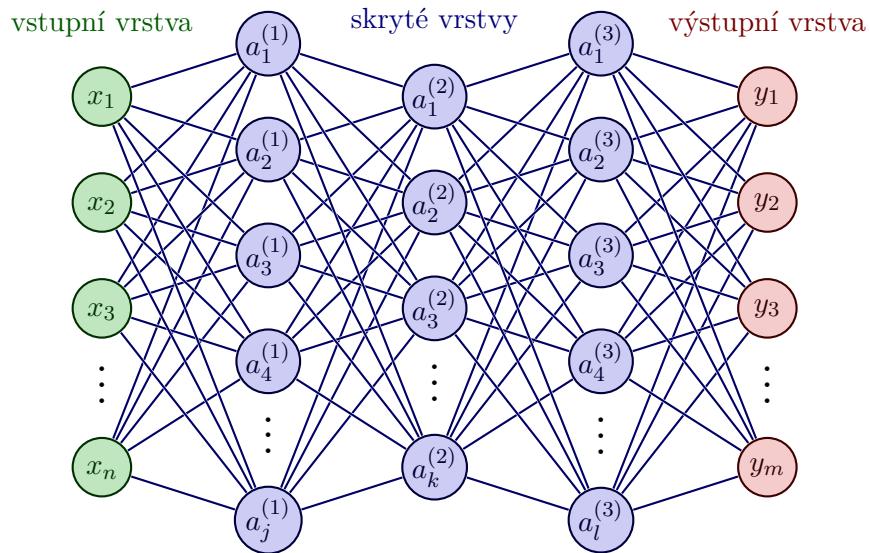
$$g(a) = \frac{a}{|a| + 1}$$

Používána je zejména v regresních problémech. (Nwankpa et al., 2018)

1.2.3 Dopředná neuronová síť

Dopředná neuronová síť, někdy také nazývaná vícevrstvý perceptron (přestože se používají jiné aktivační funkce než u původního perceptronu), je zřetězení umělých neuronů tak, že aktivace některých neuronů mohou být vstupem jiných. Aby síť byla dopředná, musí být splněna podmínka, že nemusí obsahovat žádnou zpětnovazební smyčku, tedy že výstup neuronu nesmí být vstupem některého z předchozích neuronů, na jejichž aktivaci tento výstup záleží. Přestože to není podmínkou, neurony se obvykle uspořádávají do vrstev tak, že aktivace neuronů v dané vrstvě slouží jako vstupy neuronů ve

vrstvě další. Výstup pak lze vyhodnocovat postupně po vrstvách. Poslední z vrstev se nazývá výstupní, ostatní vrstvy neuronů jsou tzv. skryté. Vstupy, přestože se nejedná o neurony, jsou označovány jako vstupní vrstva, ale obvykle se nezapočítávají do počtu vrstev sítě. Jako počet vrstev se označuje součet počtu skrytých vrstev a 1 (výstupní vrstva). Počet neuronů v jednotlivých vrstvách je libovolný a záleží na návrháři sítě (Bishop, 1995). Příklad 4vrstvé sítě je na obrázku 1.2. Obvykle se propojují všechny neurony z dané vrstvy s každým z neuronů ve vrstvě následující, obecně ale mohou některá tato spojení chybět.



OBRÁZEK 1.2: Příklad dopředné neuronové sítě o 3 skrytých vrstvách

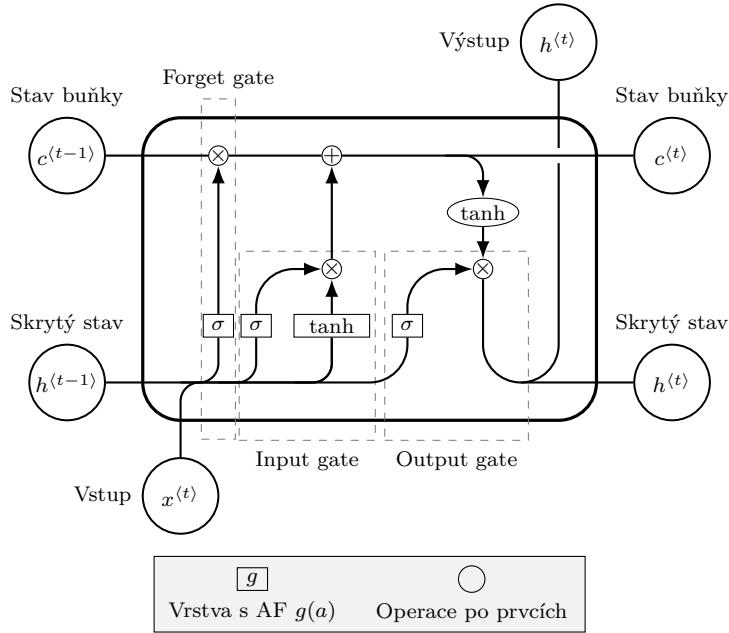
1.2.4 Rekurentní neuronová síť, LSTM

Rekurentní neuronová síť se od dopředné liší tím, že obsahuje smyčky, kterými se přenáší informace mezi diskrétními časovými okamžiky, ve kterých síť pracuje. Takto přenášeným informacím se také říká skryté stavy. Vstupem jednotlivých vrstev je tak kromě běžného vstupu také jejich výstup v předchozím časovém okamžiku. Výstup rekurentní sítě se vyhodnocuje postupně v jednotlivých okamžicích pro časovou posloupnost jednotlivých vstupů.

Long Short-Term Memory (LSTM) je speciální architektura rekurentní neuronové sítě, která ve spojení s vhodným trénovacím algoritmem řeší problém mizejícího gradientu (*gradient vanishing problem*), který se vyskytoval při použití základní rekurentní neuronové sítě (Rumelhart, Hinton a Williams, 1986). Hochreiter a Schmidhuber navrhli použití LSTM jendnotky, ve které se pomocí 2 hradel (vstupní a výstupní) řídí tok informací mezi jednotkou a okolím a mezi časovými okamžiky. Díky tomu je možné sestavit síť, která umí v datech překlenout různě dlouhé časovové intervaly a reagovat na vzorce patrné ve vstupech v průběhu času. Také je možné takovou síť efektivně trénovat (Hochreiter a Schmidhuber, 1997).

LSTM jednotka má několik vstupů a výstupů (všechny jsou znázorněny na obrázku 1.3). Jednotlivé vstupy jsou:

1. běžný vstup $x^{(t)}$ v daném časovém okamžiku t ;



OBRÁZEK 1.3: Schéma Long Short-Term Memory jednotky.
AF je zkratka aktivační funkce.

2. skrytý stav $h^{(t-1)}$ z předchozího časového okamžiku $t - 1$ a
3. stav buňky $c^{(t-1)}$ z předchozího časového okamžiku $t - 1$.

Výstupy jsou pak:

1. běžný výstup a zároveň skrytý stav $h^{(t)}$ v časovém okamžiku t a
2. stav buňky $c^{(t)}$ v daném časovém okamžiku t .

Klíčovou součástí LSTM jednotky je LSTM buňka, která slouží jako paměť a umožňuje selektivně uchovávat informace napříč časovými okamžiky. Jedná se o přímé propojení v rámci jedné jednotky s jednokrokovým zpožděním, její stav je označen c .

Později bylo navrženo přidání 3. hradla, tzv. *forget gate*, které řídí uchovávání informace v LSTM buňce, což mělo pozitivní vliv při aplikaci LSTM na souvislé posloupnosti. (Gers, Schmidhuber a Cummins, 2000) Struktura LSTM jednotky se všemi třemi hradly je na obrázku 1.3.

1.3 Venkovní žaluzie s pohonem

Venkovní žaluzie jsou stínící prvek umístěný z vnější strany budovy (příklad instalovaných žaluzí na domě je na obrázku 1.4) a jsou tvořeny jednotlivými lamelami vzájemně spojenými tzv. žebříčkem. Lamely se mohou pohybovat jednak ve svislém směru při zatáhování a vytahování žaluzie, jednak mohou rotovat kolem své podélné osy - sklápění lamel. Venkovní žaluzie mohou být ovládány několika způsoby - šňůrou, klikou nebo motoricky. Poslední zmíněné ovládání zvyšuje uživatelský komfort, protože není nutné se žaluziemi fyzicky manipulovat, a je také možné realizovat automatické funkce (Lubinová, 2013). Motorický pohon žaluzie je možné zapojit do IoT a využívat ho tak v různých systémech.



OBRÁZEK 1.4: Příklad venkovních žaluzií instalovaných na domě

Stav žaluzie lze jednoznačně popsat pomocí výšky vytažení poslední lamely a úhlu, který lamely svírají se svislou rovinou. V případě žaluzie použité v této práci jejich výrobce Somfy ve svém systému pro ovládání využívá proměnné se stejným významem, ale s odlišnou škálou. Jedná se o míru zatažení, kde 100 značí zataženou žaluzii a 0 vytaženou, a o míru sklopení, kde 100 značí zcela sklopené lamely a 0 vodorovné.

2 Definice úlohy

Hlavním cílem práce je navržení systému pro automatické řízení žaluzií. V této kapitole je definována úloha automatického řízení, kterou tento systém bude řešit.

Jak již bylo zmíněno v sekci 1.3, žaluzie mohou být vytažené do různé výšky a jejich lamely mohou svírat různé úhly se svislou rovinou. Jejich stav tak lze obecně vyjádřit pomocí dvou proměnných, které je nutné v systému odhadovat (tzv. *target*). V této práci jsou tyto proměnné označované jako *position* a *tilt* a jejich bližší význam je uveden v sekci 2.2.

Za běžných okolností stav žaluzie určuje její uživatel, jehož chování má systém napodobovat. V sekci 2.1 jsou vyjádřeny předpoklady o rozhodování uživatele, ze kterých vzešlo celkem 15 proměnných, které budou systému sloužit jako vstup (tzv. *příznaky*).

Vstupy uspořádáme do příznakového vektoru $\mathbf{x} = [x_1, x_2, \dots, x_{15}]^T$, kde x_i je příznak uvedený v i -té řádku tabulky 2.1. Podobně výstupy uspořádáme do výstupního vektoru $\mathbf{y} = [\text{position}, \text{tilt}]$.

Poté lze definovat úlohu tak, že na základě obecné p -prvkové posloupnosti příznakových vektorů $(\mathbf{x}^{(t)})_{t=0}^{p-1}$ má být generován odhad $\hat{\mathbf{y}}$ výstupního vektoru tak, aby nastavení žaluzie dle výstupního vektoru odpovídalo preferencím uživatele. Jedná se tedy o úlohu regrese. Parametr p závisí na zvoleném řešení a umožňuje k odhadu využít historický kontext příznaků.

V kapitole 5 jsou navržena 3 různá řešení této úlohy. První z nich je ručně vytvořený systém založený na pravidlech, který slouží pro porovnání (baseline). Další 2 navržená řešení využívají NS jako regresory. S využitím příkladů p -prvkové posloupnosti příznakových vektorů $(\mathbf{x}^{(t)})_{t=0}^{p-1}$ v časových okamžicích t a odpovídajího výstupního vektoru $\mathbf{y}^{(t)}$ v čase $t = p - 1$ (trénovací data) se natrénuje neuronová síť, která tak řeší výše definovanou úlohu. Pro dopřednou NS je $p = 1$, pro rekurentní je obecně $p > 1$, v této práci je uvažováno $p = 64$. Sběr reálných trénovacích dat je popsán v kapitole 4.

2.1 Vstupy systému

Při návrhu systému se vycházelo z předpokladu, že uživatel rozhoduje o ručním řízení žaluzí na základě odhadu, případně měření, některých fyzikálních veličin. Pro nahrazení rozhodování uživatele automatickým řízením je nutné stanovit množinu veličin, které se budou v pravidelných intervalech měřit a na základě jejich hodnot automaticky generovat vhodné řízení pomocí modelů získaných strojovým učením. Tato sekce se zabývá volbou jednotlivých prvků příznakového vektoru, tedy příznaků.

Uživatel může nastavením žaluzie sledovat různé cíle, hlavní 4 byly identifikovány takto:

1. Za šera a tmy mají být žaluzie zatažené pro zajištění soukromí uvnitř místnosti při použití umělého osvětlení a zabránění vniknutí světla od projíždějících vozidel v době spánku uživatele v místnosti.
2. Při slunečných dnech, kdy venkovní vzduch dosahuje teplot blízkých pokojové teplotě nebo je převyšuje, mají žaluzie bránit průniku přímého slunecního záření do místnosti.
3. Při teplotách nižších, kdy je nutné interiér vytápět, by žaluzie naopak měly umožnit maximální průchod záření oknem, aby se tak místnost vytápěla a nebylo nutné využívat zbytečně vysoké množství energie na běžné vytápění.
4. V případě silného větru by měly být žaluzie vytažené, aby se zabránilo jejich poškození.

Z bodu 1 vyplývá, že uživatel sleduje intenzitu osvětlení exteriéru, orientačně je možné ji určit podle času a data, ale silně závisí také na počasí (zejména oblačnosti) a protože jsou na trhu dostupné senzory, které komunikují pomocí standardních sběrnic, je vhodnější ji přímo měřit. Použití umělého osvětlení uvnitř místnosti jimi lze detektovat také. Vnitřní a vnější intenzita osvětlení tedy byly zvoleny jako příznaky.

Body 2 a 3 zmiňují vliv teploty venkovního vzduchu na manuální ovládání uživatelem. Dá se ale předpokládat, že pokud by teplota vzduchu uvnitř místnosti byla podle uživatele příliš nízká, nezastiňoval by okno a nechal místnost vytápět i slunečním zářením i přes vysokou vnější teplotu. Proto byly dalšími příznaky zvoleny teplota vzduchu uvnitř a teplota vzduchu venku. Kromě toho uživatel může sledovat předpověď počasí a na jejím základě vyhodnotit, že je vhodné stínit dříve než teplota v průběhu dne vzroste, protože by už nemusel mít možnost vyhřátý interiér ochladit venkovním vzduchem. Jako příznaky se tedy zvolily předpovězené teploty na hodinu, 2 a 3 dopředu, předpověď nejvyšší denní teploty a odhadovaný stav počasí ve formě číselného kódu. Okamžité teploty byly měřeny pomocí digitálního teploměru (sekce 3.1), předpovědi a odhad se získávaly z internetu pomocí OpenWeather API (sekce 4.3).

Z bodu 4 vyplývá vliv rychlosti větru na žaluzie, proto byla zvolena jako další příznak. Manuální vyhodnocení uživatelem probíhá obvykle subjektivně podle hluku způsobeného kmitáním lamel žaluzií v boční vodicí drážce. Na něj může kromě rychlosti mít vliv i směr větru, kvůli různímu se obtékaní domu proudícím vzduchem. Obě veličiny byly zvoleny jako příznaky, jejich měření bylo vyhodnoceno jako příliš složité a nákladné a odhad jejich hodnot se tedy získávají stejně jako předpovědi teploty (sekce 4.3). Systém žaluzií od firmy Somfy také obsahuje vlastní anemometr, který žaluzie v případě silného větru zadá žaluziím příkaz k vytažení, uživatel ale může být opatrny a žaluzie vytahovat již při nižších rychlostech větru.

Lidé dále mohou mít pravidelné zvyky, které ovlivňují nastavení žaluzií. Z toho důvodu byly jako příznaky zvoleny některé časové údaje: počet uplynulých sekund v rámci dne, počet uplynulých dnů v rámci týdne a pořadí dne

v roce. Poslední z nich souvisí také s obvyklým počasím, které může ovlivňovat nastavení žaluzií. V případě nepřítomnosti uživatele v domácnosti nemusí nastavení žaluzií přesně odpovídat obvyklému záměru uživatele, proto je jedním z příznaků také jeho přítomnost v domácnosti. Ta se vyhodnocuje v systému domácí automatizace na základě periodicky přenášené polohy uživatelského telefonu.

Přehled všech 15 příznaků, jejich označení, jednotky a očekávané nejnižší a nejvyšší hodnoty je uveden v tabulce 2.1

Příznak	Označení	Množina
Den v roce	year_day	$\langle 1, 365 \rangle$
Den v týdnu	week_day	$\langle 0, 6 \rangle$
Denní čas [s]	day_secs	$\langle 1, 86400 \rangle$
Uživatel je doma	home	$\{0, 1\}$
Vnitřní teplota [$^{\circ}\text{C}$]	temp_in	$\langle 10, 35 \rangle$
Venkovní teplota [$^{\circ}\text{C}$]	temp_out	$\langle -20, 40 \rangle$
Osvětlení uvnitř [lux]	lum_in	$\langle 0, 2000 \rangle$
Osvětlení venku [lux]	lum_out	$\langle 0, 60000 \rangle$
OWM - maximální denní teplota [$^{\circ}\text{C}$]	owm_temp_max	$\langle -20, 40 \rangle$
OWM - předpověď teploty za 1h [$^{\circ}\text{C}$]	owm_temp_1h	$\langle -20, 40 \rangle$
OWM - předpověď teploty za 2h [$^{\circ}\text{C}$]	owm_temp_2h	$\langle -20, 40 \rangle$
OWM - předpověď teploty za 3h [$^{\circ}\text{C}$]	owm_temp_3h	$\langle -20, 40 \rangle$
OWM - kód stavu počasí	owm_code	$\langle 200, 804 \rangle$
OWM - rychlosť větru [$\frac{\text{m}}{\text{s}}$]	owm_wind_speed	$\langle 0, 50 \rangle$
OWM - směr větru [$^{\circ}$]	owm_wind_heading	$\langle 0, 359 \rangle$

TABULKA 2.1: 15 příznaků použitých pro automatické ovládání žaluzie, jejich označení v systému a množiny očekávaných hodnot.

Po natrénování použitých algoritmů (kapitola 5) byla vyhodnocena důležitost jednotlivých příznaků pomocí algoritmu *permutation feature importance* (Breiman, 2001). Výsledky jsou uvedeny v sekci 7.3.

2.2 Výstupy systému

Výstupem systému je dvouprvkový vektor \hat{y} , který je odhadem stavu žaluzie v závislosti na předloženém příznakovém vektoru (nebo posloupnosti takových vektorů). Stav žaluzie je v systému vyjádřen jako míra vytažení žaluzie (0 - zataženo, 100 - vytaženo; position) a míra otevření lamel (0 - zcela sklopené, 100 - vodorovné lamely; tilt).

3 Hardwarové komponenty

K provozování určitých součástí systému a měření 4 příznaků je nutné použít fyzické technické vybavení, označované také jako hardware. Tato kapitola popisuje jeho funkční celky složené z jednotlivých použitých součástí a zmíňuje využití hardwaru, které je blíže specifikováno v příslušných kapitolách (kapitoly 4 a 6).

K měření příznakových veličin (sekce 2.1) intenzity osvětlení a teploty vzduchu ve vnějším a vnitřním prostředí se využívá 2 desek plošných spojů (DPS) vlastní konstrukce osazených modulem ESP-12E od společnosti Ai-Thinker (sekce 3.1). Na jednodeskovém počítači Raspberry Pi 2B, běží následující SW součásti:

- skript `mqttDataCollector.py`, který pomocí MQTT komunikuje s ostatními uzly a sbírá z nich data;
- skript `broker2mongo.py`, který sesbíraná data ukládá do databáze MongoDB;
- MQTT broker¹ Mosquitto, ke kterému jsou připojeny všechny zdroje dat.

Posledním využívaným celkem je workstation označovaná jako KKY-PC, která:

- Hostuje databázový server MongoDB.
- Hostuje webový server vlastního grafického uživatelského rozhraní (GUI).
- Provozuje vlastní REST a WebSocket backend implementovaný ve frameworku Tornado.
- Sloužila k trénování neuronových sítí regresorů.

Jedná se o výkonný osobní počítač s 6 jádrovým 64 bitovým procesorem Intel Core i7-7800X, 64 GiB operační paměti a grafickou kartou NVIDIA GTX 1080. Pohání ji operační systém Ubuntu 20.04.3 a po dobu vývoje běžela na Katedře kybernetiky Západočeské univerzity v Plzni.

3.1 Zařízení pro měření teploty vzduchu a intenzity osvětlení

Čtyři z příznaků pro regresory tvoří teplota vzduchu a intenzita osvětlení, obojí v interiéru i exteriéru (sekce 2.1). Obě veličiny byly měřeny periodicky každých 5 minut a v případě změny stavu žaluzie uživatelem pomocí

¹MQTT broker je centrální uzel, který zprostředkovává přenos zpráv od klienta, který zprávu odeslal, ke klientům, kteří se přihlásili k jejich odběru. Více v sekci 4.1.



OBRÁZEK 3.1: Skutečné umístění měřicího zařízení na střeše rodinného domu.

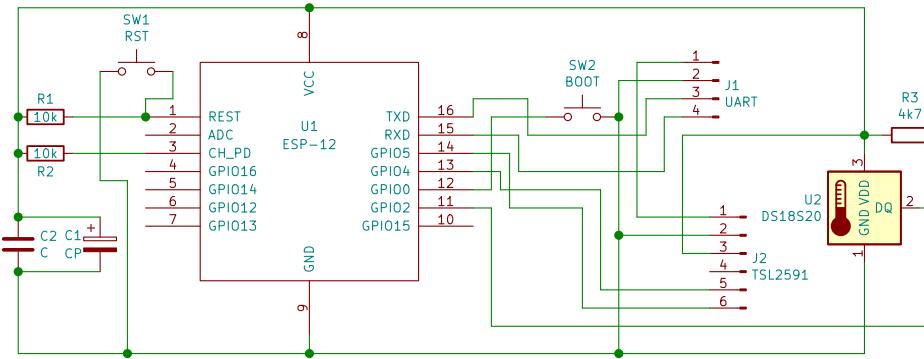
již v úvodu této kapitoly zmíněných, osazených DPS. Jednotlivá měření se předávají po WiFi dalším součástem systému pomocí protokolu MQTT.

Jedna z DPS je umístěna na polici v žaluzií zatemňované místnosti v domě autora této práce tak, aby na připojené senzory nedopadalo přímé sluneční světlo. Druhá je položena na střeše tohoto domu uvnitř krabičky s průhledným okénkem tak, aby byla elektronika krytá před povětrnostními vlivy, ale na senzor intenzity osvětlení i tak dopadalo světlo. Voděodolnost krabičky zajišťuje přišroubované víko se silikonovým těsněním a silikonem utěsněné kabelové průchody. Umístění na střeše bylo zvoleno tak, aby senzor intenzity osvětlení nebyl zastíněn dalšími nesouvisejícími předměty na střeše, jako jsou antény, jejich stožáry, komíny atp. Teplotní senzor vyvedený na kabelu o délce 0,75 m je umístěn za severní hranou střechy, kde je krytý před dopadajícím slunečním zářením. Fotografie skutečného umístění zařízení je na obrázku 3.1. Napájecí kabel je zatížen betonovou kostkou, aby se předešlo samovolnému pohybu zařízení po střeše vlivem silného větru.

Hlavními součástmi těchto zařízení, které jsou podrobněji popsány dále, jsou: DPS, modul ESP-12E s MCU ESP8266, modul se senzorem intenzity osvětlení TSL2591 a vodotěsný senzor teploty DS18B20 na kabelu. Na desce je dále přítomno 5 pasivních součástek, z toho 3 rezistory a 2 kondenzátory, a 2 tlačítka. Napájeny jsou síťovými zdroji stejnosměrného proudu o velikosti až 0,5 A a napětí 5 V.

3.1.1 Zapojení

Zapojení vychází z dokumentace součástek a jeho schéma je na obrázku 3.2. Do paměti MCU je nutné nahrát firmware, nejjednodušší možnost je využít vyvedeného rozhraní UART na pinech 15 a 16, dále k tomu slouží tlačítko SW2, které připojí GPIO0 na 0 V a přepne tak MCU do režimu nahrávání firmwaru. Společně s napájecími piny je tedy toto rozhraní vyvedeno z desky na konektoru J1 se 4 piny. Tlačítko SW1 slouží k propojení REST pinu (1) MCU a 0 V, což vyvolá jeho reset. Jinak je přes pull-up rezistor R1 připojen na napájecí napětí, aby se předešlo náhodnému resetování. Rezistor R2 pak



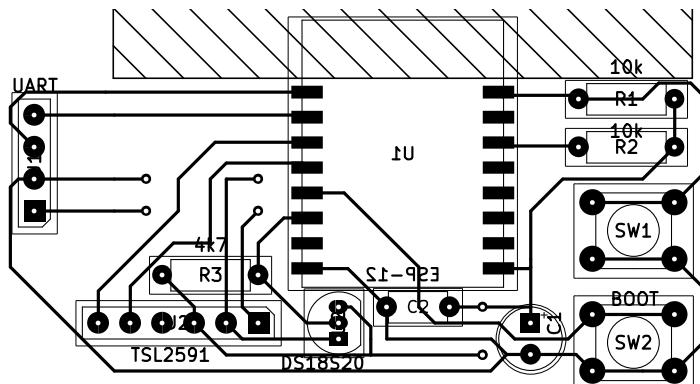
OBRÁZEK 3.2: Schéma zapojení zařízení, která měří intenzitu osvětlení pomocí senzoru TSL2591 a teplotu vzduchu pomocí senzoru DS18B20. Hodnoty se přenáší přes WiFi.

přivádí napětí na CH_PD pin, který tak uvede do chodu interní regulátory napětí pro procesor. Kondenzátory C1 a C2 slouží k vyhlazení napájecího napětí, omezení rušení a překlenutí odběrových špiček. Pro připojení senzoru teploty se využívá sběrnice 1-Wire, její datový vodič je připojen jednak na GPIO2, jednak přes rezistor R3 na napájecí napětí jako pull-up. Napájecí vodiče sběrnice jsou přímo spojené s napájecími vodiči MCU.

Na GPIO4 a GPIO5 jsou zapojeny po řadě datový a hodinový vodič sběrnice I²C, která slouží ke komunikaci se senzorem intenzity osvětlení. Ten dále slouží jako stabilizovaný zdroj napětí pro celé zařízení, součástí modulu je totiž lineární snižující zdroj napětí o velikosti 3,3 V.

3.1.2 Deska plošných spojů (DPS)

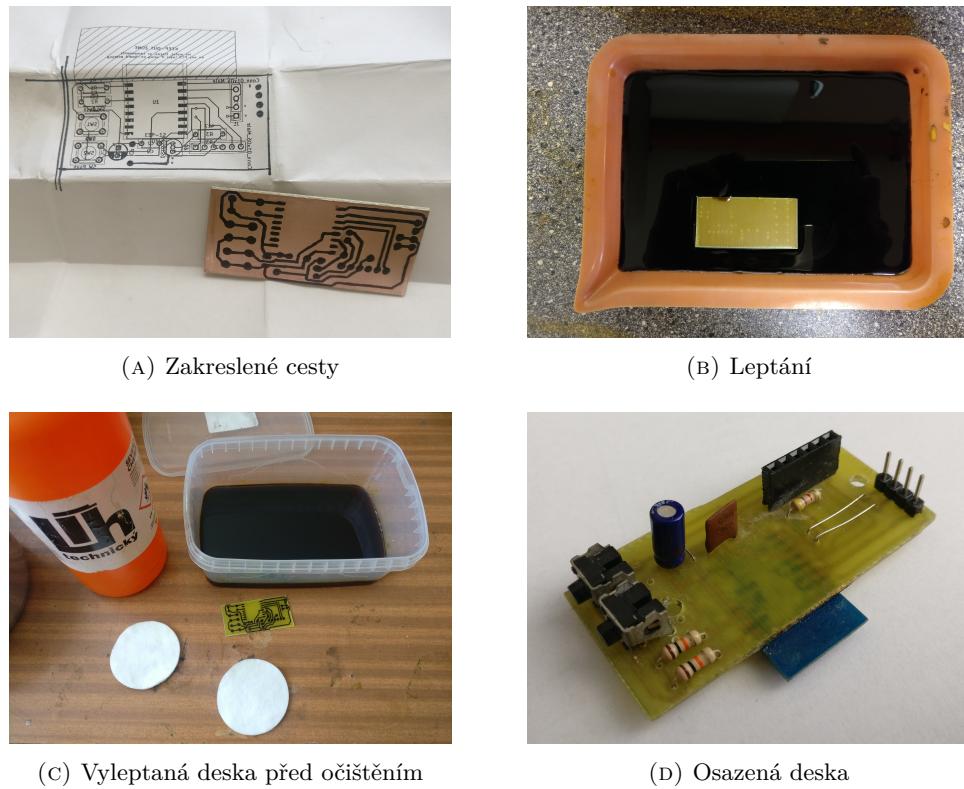
DPS elektricky propjuje ostatní komponenty těchto zařízení a byla vyrobena po domácku (obrázek 3.4). V opensource programu pro návrh elektronických zařízení KiCad byl sestaven obvod a návrh DPS (obrázek 3.3), jenž byl pomocí kancelářské tiskárny překlopeně vytisknán na běžný papír v měřítku 1:1.



OBRÁZEK 3.3: Návrh desky plošných spojů zařízení, která měří intenzitu osvětlení pomocí senzoru TSL2591 a teplotu vzduchu pomocí senzoru DS18B20. Hodnoty se přenáší přes WiFi. Šrafováná oblast je ochranná zóna antény.

Z jednostranně poměděného laminátu (cuprexit) byla pákovými nůžkami vystřížena deska o příslušných rozměrech dle návrhu, přebroušena brusným papírem a byla zabalena do papíru s návrhem tak, aby strana s mědí překrývala celou oblast návrhu, který byl ponechán na vnější straně. Kladivem a důlčíkem se do mědi poklepem na důlčík vyznačily budoucí díry pro jednotlivé součástky (v návrhu černá mezikruží, ze kterých mohou vést cesty) a krajní body pro osazení modulu ESP-12E (v návrhu černé obdélníky, ze kterých mohou vést cesty). Takové značky slouží jednak k orientaci při zakreslování vodivých cest, ale také při následném vrtání jako vodítka pro vrták.

Po vyznačení všech děr byla deska vyňata z papíru a lihovým fixem se zakresly pájecí body součástek (černá mezikruží a obdélníky) a cesty mezi nimi podle papírové předlohy (nejtenčí linie jsou ohrazení součástek a jejich pouzder a do desky se nezakreslují, protože nemají tvořit vodivé spojení). Pájecí body pro modul U1 (tedy ESP-12E) byly zakresleny pomocí samotného modulu použitého jako šablony zarovnané s důlčíkem vyznačenými krajními body. Deska po provedení tohoto kroku vypadá jako na obrázku 3.4a.



OBRÁZEK 3.4: Postup při výrobě desky plošných spojů

Přebytečná měď, která nebyla překryta barvou, se vyleptala při pokojové teplotě v roztoku chloridu železitého pro leptání plošných spojů za 25 minut a na desce tak zůstaly jen potřebné vodivé cesty. Leptání proběhlo tak, že se deska položila na hladinu roztoku nalitého do plastové leptací vaničky (obrázek 3.4b). Díky tomu mohly produkty chemické reakce klesat ke dnu a uvolnit prostor ještě nepoužitému roztoku. Deska byla několikrát z lázně vyjmuta a vizuálně zkонтrolována, jestli barva nebyla smyta a jestli je ještě potřeba leptat dále. Když už na desce zbývaly jen části překryté barvou

(obrázek 3.4c), tedy cesty a pájecí body, se barva očistila technickým lihem pomocí vatového polštářku a do desky se vyvrtaly otvory pro součástky. Nakonec byla deska znova přebroušena a opatřena pájitelným ochranným lakem (v tomto případě kalafunou rozpuštěnou v lihu). Součástky byly osazeny na příslušná místa dle návrhu desky a schématu (obrázek 3.4d).

3.1.3 Modul ESP-12E s MCU a WiFi

Modul ESP-12E vyvíjený společností Ai-Thinker obsahuje čip ESP8266, který integruje kompletní řešení WiFi, 32 bitový procesor *Tensilica L106 Diamond Series*, taktovaný na 80 MHz, s vestavěnou SRAM a 16 univerzálních vstupně-výstupních pinů. Další součástí modulu je flash paměť připojená přes rozhraní SPI, na kterou se ukládá program. (AI-Thinker Team, 2015) V popisovaném měřicím zařízení slouží tento modul ke zpracování dat naměřených pomocí senzorů teploty a intenzity osvětlení a jejich přenosu do dalších součástí systému.

Jako firmware byl použit MicroPython (George a Sokolovsky, 2014), který ve flash paměti modulu vytvoří jednoduchý souborový systém, z něhož spouští programy v jazyce MicroPython. Komunikace s modulem při vývoji včetně nahrávání souborů do paměti modulu probíhala přes sériové rozhraní modulu prostřednictvím nástroje MPFShell (Wendler, 2018).

3.1.4 Teplotní senzor DS18B20

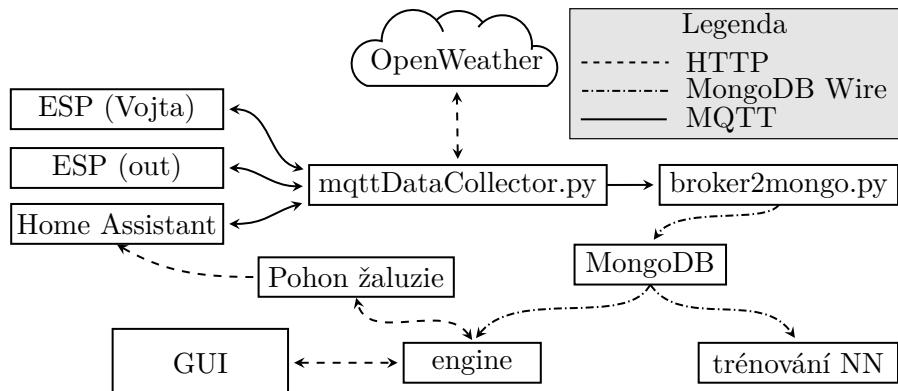
DS18B20 je digitální teploměr s volitelnou přesností 9 – 12 bitů, který svá měření poskytuje po sběrnici 1-Wire, k jeho provozu tedy postačuje pouze datový a zemnící vodič. Volitelně lze také připojit napájecí napětí o velikosti 3 – 5,5 V třetím vodičem. Na stejně sběrnici může být připojeno více teploměrů DS18B20, protože každý z nich má z výroby přidělené unikátní sériové číslo. Je dostupný v pouzdrech TO92 a SOP8 a naleze využití ve stavebnictví při řízení vytápění, ventilace nebo klimatizace, měření teplot v budovách, v průmyslu při měření a monitorování teploty zařízení nebo procesů a jejich řízení. (Maxim Integrated Products, Inc., 2019) V měřicím zařízení poskytuje data o teplotě vzduchu.

3.1.5 Senzor intenzity osvětlení TSL2591

TSL2591 je digitální senzor intenzity osvětlení, který pracuje ve viditelném a infračerveném spektru. Parametry měření, kterými jsou zesílení a čas integrace, jsou volitelné a je tak možné měřit intenzitu osvětlení od 188 lux do 88000 lux. Měření lze číst po standardní sběrnici I²C, která využívá dvou vodičů - datový vodič a vodič s hodinovým signálem. Použitý modul od firmy Adafruit s tímto integrovaným obvodem má také vlastní lineární regulátor napětí MIC5225 proudově zatížitelný až do 150 mA. (Adafruit Industries, 2014) Proto byl kromě měření intenzity osvětlení interiéru i exteriéru využit také pro napájení celého zařízení z rozšířených 5 V USB síťových zdrojů (např. adaptéry pro nabíjení spotřební elektroniky).

4 Sběr dat a komunikace komponent

Použitým algoritmům strojového učení je nutné dodat data sestávající z počtu hodnot příznaků (více v sekci 2.1) a odpovídajících stavů žaluzí (výška vytažení a sklon lamer) v daných časových okamžicích. Tato kapitola se zabývá metodami sběru dat v jednotlivých součástech systému, přenosem z nich a jejich vzájemnou komunikací.



OBRÁZEK 4.1: Schéma komunikace komponent systému automatického řízení žaluzí, tok dat mezi nimi a komunikační protokoly.

Data se získávají ze 3 hlavních zdrojů na základě požadavku zasláného skriptem `mqttDataCollector.py`. Stav žaluzie a informaci, zda je uživatel přítomen v domácnosti¹ odesílá systém domácí automatizace *Home Assistant*. Měřené veličiny (teplota a intenzita osvětlení) uvnitř a venku pak sbírájí příslušná zařízení². Ostatní informace o počasí³ se získávají přímo v rámci skriptu `mqttDataCollector.py` z REST API *OpenWeather*. Data získaná z komponent se pak v jedné společné zprávě odesílají přes MQTT skriptu `broker2mongo.py`, který je uloží do databáze MongoDB (sekce 4.4). Na obrázku 4.1 je přehled komponent, které se podílejí na sběru dat, a jejich komunikace včetně používaných protokolů.

¹příznaky position, tilt a home

²příznaky lum_in, lum_out, temp_in a temp_out

³rychllosť (`owm_wind_speed`) a směr (`owm_wind_heading`) větru, předpověď teploty na $x = 1, 2, 3$ h dopředu (`owm_temp_xh`) a předpověď nejvyšší denní teploty (`owm_temp_max`)

4.1 Použití komunikačního protokolu MQTT

Protokol MQTT (OASIS Open, 2014) se využívá pro veškerou komunikaci součástí při sběru dat. Centrální uzel se nazývá broker a jedná se o software spuštěný na Raspberry Pi (kapitola 3) na výchozím TCP portu 1883. Připojují se k němu všechny komponenty, které využívají MQTT. Po připojení mohou publikovat zprávy do hierarchicky uspořádaných témat a přihlašovat se k jejich odběru. Broker pak zajistí doručení zpráv publikovaných v určitém tématu klientům, kteří jsou přihlášeni k jeho odběru.

Při sběru dat se periodicky nebo na základě změny stavu žaluzie odešle do tématu smartblinds/command zpráva ve formátu JSON, která obsahuje pod klíčem „*command*“ hodnotu „*request_values*“. K odběru tohoto tématu jsou přihlášena obě zařízení s ESP8266 (sekce 3.1) i systém domácí automatizace a po jejím přijetí odešlou aktuální hodnoty jimi sledovaných příznaků do odpovídajících témat dle tabulky 4.1. Všechna tato téma odebírá skript `mqttDataCollector.py`, který hodnoty příznaků společně s časovou známkou odesílá v jedné zprávě do tématu smartblinds/data. Tato zpráva dále obsahuje informaci o tom, zda je aktivní testovací režim využívaný při vývoji. Téma smartblinds/data odebírá skript `broker-2mongo.py` popsaný v sekci 4.4. Ten všechna takto přenesená data ukládá do databáze k pozdějšímu použití.

Příznak nebo stav	Téma
temp_inside	smartblinds/temp/Vojta
temp_outside	smartblinds/temp/out
lum_inside	smartblinds/lux/Vojta
lum_outside	smartblinds/lux/out
home	smartblinds/presence/Vojta
position	smartblinds/position/Vojta
tilt	smartblinds/tilt/Vojta

TABULKA 4.1: Názvy příznaků a MQTT téma, kam se hodnoty příznaků odesírají, a která odebírá skript `mqttDataCollector.py`.

4.2 Komunikace s pohonem žaluzie

S pohonem žaluzie komunikuje centrální jednotka Tahoma, dodávaná výrobcem žaluzie, pomocí bezdrátového proprietárního protokolu *io*. Tato jednotka je dále připojená do internetu a je možné s ní komunikovat prostřednictvím rozhraní pro programování aplikací (API), které nabízí výrobce. Komunikace s API je šifrovaná a probíhá pomocí protokolu HTTPS zasláním požadavků na *endpointy* serverů výrobce dostupné na adrese <https://api.somfy.com/api/v1/>. Použitý systém domácí automatizace i backend vyvýjeného systému používají ke komunikaci s API knihovnu Pymfy v jazyce Python, která mapuje akce pro manipulaci se žaluzií a její stav na metody a proměnné objektu,

který je tak modelem žaluzie. Dále zpřístupňuje některé obecnější metody pro práci s API jako je například zjištění všech montáží zákazníka či zjištění všech dostupných zařízení v rámci konkrétní montáže.

Jednotlivé požadavky se autorizují na základě krátkodobého tokenu s platností 1 h. Pokud vyprší jeho platnost, je nutné pomocí obnovovacího tokenu ze serveru získat nový a přikládat ho k budoucím požadavkům. Oba tokeny se získávají OAuth2 metodou „Authorization Code Grant“, z důvodů odlišností od dokumentace v implementaci Somfy se ale ani po kontaktování technické podpory nepodařilo získat nové přístupové údaje k API a tak muselo být využito nedokumentovaného postupu k jejich získání. Přestože jsou tokeny v obou systémech odvozené od stejných přístupových údajů, zdají se být nezávislé (včetně kvóty na četnost požadavků) a na funkčnost to tedy nemá vliv. Hledání tohoto postupu se zdálo být časově nákladné a proto se ke zjišťování aktuálního stavu žaluzí využívá právě systém domácí automatizace.

Běžně je nutné si na webových stránkách na adrese <https://developer.somfy.com> vytvořit tzv. aplikaci pod uživatelským účtem, ke kterému je technikem při instalaci žaluzí přiřazena konkrétní montáž. Na základě zadaného názvu, *callback URL*¹ a popisu aplikace se získájí dva řetězce: *Consumer key* a *Consumer secret*. (Somfy, 2018) Ty se pak společně s *callback URL*¹ předají konstruktoru objektu, který v rámci knihovny Pymfy reprezentuje API. (Etienne, 2021)

Takto vytvořené *Consumer key* a *Consumer secret* se však nedařilo použít, server Somfy je totiž zamítal. Byla tedy kontaktována technická podpora, ale ani po několika týdnech se nedostavila odpověď. Mezitím pokračovaly pokusy o získání tokenu jinak. Pro systém domácí automatizace, který se se žaluziemi již používal, existovaly tyto údaje a ukázalo se, že jsou funkční. Místo správné *callback URL* se tedy do konstruktoru zadala ta, která příslušela k aplikaci ve vývojářském portálu, a byl zahájen proces získání tokenů. Po přesměrování zpět se v adresním rádku prohlížeče ručně změnila adresa tak, aby odpovídala endpointu vytvořenému k získávání a ukládání tokenů v rámci backendu. Pokud by bylo možné postupovat standardním způsobem, po přihlášení na stránkách Somfy by byl prohlížeč přesměrován právě na tuto adresu. Tokeny se tak uložily do souboru `somfycache` pro pozdější použití.

Data a služby, které API poskytuje jsou využívány ke dvěma účelům. Jednak systém domácí automatizace každou minutu kontroluje aktuální stav žaluzí a pokud se změní, odešle novou hodnotu přes MQTT a spustí tak posloupnost akcí (popsanou v sekci 4.1), které vedou k vytvoření nového záznamu v databázi, jednak stránka „Control“ v GUI umožňuje zobrazit aktuální stav žaluzí a zadávat příkazy k jeho změně. Samotnou komunikaci s API zajišťuje backend systému, se kterým se komunikuje přes protokol WebSockets (Fette a Melnikov, 2011). Po připojení alespoň jednoho klienta se každých 15 s zjistí stav žaluzie a připojeným prohlížečům se odešle zpráva ve formátu JSON. Její struktura je uvedená v úryvku kódu 4.1. Naopak po přijetí zprávy backendem se v závislosti na jejím obsahu odešle požadavek API

¹Návratová URL v rámci vyvýjené aplikace, na kterou se přesměruje webový prohlížeč uživatele po úspěšné autorizaci, prostřednictvím parametrů v URL se aplikaci předá kód, na základě kterého může získat tokeny ze serveru.

na změnu stavu žaluzie. Zpráva má stejnou strukturu jako zpráva v úryvku kódu 4.1, ale obsahuje jen jeden z klíčů. Navíc může obsahovat speciální klíč `testing` s hodnotou datového typu `boolean`, pomocí kterého lze měnit příznak `testing` dat ukládaných do databáze, který slouží k rozlišení akcí vyvolaných při vývoji v rámci testování a skutečných akcí, které má systém napodobovat. Zpráva o změně režimu se v tomto případě odešle přes MQTT skriptu `mqttDataCollector.py`, který příznak zaznamenává k datům. Zprávy se odesírají na základě interakce uživatele s ovládacími prvky GUI.

```
{
    "position": int
    "tilt": int
}
```

ÚRYVEK KÓDU 4.1: Struktura zprávy o stavu žaluzie, která je všem klientům zasílána každých 15 s. Klíč `position` označuje výšku vytažení žaluzie (0 – zavřeno, 100 – otevřeno), obdobně hodnota `tilt` vyjadřuje naklopení lamel.

4.3 Využití služby OpenWeather

Skript `mqttDataCollector.py` kromě měřených příznaků a stavu žaluzí zjišťuje také odhad počasí a jeho předpověď. Tato data poskytuje služba OpenWeather prostřednictvím svého API, které má několik možností využití. Z důvodu jednoduchosti byla pro přístup k API zvolena knihovna PyOWM, která používá variantu „One-Call“. Na základě jednoho požadavku se tak získá aktuální počasí, předpověď po minutách na následující hodinu, předpověď po hodinách na následující 2 dny, předpověď po dnech na následující týden, výstrahy vydané Českým hydrometeorologickým ústavem a historická data z posledních 5 dnů. Vše je pak přístupné pomocí atributů a metod objektu v Pythonu. K požadavku je vždy připojen klíč, který byl bezplatně získán po registraci na webových stránkách <https://openweathermap.org>. Limity četnosti požadavků stanovené poskytovatelem jsou vyšší než 1 požadavek za 5 minut. Jako příznaků se využívá kód počasí (vyjadřuje jeho shrnutí – např. slunečno, polojasno, dešt, jasná noc atp.), rychlosti a směru větru, hodinové předpovědi teploty na následující 3 hodiny a předpovědi nejvyšší denní teploty.

4.4 Databáze

Data, která se sbírají pomocí skriptu `mqttDataCollector.py` v pětiminutových intervalech nebo na základě změny stavu žaluzie uživatelem, ukládá skript `broker2mongo.py` do databáze MongoDB (MongoDB, Inc., 2021). Jednotlivé vzorky jsou získávány z MQTT zpráv zasílaných do tématu `smartblinds/data` a ukládají se ve stejně podobě jako příchozí zpráva. Struktura je uvedena v úryvku kódu 4.2.

```
{
    "timestamp": float,
    "testing": boolean,
    "periodical": boolean,
    "features": {
        "year_day": int,
        "week_day": int,
```

```

    "day_secs": int,
    "home": boolean,
    "temp_in": float,
    "temp_out": float,
    "lum_in": float,
    "lum_out": float,
    "owm_temp_max": float,
    "owm_temp_1h": float,
    "owm_temp_2h": float,
    "owm_temp_3h": float,
    "owm_code": int,
    "owm_wind_speed": float,
    "owm_wind_heading": float
},
"targets": {
    "position": int,
    "tilt": int
}
}

```



ÚRYVEK KÓDU 4.2: Struktura vzorku dat uloženého jako záznam v databázi MongoDB. Obsahuje hodnoty příznaků i stav žaluzí v okamžiku jeho pořízení a časovou známku. Místo hodnot jsou zde uvedeny jejich datové typy (float - číslo s plovoucí řádovou čárkou, boolean - pravdivostní hodnota, int - celé číslo)

Uložená data se využívají při strojovém učení modelů, ve vizualizaci dat na stránce „Live“ v GUI (sekce 6.4) a v porovnání skutečného řízení a řízení jednotlivých regresorů tamtéž. Také se zobrazují v tabulce na stránce „Data“. Vyhodnocení sběru dat je uvedené v sekci 7.1.

4.5 Webový server pro komunikaci s GUI

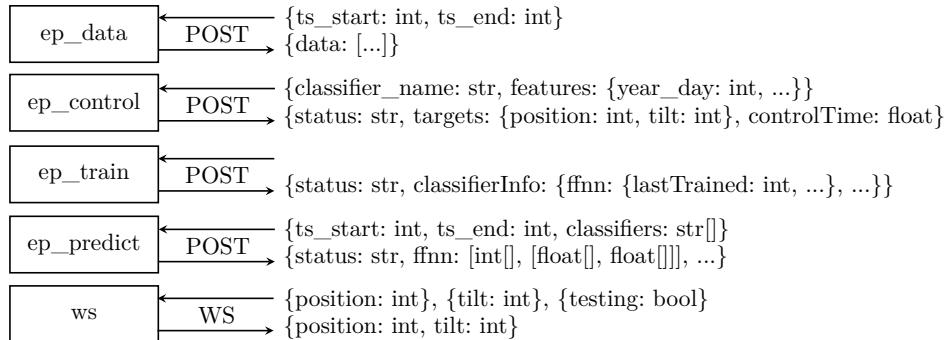
Systém má webové uživatelské rozhraní poskytované *Next.js* serverem (kapitola 6). Jeho propojení s jádrem systému (backend) je řešeno pomocí frameworku Tornado v jazyce Python. Tento webserver tak tvorí rozhraní mezi GUI a databází, vsemi regresory (predikce, simulace i učení) a Somfy API. Klient v podobě webového prohlížeče odesílá HTTP požadavky na tyto endpointy:

- `ep_data` poskytuje data z databáze na základě POST požadavku s volitelnými JSON parametry (`ts_start` a `ts_end`) v těle označujícími časovou známku začátku a konce intervalu, vrací seznam vzorků (úryvek kódu 4.2) pod klíčem `data` ve formátu JSON jako tělo odpovědi.
- `ep_control` slouží k predikci řízení jednoho z regresorů (parametr `classifier_name` na základě hodnot příznaků předaných jako JSON parametr `features` (jeden vzorek), vrací `controlTime` - dobu trvání predikce, `status: ok` a navržené hodnoty řízení (jako v úryvku kódu 4.1)).
- `ep_train` spustí přetrénování NS (sekce 5.5) na všech dostupných datech.
- `ep_predict` slouží k predikci řízení regresorů předaných v parametru `classifiers` pro reálná data (přijímá začátek a konec intervalu

stejným způsobem jako ep_data, načte všechny vzorky z tohoto interavlu). Pro každý z regresorů vrací pod klíčem s jejich názvem dobu predikce a seznam dvojic doporučeného řízení a odpovídajících časových známk.

- ws zajišťuje spojení pomocí protokolu WebSockets pro pravidelnou aktualizaci vizualizace stavu žaluzí a pro přenos požadavků na jeho změnu.

Schéma těchto endpointů a některých přenášených dat je na obrázku 4.2.



OBRÁZEK 4.2: Schéma endpointů backendu systému pro automatické ovládání žaluzí a datové struktury přenášené v těle jednotlivých požadavků a odpovědí včetně metody.

V posledním případě se jedná o protokol WebSockets.

5 Automatické ovládání žaluzií

Pro automatické ovládání žaluzií byly navrženy tři regresory, které odhadují vhodné nastavení žaluzie (výška vytážení „position“ a míra naklopení „tilt“) na základě 15 místního příznakového vektoru (případně časové posloupnosti těchto vektorů, více v sekci 2.1). Použité regresory byly nazvány „If-else“, „FFNN“ a „LSTM“ podle svého principu. Tato kapitola se zabývá jejich návrhem, specifickými vlastnostmi, které z něj vyplývají, a jeho metodami.

Poslední dva zmíněné regresory (FFNN a LSTM) jsou neuronové sítě (NS). K práci s jejich modely bylo využíváno rozhraní Keras v jazyce Python, které umožňuje jejich učení, vyhodnocování i následné použití pro regresi (Keras, 2022). Při volbě jejich vnitřní struktury a parametrů učení se natrénovaly sítě se všemi možnými kombinacemi určitých parametrů (několik realizací s náhodnou inicializací vnitřních parametrů) a následně se vybrala nejlepší z nich ve smyslu minimálního průměru střední kvadratické odchylky (MSE) na testovací datové sadě (sekce 5.1). Parametry, prohledávané hodnoty a počty realizací jsou uvedeny v sekcích 5.3 a 5.4.

Všechny implementace regresorů mají jednotné rozhraní a jimi doporučené řízení poskytují prostřednictvím metod `control()` a `predict()`. První z nich je určena pro jednotlivé vzorky, druhá pak pro jejich množinu. Obě se volají při obsluhování příslušných požadavků z uživatelského rozhraní při použití Simulátoru (sekce 6.2) nebo grafu řízení na záložce Live (sekce 6.4). Metoda `predict()` se od metody `control()` liší tím, že u NS se celá vstupní data zpracovávají najednou, což má pozitivní vliv na rychlosť.

Přetrénování regresorů založených na NS je možné spustit pomocí metody `train()` na předaných datech.

5.1 Příprava dat pro strojové učení

Naměřená data jsou uložená v databázi (kapitola 4) ve svých původních jednotkách a jednotlivě podle okamžiku pořízení. Pro jejich využití při strojovém učení byla data načtená z databáze přeškálována a vytvořily se z nich vzorky dle architektury jednotlivých NS. Vzorky byly náhodně a disjunktně rozděleny do 3 datových sad (trénovací, validační a testovací) v poměru 8:1:1. Výše popsané činnosti vykonával skript `mongo2h5.py`, který na závěr 3 datové sady pro každou síť uložil do souboru pro pozdějsí využití.

Hodnoty jednotlivých veličin byly převedeny do intervalu $\langle 0, 1 \rangle$ posunutím a škálováním očekávaného intervalu jejich hodnot uvedeného v tabulce 2.1. Pro regresor FFNN každý vzorek v datových sadách odpovídal jednomu naměřenému vzorku, zatímco pro regresor LSTM každý obsahoval příznaky ze

64 časových okamžiků a požadované hodnoty nastavení žaluzie pro poslední z nich. Uložený soubor s datovými sadami byl ve formátu h5 a jeho název obsahoval informaci o časovém intervalu, ze kterého data v něm pochází, a o architektuře NS, pro kterou je určen.

Datová sada se disjunktně dělí na 3 menší datové sady. Každá z nich slouží k různým účelům při vývoji systému.

Trénovací datovou sadu tvoří 80% náhodně vybraných vzorků a při trénování jsou její vzorky po dávkách předkládány NS. V každém kroku je pro ně vyhodnocována ztrátová funkce, jejíž hodnoty slouží k nastavení vnitřních parametrů učícím algoritmem.

Validační datovou sadu tvoří 10% náhodně vybraných vzorků. V každém kroku učení se na ní vyhodnocuje několik metrik. Neslouží přímo pro učení NS - ověřuje se na ní, zda síť dobře zobecňuje a nedochází k přetřenování na trénovací datovou sadu (overfitting). Vybíral se vždy doposud nejlepší model podle hodnoty ztrátové funkce vyhodnocené na validační datové sadě.

Testovací datová sada sestává z 10% náhodně vybraných vzorků. Byla použita pro vyhodnocení úspěšnosti učení sítí s různou strukturou při hledání té nejvhodnější (sekce 5.3 a 5.4) a také pro celkové vyhodnocení regresorů a vlivu příznaků na odhad regresorů (sekce 7.3 až 7.5).

5.2 Regresor založený na pravidlech (If-else)

Základní regresor „If-else“ tvoří výstup deterministicky na základě pravidel vyjádřených ve formě zřetězených konstrukcí if-elif-else (odtud tedy název) v programovacím jazyku Python. Slouží k porovnání navržených metod strojového učení s rozhodováním dle neměnných pravidel, které se v současnosti v praxi v domácí automatizaci obvykle používá (Apple Inc., 2022; OpenHAB Foundation, 2022; Home Assistant, 2022).

Pravidla byla konstruována podle slovy formulovaných požadavků uživatele na stav žaluzí v závislosti na aktuálních podmínkách (vyjádřených příznamkovým vektorem) s ohledem na to, že konkrétní parametry může být nutné změnit, protože je uživatel není schopen přesně definovat, nastavují se proto v konfiguračním souboru `cfg_ifelse.yml`. Formulována byla tato pravidla  výšku vytažení žaluzí:

1. Ve dne¹:

¹Den znamená, že je vnější intenzita osvětlení vyšší než nastavená mez a zároveň je čas vyšší než nastavený čas vstavání v daný den týdne. Jinak je noc.

- (a) Během jara a podzimu²: Je-li chladno³, žaluzie má být vytažená, jinak v případě střední teploty⁴ se má rozhodovat na základě rozdílu mezi teplotami předpovězenými za 2 a 1 hodinu (růst teploty). Pokud je růst teploty v nastaveném pásmu nebo vyšší nebo je teplota vnějšího vzduchu vyšší než obě meze, žaluzie má být ve stejné pozici jako by bylo léto (následující bod), v případě růstu nižšího se má vytáhnout.⁵
- (b) V léte⁶: Pokud to dovolují podmínky a uživatel je doma, žaluzie má být vytažená, jinak zatažená. Podmínky vhodné pro vytažení žaluzie jsou takové, kdy nehrozí přehřátí interiéru - rozdíl azimutu slunce a azimutu směru kolmého na žaluzii (směrem od domu) v absoultní hodnotě je vyšší než 90° nebo je předpovězená nejvyšší denní teplota nižší, než nastavená mez (například z důvodu oblačnosti, deště atp.).
- (c) V zimě⁷: Žaluzie má být vytažená.

2. V noci v kterémkoliv období má být žaluzie zatažená.

Kromě pravidel pro výšku vytažení byla formulována také pravidla pro míru naklopení:

1. Přes den:

- (a) Během jara a podzimu: V případě potřeby (vysoká intenzita osvětlení venku a zároveň alespoň vysoká teplota vzduchu uvnitř nebo teplota venkovního vzduchu, vše vzhledem k nastavitelným mezmí) se má žaluzie naklopit jako by bylo léto, jinak mají být její lamely vodorovně.
- (b) Během léta se má žaluzie naklopit o nejmenší možný úhel vzhledem k vodorovné rovině tak, aby zabránila (je-li to vzhledem k vzájemné poloze Slunce a žaluzie nutné) průchodu přímého slunečního záření oknem. Jestliže je ale intenzita osvětlení nižší než nastavená mez (vlivem počasí, nebo času), mají být lamely žaluzie vodorovně.
- (c) V zimě je žaluzie vytažená a úhel naklopení nemá smysl uvažovat, z konstrukce bude žaluzie vodorovně.
- (d) V případě, že uživatel není doma, naklopení žaluzie (jiné než vodorovné) má odpovídat $\frac{8}{10}$ jinak definovaného naklopení.

2. V noci: Žaluzie úplně naklopená, vyjímkou je případ, kdy je vnitřní teplota nižší než nastavená mez - to může znamenat, že se uživatel snaží větrat a je tak vhodné žaluzie otevřít jen tak, aby mezi lamelami

²Jaro je období, které zahrnuje právě dny v roce z intervalu $\langle 80, 134 \rangle$, obdobně podzim zahrnuje dny z $\langle 274, 320 \rangle$

³Chladno znamená, že je teplota venkovního vzduchu nižší než nastavená mez. Zvolena byla podle zkušenosti s vytápěním použitého domu na 10°C .

⁴Střední teplota znamená že je teplota venkovního vzduchu v intervalu mezi zmíněnoumezí pro „chladno“ a další nastavitelnou mezí, která byla zvolena na 18°C .

⁵Nižší hranice pásmá byla zvolena na $-1,5^\circ\text{C}$, vyšší pak na $1,5^\circ\text{C}$.

⁶Léto jsou dny v roce z intervalu $\langle 135, 273 \rangle$

⁷Zima jsou ostatní dny, tedy dny v roce z intervalů $\langle 321, 366 \rangle \cup \langle 1, 79 \rangle$

mohl lépe proudit vzduch, ale i přes to bránily průchodu světla pod malými úhly (po ulici projíždějící auta atp.).

Z návrhu vyplývá, že se tento regresor nepřizpůsobuje novým návykům uživatele, protože využívá v čase neměnných pravidel. Některá pravidla mají volitelné parametry, které lze ladit a dosáhnout tak přesněji uživatelem požadovaných výsledků.

5.3 Dopředná neuronová síť jako regresor (FFNN)

První regresor založený na neuronových sítích byl nazvaný „FFNN“ podle anglického sousloví *Feedforward Neural Network*, které označuje dopřednou neuronovou síť. Při volbě její vnitřní struktury byly prohledávány všechny kombinace hodnot parametrů uvedených v tabulce 5.1. Každá z kombinací se trénovala 10x pro různé náhodné inicializace parametrů sítě. Všechny

Parametr	Seznam hodnot
struktura sítě	[15, 10], [10, 15, 10], [20, 15, 10], [10, 15, 20]
počet epoch	500, 200, 700
batch size	16, 32

TABULKA 5.1: 3 parametry dopředné neuronové sítě a jejího učení, jejichž kombinace se prohledávaly za účelem získání nejlepší sítě pro úlohu regrese stavu žaluzie podle 15 příznaků. Parametr struktura sítě reprezentují posloupnosti (v hranatých závorkách) počtu neuronů v jednotlivých vrstvách sítě.

vrstvy sítě jsou plně propojené. Neurony v nich využívají aktivační funkci ReLU, kromě výstupní vrstvy, která je aktivována sigmoidou (obě funkce jsou popsány v sekci 1.2.2), která je omezená shora i zdola a výstup vynásobený koeficientem je tak možné přímo  získat jako případný akční zásah. Jako ztrátová funkce byla použita MSE.

5.4 Rekurentní neuronová síť jako regresor (LSTM)

Tento regresor je rekurentní neuronová síť, která používá *Long Short-Term Memory* (LSTM) buňky (sekce 1.2.4). Vnitřní struktura byla volena na základě výsledků prohledávání parametrů (popsáno v úvodu této kapitoly), prohledávané hodnoty každého z nich jsou uvedené v tabulce 5.2. Každá z kombinací byla natrénována 3x s různou inicializací vnitřních parametrů sítě (vah a biasů).

Aktivační funkcí je v tomto případě ve všech vrstvách hyperbolický tangens a funkce sigmoid pro rekurentní krok. Výstupní vrstva (sigmoide) i ztrátová funkce (MSE) je stejná jako v případě NS regresoru FFNN.

Parametr	Seznam hodnot
struktura sítě	[64, 32], [64, 64], [64, 16, 16]
počet epoch	100, 200
batch size	16, 64

TABULKA 5.2: 3 parametry rekurentní neuronové sítě a jejího učení, jejichž kombinace se prohledávaly za účelem získání nejlepší sítě pro úlohu regrese stavu žaluzie podle 15 příznaků. Parametr struktura sítě reprezentuje posloupnosti (v hranatých závorek ) počtu neuronů v jednotlivých vrstvách sítě.

5.5 Přetrénování neuronových sítí (retraining)

Přetrénováním se rozumí opakování trénování NS regresorů na všech v daný okamžik dostupných datech. Přetrénování probíhá na již existujícím modelu, který se jen dále upravuje a zpřesňuje. Automaticky se v backendu plánuje na půlnoc každého dne, ale je ho možné také ručně spustit v záhlaví každé ze stránek GUI. Díky tomu se systém může přizpůsobovat novým okolnostem v datech, kterými mohou být zejména nové návyky uživatele.

Aby bylo přetrénování rychlé, používá se méně epoch a větší batch size než při běžném trénování. Obě NS se přetrénovávají s batch size 128, FFNN 50 epoch, LSTM pak 2 epochy.

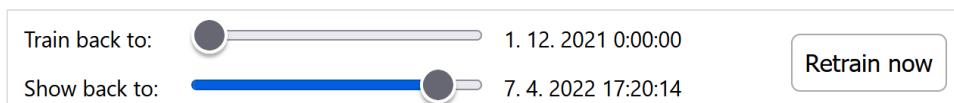
Vyhodnocení vlivu přetrénování neuronových sítí na kvalitu nasazeného modelu je uvedeno v sekci 7.5.

6 Grafické uživatelské rozhraní

K ovládání navrhovaného systému slouží grafické uživatelské rozhraní (GUI) implementované jako webová aplikace s využitím frameworku Next.js (Vercel, 2022). Jednotlivé funkce systému jsou zpřístupněny na 4 oddělených stránkách, které jsou dále popsány v sekcích 6.1 až 6.4. Klíčové funkce jednotlivých stránek jsou následující:

- Data - zobrazení hodnot naměřených příznaků a ručního řízení v tabulce;
- Simulator - vyhodnocování výstupu regresorů pro uživatelem libovolně nastavené hodnoty příznaků (simulace výstupu regresorů pro zadанé podmínky);
- Control - ruční řízení žaluzie;
- Live - grafické zobrazení hodnot skutečného řízení a řízení získaných z jednotlivých regresorů společně s naměřenými daty.

Celá aplikace se skládá z komponent psaných v jazyce TypeScript, Next.js se správcem balíků Yarn se pak stará o jejich komplikaci do JavaScriptu a následný přenos do prohlížeče uživatele. Všechny stránky mají společné záhlaví, které obsahuje ovládací panel (obrázek 6.1) pro nastavení intervalu pro zobrazení dat a pro trénování neuronových sítí regresorů (kapitola 5), zároveň je možné pomocí tlačítka spustit přetrénování na zvolených datech (sekce 5.5). V pravém rohu záhlaví je pak menu, které slouží k přepínání stránek v rámci GUI.



OBRÁZEK 6.1: Ovládací panel umístěný v záhlaví grafického uživatelského rozhraní systému pro automatické ovládání žaluzie. Umožňuje nastavení intervalu pro zobrazení dat a pro trénování neuronových sítí využívaných pro ovládání žaluzie a spuštění přetrénování na zvolených datech.

6.1 Stránka Data

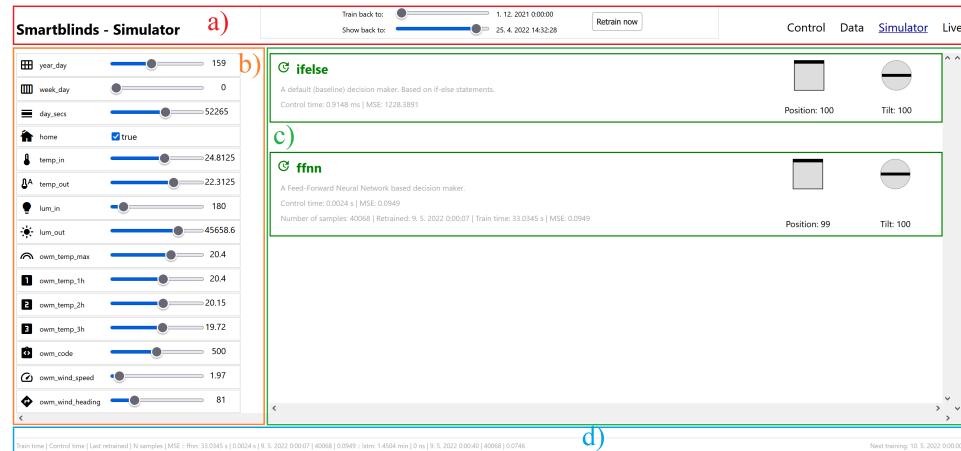
Většinu stránky Data (obrázek 6.2) zaujímá tabulka (b), ve které jsou v řádkách uvedeny jednotlivé vzorky naměřených dat s hodnotami všech příznaků a řízení žaluzií uživatelem společně s časem pořízení vzorku. Vlevo od každého řádku je navíc ikona, která reprezentuje příčinu pořízení vzorku:



OBRÁZEK 6.2: Stránka Data v grafickém uživatelském rozhraní systému pro automatické ovládání žaluzí s vyznačenými částmi pomocí barevných obdélníků a) – d)

C pro periodické vzorky, **+/-** pro vzorky pořízené na základě změny stavu žaluzie (toto chování je popsáno v kapitole 4). Vzorky jsou seřazeny chronologicky od nejmladších k nejstarším a zobrazují se pouze ty, které jsou z časového intervalu daného nastavením v záhlaví aplikace (a), popsáno v úvodu této kapitoly). V záhlaví tabulky jsou místo označení příznaků použity ikony, které nezabírají tolik místa. Jejich význam vysvětluje legenda, která je umístěna vpravo od tabulky (c)). Záhlaví tabulky i legenda zůstávají při rolování tabulkou směrem ke starším datům zafixované v horní části pohledu tak, aby měl uživatel stále přehled o významu hodnot, které sleduje.

6.2 Stránka Simulator



OBRÁZEK 6.3: Stránka Simulator v grafickém uživatelském rozhraní systému pro automatické ovládání žaluzí s vyznačenými částmi pomocí barevných obdélníků a) – d)

Tato stránka slouží pro ruční vyhodnocování výstupu regresorů v závislosti na nastavených hodnotách příznaků. Je rozdělena do dvou částí, vlevo je sloupec vstupních prvků (b)), vpravo pak rámce jednotlivých regresorů s jejich výstupem a jeho jednoduchou vizualizací (c)). Výška vytažení žaluzie



Position: 33 Tilt: 91

OBRÁZEK 6.4: Příklad vizualizace výstupu regresoru v úloze automatického ovládání žaluzie na základě 15 příznaků.

(Position) je reprezentována částečným vybarvením obdélníku shora, úhel naklopení lamel (Tilt) pak rotací čáry uvnitř kruhu (příklad je na obrázku 6.4).

Vzhledem k povaze vstupních prvků (b)), pomocí kterých je možné zadat hodnoty pouze v jednom časovém okamžiku, se vyhodnocuje výstup pouze regresoru If-else a FFNN, regresor LSTM totiž vyžaduje hodnoty příznaků ne z jednoho, ale z 64 okamžiků, což by vyžadovalo složitější způsob zadávání.

Při každé změně vstupu se pro každý sledovaný regresor odešle požadavek na endpoint ep_control backendu systému (sekce 4.5), v odpovědi se vrátí výstup regresoru, který je poté zobrazen.

6.3 Stránka Control

OBRÁZEK 6.5: Stránka Control v grafickém uživatelském rozhraní systému pro automatické ovládání žaluzí s vyznačenými částmi pomocí barevných obdélníků a) – d)

Další stránkou v GUI je Control, která slouží k zobrazení aktuálního stavu žaluzie a k jeho ruční změně. Skládá se ze dvou hlavních částí uspořádaných pod sebou - tabulky se stavy žaluzie (nahoře, b)) a vizualizace (dole, c)).

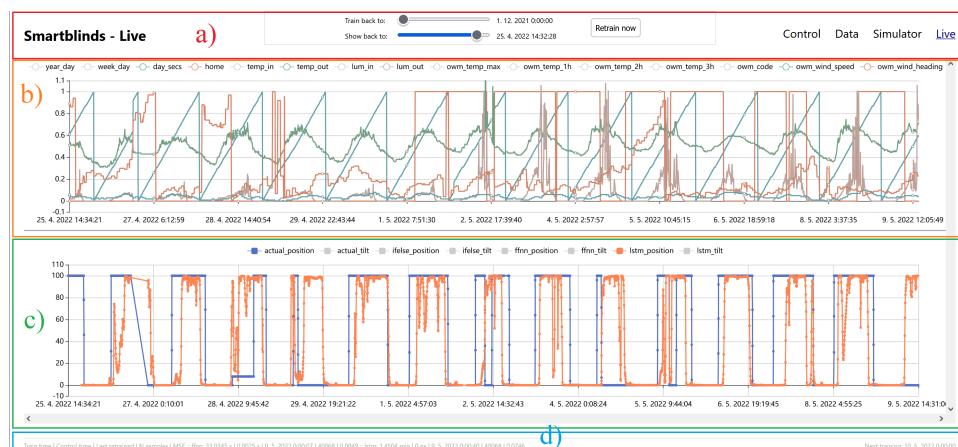
Tabulka (b)) má tři sloupce, v prvním z nich se zobrazuje aktuální stav, ve druhém jsou pole pro nastavení nového stavu a ve třetím tlačítka pro odeslání požadavku na změnu stavu. Každý řádek je pro jednu stavovou proměnnou žaluzie.

Vizualizace (c) zobrazuje žaluzii v řezu. Počet lamel a vzdálenost spodní lamely od okraje odráží výšku vytažení žaluzie, jejich odchylka od vodorovné osy pak odpovídá náklonu lamel skutečné žaluzie. Při změně hodnot v tabulce se vizualizace přepne do režimu náhledu, kdy je možné sledovat, jak by skutečná žaluzie vypadala, pokud by se odeslaly hodnoty nastavené v tabulce. Uživatele o tom informuje text *Preview* v pravém horním rohu vizualizace. Tlačítkem *Reset*, nacházejícím se tamtéž, je možné tento režim opustit a vizualizace pak bude opět zobrazovat aktuální stav žaluzie. Vizualizace je psána ve značkovacím jazyce pro dvouozměrnou vektorovou grafiku SVG (Dahlström et al., 2011).

Mezi tabulkou a vizualizací je navíc ještě zaškrťávací pole *Test Mode*, které umožňuje aktivovat nebo deaktivovat testovací režim při sběru dat, který se využívá při vývoji, aby databáze neobsahovala data vzniklá na základě falešných událostí.

Pro získávání aktuálního stavu a nastavení nového se používá spojení s backendem (sekce 4.5) pomocí protokolu WebSockets na jeho endpointu ws. Po připojení je každých 15 s do webového prohlížeče uživatele GUI doručován aktuální stav žaluzie ve zprávách ve formátu JSON a stejným způsobem se odesílá požadavek na jeho změnu. V pravém horním rohu stránky je indikátor spojení pomocí protokolu WebSockets.

6.4 Stránka Live



OBRÁZEK 6.6: Stránka Live v grafickém uživatelském rozhraní systému pro automatické ovládání žaluzií s vyznačenými částmi pomocí barevných obdélníků a) – d)

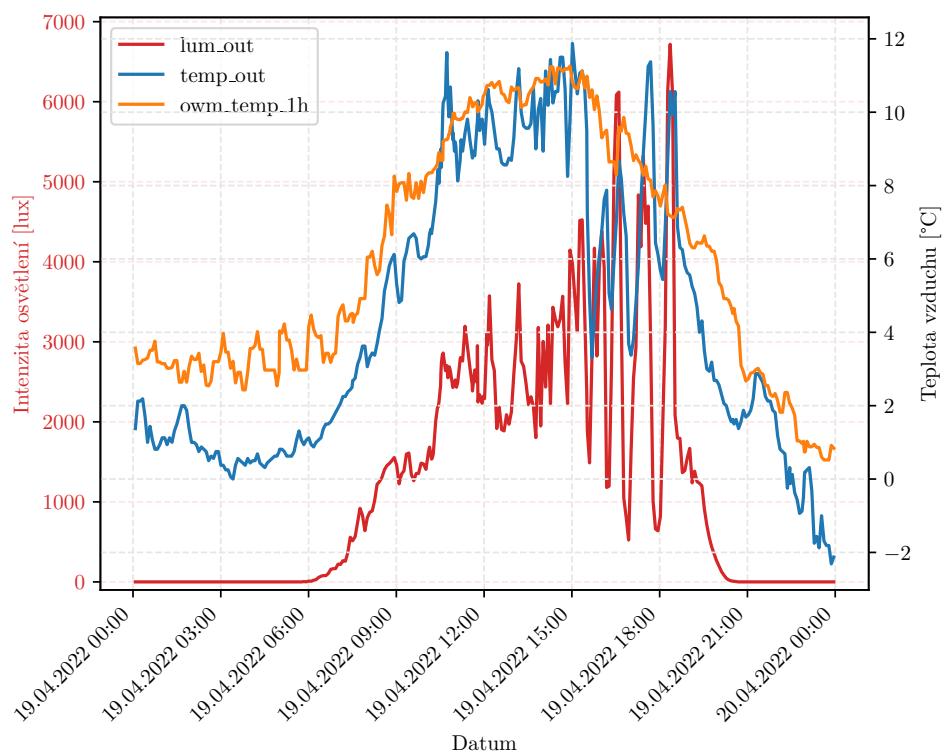
Poslední ze 4 stránek GUI je stránka Live, která slouží k živému vyhodnocování řízení doporučeného jednotlivými regresory na reálných datech. Skládá se ze dvou grafů uspořádaných pod sebou. V horním grafu ((b)) jsou zobrazena naměřená data, v dolním (c) pak řízení žaluzie (manuální řízení i doporučené řízení regresorů). Nad každým z grafů se nachází interaktivní legenda, která umožňuje skrývat některé linie v grafu pro větší přehlednost podle požadavků uživatele GUI. Kurzorem myši lze sledovat konkrétní hodnoty všech veličin v grafu v okamžiku daném jeho polohou.

Data zobrazovaná v grafech se získávají pomocí 2 požadavků zasílaných na endpointy backendu (sekce 4.5). Naměřená data se přenáší vždy všechna na základě požadavku na `ep_data` při prvním načítání aplikace, predikce regresorů kromě okamžiku prvního načtení aplikace také vždy při změně časového intervalu pro zobrazení dat v záhlaví aplikace (popsáno v úvodu této kapitoly). Oba grafy ale vždy zobrazují pouze data z časového intervalu daného nastavením v záhlaví stránky ([a](#)), *Show back to:*).

7 Vyhodnocení

7.1 Vyhodnocení sběru dat a spolehlivosti

Pro účely této práce se sbírala data (kapitola 4) od 9. prosince 2021 17:38 (UTC) do 2. května 2022 11:59 (UTC). Za tu dobu bylo sesbíráno 38128 vzorků dat, sestávajících z 15 hodnot příznakových veličin, 2 hodnot stavu žaluzie, časové známky, důvodu pořízení vzorku a aktivity testovacího režimu. Z toho bylo 36306 pořízeno periodicky v pětiminutových intervalech a 1822 na základě události změny stavu žaluzie - ruční ovládání uživatelem. Příklad 3 sbíraných veličin na časovém úseku jednoho dne (19. dubna 2022) je v grafu na obrázku 7.1.



OBRÁZEK 7.1: Graf sesbíraných hodnot 3 vybraných příznakových veličin z celkových 15. Zobrazena je pouze část dat ze dne 19. dubna 2022

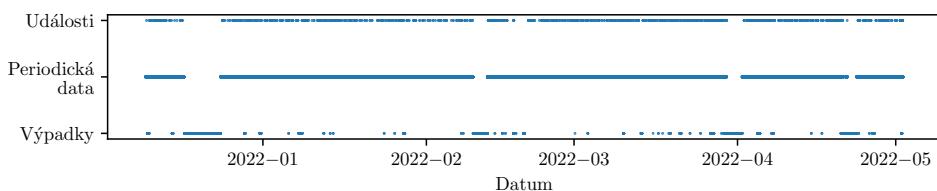
V průběhu sběru dat došlo ke 125 výpadkům při kterých bylo ztraceno 5240 vzorků. Nejčastějším důvodem výpadku byly problémy s připojením k API výrobce žaluzie (Somfy, celkem 105 výpadků). Mezi další příčiny patří výpadky senzorů (odpojení), plný disk databázového serveru (MongoDB) a chyba v systému domácí automatizace (Home Assistant). Poslední dvě

příčiny společně způsobily ztrátu asi 81 % měření z celkového počtu ztracených měření. Největší z výpadků, způsobený Home Assistantem, vznikl ještě před implementací monitoringu přicházejích MQTT zpráv a byl tak odhalen pozdě. Jednou došlo k výpadku připojení k DNS serverům poskytovatele připojení k internetu a nebylo kvůli tomu možné zjistit stav žaluzie. Počet výpadků, jejich důvod a počet při nich ztracených měření shrnuje tabulka 7.1.

Důvod výpadku	Počet výpadků	Počet ztracených vzorků
Somfy	105	383
Senzory	14	605
MongoDB	4	2244
DNS	1	27
Home Assistant	1	1981
Celkem	125	5240

TABULKA 7.1: Výpadky součástí systému automatického ovládání žaluzií při sběru dat a výpadky třetích stran, na kterých je systém závislý.

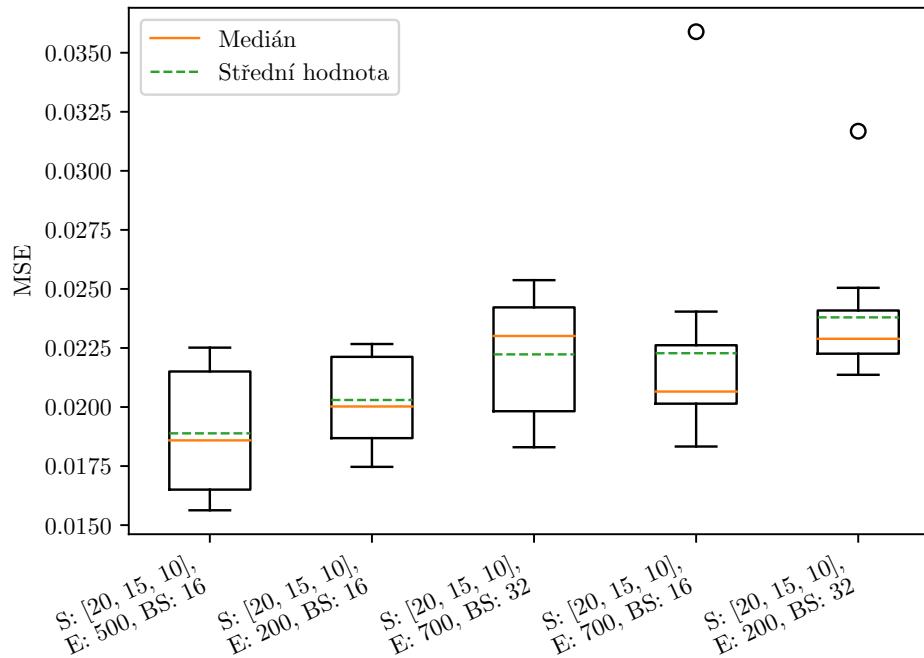
Vzorky podle důvodu sběru (událost, periodické) a výpadky v závislosti na čase jsou v grafu na obrázku 7.2. Každý vzorek je vyobrazen jako bod podle důvodu svého vzniku (Periodická data - vzorky, které byly pořízeny v pětiminutových intervalech; Události - vzorky získané na základě ruční změny stavu žaluzie uživatelem; Výpadky - vzorky, které měly být periodicky pořízeny, ale nebyly).



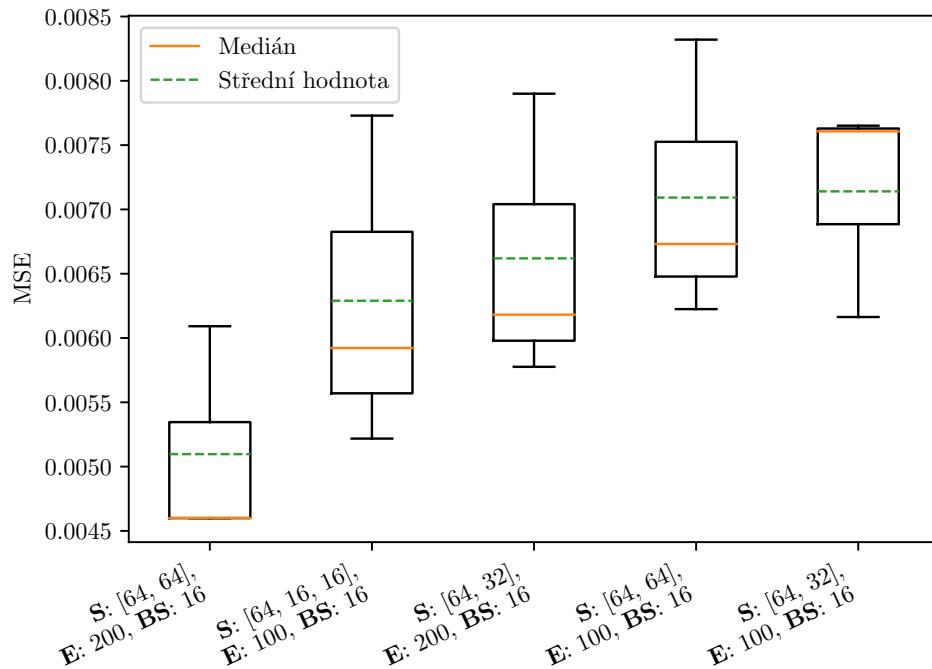
OBRÁZEK 7.2: Graf druhu sesbíraných vzorků dat a výpadků.

7.2 Volba struktury NS a parametrů učení

Na datech sesbíraných ve dnech od 1. prosince 2021 do 17. března 2022 bylo natrénováno 24 dopředných NS (regresor FFNN, sekce 5.3) a 12 rekurentních NS (regresor LSTM, sekce 5.4). Trénování bylo opakováno, aby se snížil vliv náhodné inicializace parametrů sítě, u dopředné NS se trénování opakovalo 10x, u rekurentní 3x (z důvodu úspory času). Struktura sítě, parametry použité při jejím trénování, průměrné dosažené MSE na testovací datové sadě a jeho rozptyl jsou v tabulkách A1.1 a A1.2. Od každé ze sítí byl vytvořen boxplot z realizací prvních 5 nejlepších modelů dle průměrné MSE (obrázky 7.3 a 7.4).



OBRÁZEK 7.3: Boxplot realizací prvních 5 nejlepších modelů
dopředné neuronové sítě (FFNN)



OBRÁZEK 7.4: Boxplot realizací prvních 5 nejlepších modelů
rekurentní neuronové sítě LSTM

Na základě těchto výsledků byly pro další použití v navrhovaném systému automatického ovládání žaluzí zvoleny následující parametry pro konstrukci NS a jejich trénování:

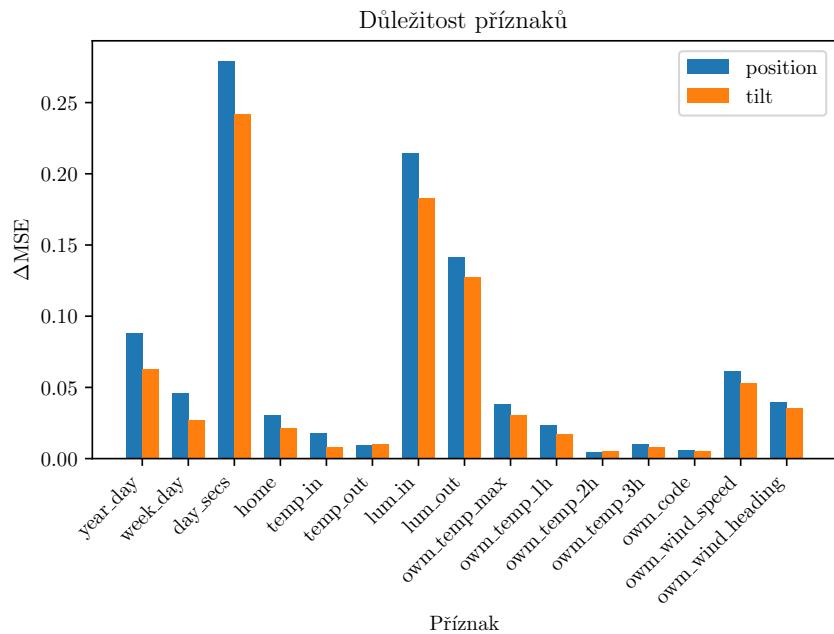
- Dopředná NS (FFNN):
 - 3 skryté vrstvy (po řadě 20, 15 a 10 neuronů);
 - 500 epoch trénování;
 - batch size 16;
 - ztrátová funkce (loss) MSE.
- Dopředná NS (LSTM):
 - 2 skryté vrstvy (po řadě 64 a 64 neuronů);
 - 200 epoch trénování;
 - batch size 16;
 - ztrátová funkce (loss) MSE.

7.3 Vliv jednotlivých příznaků na predikci

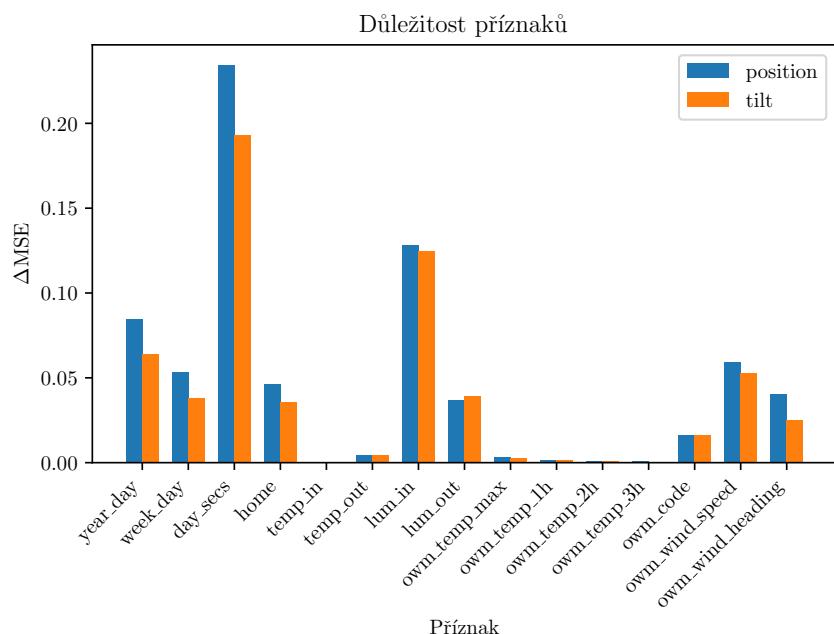
Pokud se označí MSE odhadu \hat{m} ého modelu na testovacích datech jako m a MSE odhadů tohotéž modelu, promíchají-li se náhodně hodnoty i -tého příznaku mezi předkládanými vzorky ($N = 20$ realizací), jako m_{ij}^* , pak důležitost i -tého příznaku v regresorech byla vyjádřena jako:

$$\frac{1}{N} \cdot \sum_{j=1}^N (m_{ij}^* - m).$$

Pro regresor FFNN jsou první 4 nejdůležitější příznaky day_secs, lum_in, lum_out a year_day, pro regresor LSTM pak day_secs, lum_in, year_day a owm_wind_speed. Mezi málo důležité příznaky patří u obou regresorů všechny teploty (temp_in, temp_out, owm_temp_xh pro $x = 1, 2, 3$). Grafy důležitostí všech příznaků pro jednotlivé regresory jsou na obrázcích 7.5 a 7.6.



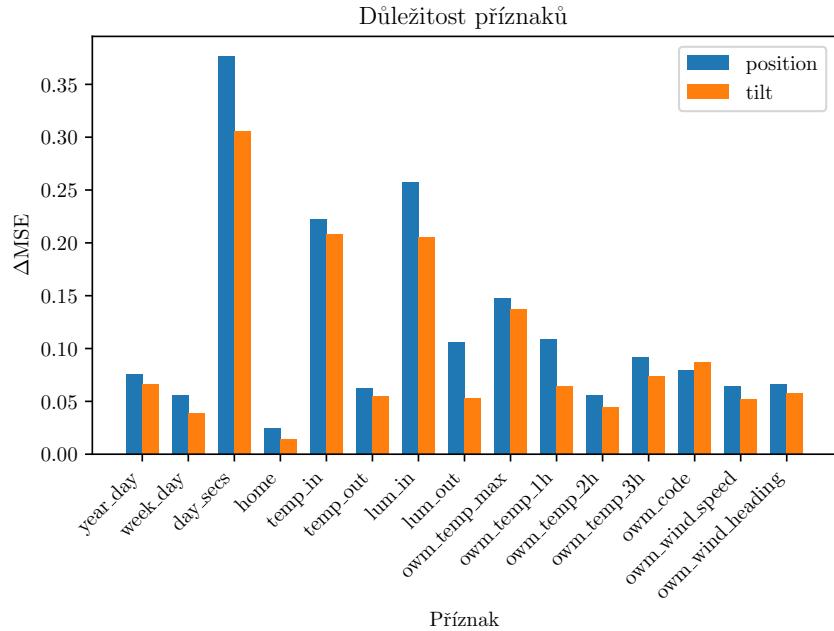
OBRÁZEK 7.5: Permutační důležitost příznaků v regresoru založeném na dopředné neuronové síti v úloze automatického ovládání žaluzie.



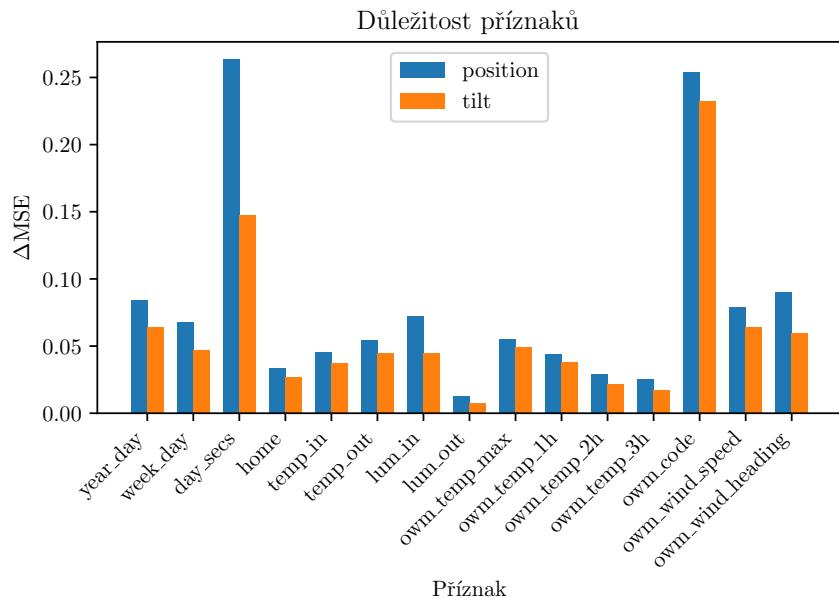
OBRÁZEK 7.6: Permutační důležitost příznaků v regresoru založeném na rekurentní neuronové síti v úloze automatického ovládání žaluzie.

Pro porovnání se ještě využila druhá metoda, která místo míchání hodnot

mezi vzorky všechny hodnoty příslušného příznaku nahradí 0. Její výsledky jsou uvedeny pro každou ze NS na obrázcích 7.7 a 7.8.



OBRÁZEK 7.7: Důležitost příznaků metodou nulování v regresoru založeném na dopředné neuronové síti v úloze automatického ovládání žaluzie.



OBRÁZEK 7.8: Důležitost příznaků metodou nulování v regresoru založeném na rekurentní neuronové síti v úloze automatického ovládání žaluzie.

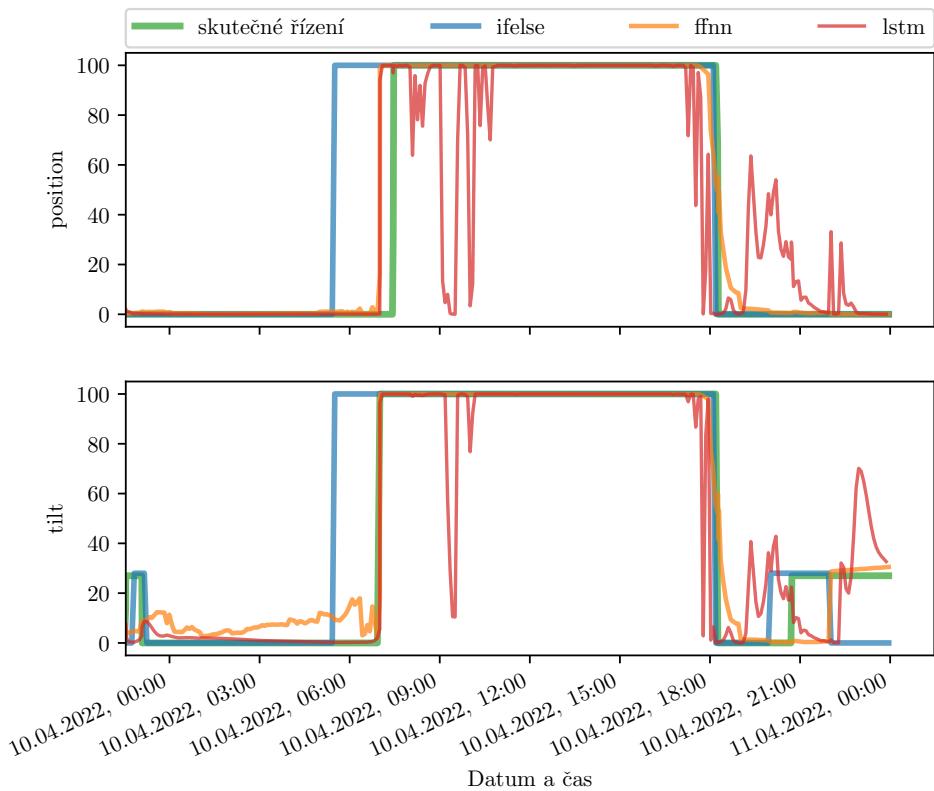
7.4 Porovnání regresorů

Porovnání regresorů použitých v systému automatického ovládání žaluzie (jestliže to vzhledem k jejich návrhu dává smysl) z hlediska počtu parametrů (u Ifelse ručně konfigurované, u FFNN a LSTM odhadnuté pomocí algoritmů strojového učení), doby učení, doby predikce výstupu a dosažené MSE na testovací datové sadě je uvedené v tabulce 7.2. Regresor Ifelse má nejmenší počet parametrů, je 2. v rychlosti predikce a má nejvyšší MSE. Největší počet parametrů, dobu trénování i predikce a nejnižší MSE má regresor LSTM.

	Ifelse	FFNN	LSTM
Počet parametrů	17	817	53634
Doba predikce [ms]	250	30	430
Doba trénování [s]	-	1839	12129
MSE	0,120	0,0246	0,00426

TABULKA 7.2: Porovnání vybraných vlastností regresorů.
Doba predikce platí pro 400 vzorků.

Ukázka predikce obou stavových veličin žaluzie jednotlivými regresory na základě dat sesbíraných 10. dubna 2022 a porovnání se skutečným řízením je v grafu na obrázku 7.9.



OBRÁZEK 7.9: Porovnání predikce 3 regresorů a skutečného řízení na datech z 10. dubna 2022

7.5 Vyhodnocení vlivu retrainingu

Vyhodnocovat retraining má smysl jen u 2 z regresorů, které využívají strojové učení (FFNN a LSTM). Pro každý z nich se porovnaly 2 modely. První z modelů se trénovaly na datech do konce února, druhé pak vznikly z prvních opakovaným trénovaním na datech do konce dubna. Tabulky 7.3 a 7.4 porovnávají MSE od referenčních dat z testovací datové sady pro každý z regresorů celkově i pro každou stavovou proměnnou zvlášť.

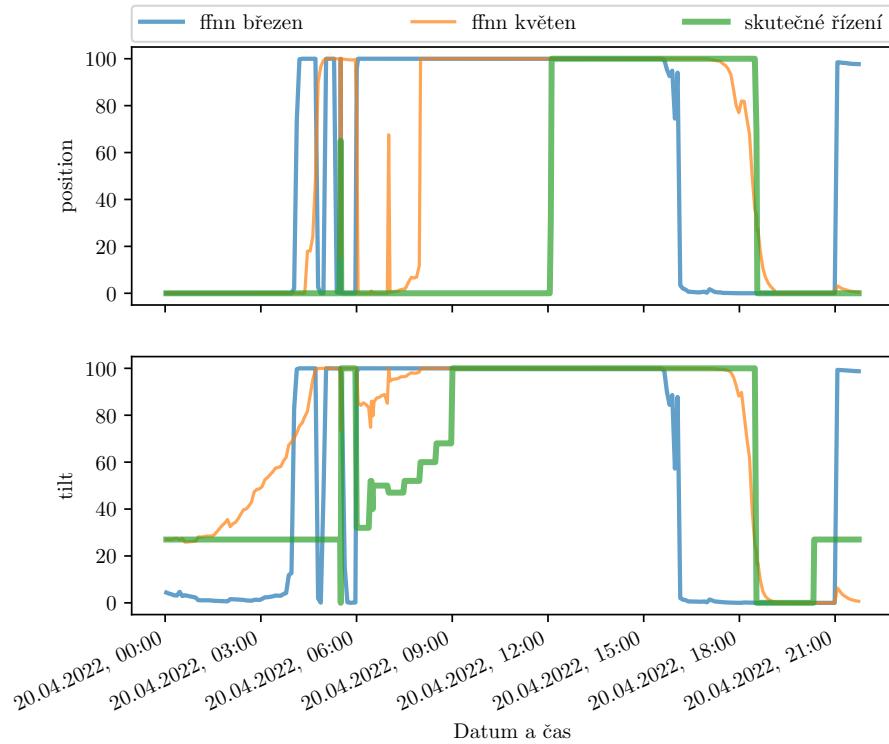
	celkově	position	tilt
březen	0,0962	0,1156	0,0769
květen	0,0246	0,0329	0,0163

TABULKA 7.3: Porovnání středních kvadratických odchylek dvou modelů dopředné neuronové sítě natrénovaných na 2 různých datových sadách z období před daným měsícem.

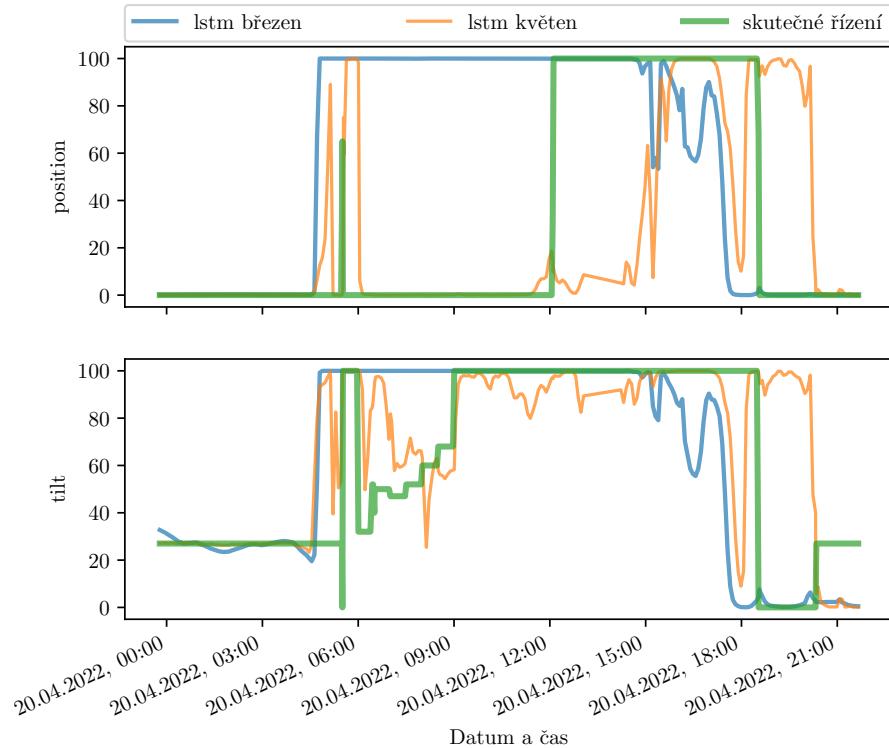
	celkově	position	tilt
březen	0,05911	0,07470	0,04351
květen	0,00426	0,00316	0,00536

TABULKA 7.4: Porovnání středních kvadratických odchylek dvou modelů rekurentní neuronové sítě natrénovaných na 2 různých datových sadách z období před daným měsícem.

Příklad predikce ovládání žaluzií (pro každou z proměnných ve vlastní soustavě souřadné) jednotlivými modely dne 20. dubna 2022 je na obrázcích 7.10 a 7.11.



OBRÁZEK 7.10: Porovnání predikce 2 modelů dopředné neuronové sítě a skutečného řízení na datech z 20. dubna 2022



OBRÁZEK 7.11: Porovnání predikce 2 modelů rekurentní neuronové sítě a skutečného řízení na datech z 20. dubna 2022

8 Diskuze

Cílem této práce bylo navrhnout systém pro automatické ovládání žaluzie na základě preferencí uživatele. Pro základní rozhodovací systém (Ifelse) se zvolil přístup pro vyjádření preferencí uživatele zřetězením jednoduchých pravidel. Další dva regresory navržené pro použití v systému jsou neuronové sítě (FFNN a LSTM), pro jejichž učení je nutné dodat data o skutečném nastavení žaluzí uživatelem a příznacích, na jejichž základě se o nastavení mohl uživatel rozhodovat.

Data se sbírala pomocí měřicích zařízení vlastní kostrukce a z některých zdrojů dostupných na internetu a ukládala se do databáze pro pozdější použití. V průběhu sběru dat docházelo opakovaně k výpadkům. Většina jich byla způsobena nedostupností služby 3. strany, ale největší ztráty měření způsobily vlastní provozované součásti. Jako rizikové se jevilo umístění jednoho z měřicích zařízení na střeše domu, kde bylo vystavené nízkým i vysokým teplotám a zejména dešti. Před zahájením sběru kvůli nedostatečné těsnosti okénka v krabičce došlo k vniknutí vody do zařízení a zničení vstupně-výstupních portů MCU. Po přetěsnění už k podobnému problému nedošlo. Na eliminaci velkých ztrát by mělo pozitivní vliv použití komplexnějšího monitoringu součástí, protože by výpadek bylo možné rychleji odstranit a případně mu i předejít. V průběhu vývoje systému byl nasazen monitoring funkčnosti všech zdrojů dat. Pokud by se zanedbaly výpadky 3. stran a výpadky, které by bylo možné kvůli monitoringu výrazně zkrátit nebo odstranit, ztraceno by bylo jen 1,4 % měření.

Jedním ze zdrojů dat byl systém domácí automatizace, který poskytoval informace o nastavení žaluzie z API výrobce, protože se v počátcích vývoje nedářilo získání přístupu k API běžným způsobem. Později se to však vedlo nedokumentovanou cestou a místo systému domácí automatizace by tak bylo možné používat knihovnu `Pymf` pro přímý přístup k API.

Za účelem zvolení vhodných modelů neuronových sítí a vhodných parametrů učení pro tuto úlohu se prohledávaly **všechny** kombinace zvolených parametrů (sekce 5.3 a 5.4) a zkoušely se s nimi trénovat neuronové sítě. Poté se vybrala nejlepší kombinace od obou použitých druhů NS ve smyslu průměrné MSE přes všechny realizace. Statistické vyhodnocení ukazuje, že i přes náhodnou inicializaci modelů je největší pravděpodobnost, že tyto zvolené modely a parametry jejich učení budou nejlepší z prohledávaných.

Každý z příznaků může mít různě významný vliv na hodnoty odhadované regresorem. Pro kvantifikaci tohoto vlivu byla využita metoda *Permutation Feature Importance* (sekce 2.1). Výsledky (sekce 7.3) napovídají, že by mohlo být možné redukovat příznakový vektor, protože některé příznaky mají na odhad zanedbatelný vliv a úloha by se tak zjednodušila.

Jeden z nejdůležitějších příznaků je v obou případech NS příznak `lum_in`. Problémem tohoto příznaku je skutečnost, že je silně ovlivňován nastavením žaluzie. V případě reálného nasazení systému tak hrozí, že by sítě nemusely vygenerovat žádný akční zásah. Přestože je tedy příznak důležitý pro predikci v navržených modelech, mohlo by být lepší ho vynechat z příznakového vektoru.

Důležitou vlastností navrhovaného systému je retraining modelů na nových datech, která jsou průběžně sbírána (sekce 7.5). Za účelem zjištění, jaký vliv na odhad poskytovaný regresory má právě přetrénování, byly pro každou z použitých NS porovnány 2 modely, z nichž jeden přetrénováním druhého na čerstvějších data (mladší model). Na testovací datové sadě byl patrný výrazný rozdíl v MSE ve prospěch modelů s čerstvějšími datech (tabulky 7.3 a 7.4), v ukázce predikce starších a novějších modelů na reálných datech (obrázky 7.10 a 7.11) je u těchto modelů pak zřejmá reakce na zvýšené riziko přehráti interiéru v dopoledních hodinách, kdy by květnové modely narození od březnových ponechaly žaluzii zataženou a sklápěly by lamely.

8.1 Monosti reálného nasazení systému

NS doporučované řízení není vhodné pro přímé použití při řízení žaluzie. Její nastavení by se totiž příliš často nevhodně měnilo a pro uživatele by používání systému kvůli tomu nemuselo být pohodlné. Pro vylepšení poskytovaných výsledků by mohlo být vhodné nasbírat větší množství dat a dále zkoumat možné způsoby trénování NS a jejich strukturu, případně poskytované odhady vhodně filtrovat nebo dále zpracovávat.

9 Závěr

V rámci této práce byl vyvinut systém, který v reálném čase sbírá hodnoty příznaků a aktuální uživatelské nastavení žaluzie, pomocí 3 různých regresorů odhaduje vhodné nastavení žaluzie a umožňuje řízení generované jednotlivými regresory porovnat. NS se v systému periodicky přetrénovávají (retraining), což má pozitivní vliv na přesnost jejich odhadu, vyskytnou-li se v datech nové skutečnosti.

Hardware součásti použité v měřicích zařízeních se osvědčily, stejně jako datové spojení s ostatními součástmi systému. Bylo by vhodné monitorovat všechny součásti i jejich spojení a případné problémy nejlépe automaticky odstraňovat, nebo hlásit uživateli. V systému se uživateli hlásí jen to, když nepřijdou data z některého zdroje dat v okamžiku pořizování vzorku.

V úloze automatického ovládání žaluzií může jejich stav záviset nejen na aktuálních podmínkách ale také na podmínkách, které panovaly v historii. Kvůli tomu je vhodné využít regresor, který historické podmínky uvažuje při svém odhadu. V této práci tuto vlastnost splňuje regresor založný na rekurentní neuronové síti s LSTM buňkami.

Přesto, že některé veličiny mohou být silně svázány s odhadovanými, nemusí být dobrými příznaky pro reálné využití, pokud je odhad řízení při jeho použití zpětně ovlivňuje.

Díky dalšímu sběru dat společně s přetrénováním NS by regresory mohly do budoucna poskytovat postupně lepší výsledky. V případě skutečného nasazení by bylo nutné ošetřit konfliktní stavy uživatelem vyžádaného nastavení žaluzie a systémem odhadnutým nastavením tak, aby interakce se systémem byla pro uživatele přívětivá.

Bibliografie

- [1] Augustin Cauchy. „Méthode générale pour la résolution des systèmes d'équations simultanées“. In: *Comptes Rendus de l'Académie des sciences* 25.55 (pros. 1847), s. 536–538. URL: <https://gallica.bnf.fr/ark:/12148/bpt6k2982c/>.
- [2] Warren S McCulloch a Walter Pitts. „A logical calculus of the ideas immanent in nervous activity“. In: *The bulletin of mathematical biophysics* 5.4 (1943), s. 115–133.
- [3] F. Rosenblatt. „The perceptron: A probabilistic model for information storage and organization in the brain.“ In: *Psychological Review* 65.6 (1958), s. 386–408. DOI: 10.1037/h0042519. URL: <https://doi.org/10.1037/h0042519>.
- [4] Seppo Linnainmaa. „The representation of the cumulative rounding error of an algorithm as a Taylor expansion of the local rounding errors“. Dis. pr. Diplomová práce, Univ. Helsinki, 1970.
- [5] David E Rumelhart, Geoffrey E Hinton a Ronald J Williams. „Learning representations by back-propagating errors“. In: *nature* 323.6088 (1986), s. 533–536.
- [6] Christopher M. Bishop. *Neural networks for pattern recognition*. Oxford University Press, 1995.
- [7] Sepp Hochreiter a Jürgen Schmidhuber. „Long Short-term Memory“. In: *Neural computation* 9 (pros. 1997), s. 1735–80. DOI: 10.1162/neco.1997.9.8.1735.
- [8] Charles Dugas et al. „Incorporating Second-Order Functional Knowledge for Better Option Pricing“. In: *Advances in Neural Information Processing Systems*. Ed. T. Leen, T. Dietterich a V. Tresp. Sv. 13. MIT Press, 2000, s. 472–478. URL: <https://proceedings.neurips.cc/paper/2000/file/44968aece94f667e4095002d140b5896-Paper.pdf>.
- [9] Felix A. Gers, Jürgen Schmidhuber a Fred Cummins. „Learning to Forget: Continual Prediction with LSTM“. In: *Neural Computation* 12.10 (2000), s. 2451–2471. DOI: 10.1162/089976600300015015.
- [10] Leo Breiman. „Random Forests“. In: *Machine Learning* 45 (říj. 2001), s. 5–32. DOI: 10.1023/A:1010950718922.
- [11] Vinod Nair a Geoffrey E. Hinton. „Rectified Linear Units Improve Restricted Boltzmann Machines“. In: *ICML*. Ed. Johannes Fürnkranz a Thorsten Joachims. Omnipress, 2010, s. 807–814. URL: <https://icml.cc/Conferences/2010/papers/432.pdf>.
- [12] Erik Dahlström et al. *Somfy Open API*. 2011. URL: <https://www.w3.org/TR/SVG11/> (cit. 21. 04. 2022).
- [13] I. Fette a A. Melnikov. *The WebSocket Protocol*. RFC 6455. <http://www.rfc-editor.org/rfc/rfc6455.txt>. RFC Editor, 2011. URL: <http://www.rfc-editor.org/rfc/rfc6455.txt>.

- [14] Štěpánka Lubinová. *Stínění oken: žaluzie, rolety, markýzy a slunolamy*. Grada Publishing, 2013. ISBN: 978-80-247-4579-4.
- [15] Adafruit Industries. *Adafruit TSL2591 High Dynamic Range Digital Light Sensor*. 2014.
- [16] Damien P. George a Paul Sokolovsky. *The MicroPython Documentation*. 2014. URL: <http://docs.micropython.org/en/latest/> (cit. 23.03.2022).
- [17] OASIS Open. *MQTT Version 3.1.1*. Ed. Andrew Banks a Rahul Gupta. Říj. 2014. URL: <http://docs.oasis-open.org/mqtt/mqtt/v3.1.1/mqtt-v3.1.1.html> (cit. 28.03.2022).
- [18] AI-Thinker Team. *ESP-12E WiFi Module*. Version 1.0. 2015.
- [19] Prajit Ramachandran, Barret Zoph a Quoc V Le. „Searching for activation functions“. In: *arXiv preprint arXiv:1710.05941* (2017).
- [20] Pradyumna Gokhale, Omkar Bhat a Sagar Bhat. „Introduction to IOT“. In: *International Advanced Research Journal in Science, Engineering and Technology* 5.1 (2018), s. 41–44.
- [21] Chigozie Nwankpa et al. „Activation Functions: Comparison of trends in Practice and Research for Deep Learning“. In: *CoRR* abs/1811.03378 (2018). arXiv: 1811 . 03378. URL: <http://arxiv.org/abs/1811.03378>.
- [22] Dan Popa et al. „Deep learning model for home automation and energy reduction in a smart home environment platform“. In: *Neural Computing and Applications* 31.5 (zář. 2018), s. 1317–1337. DOI: 10.1007/s00521-018-3724-6. URL: <https://doi.org/10.1007/s00521-018-3724-6>.
- [23] Somfy. *Somfy Open API Developer Portal / APIs & Docs*. 2018. URL: <https://developer.somfy.com/apis-docs> (cit. 23.03.2022).
- [24] Stefan Wendler. *MPFShell*. 2018. URL: <https://github.com/wendlers/mpfshell> (cit. 23.03.2022).
- [25] Maxim Integrated Products, Inc. *DS18B20 – Programmable Resolution 1-Wire Digital Thermometer*. 19-7487; Rev 6; 7/19. 2019.
- [26] John Jaihar et al. „Smart Home Automation Using Machine Learning Algorithms“. In: *2020 International Conference for Emerging Technology (INCET)*. 2020, s. 1–4. DOI: 10 . 1109 / INCET49848 . 2020 . 9154007.
- [27] Thibaut Etienne. *Somfy Open API*. 2021. URL: <https://github.com/tetienne/somfy-open-api#readme> (cit. 23.03.2022).
- [28] MongoDB, Inc. *MongoDB Documentation*. 2021. URL: <https://www.mongodb.com/docs/> (cit. 20.04.2022).
- [29] Apple Inc. *Apple Home*. 2022. URL: <https://www.apple.com/ios/home/> (cit. 21.04.2022).
- [30] Home Assistant. *Home Assistant*. 2022. URL: <https://www.home-assistant.io/> (cit. 21.04.2022).
- [31] Keras. *Keras API reference*. 2022. URL: <https://keras.io/api/> (cit. 21.04.2022).
- [32] OpenHAB Foundation. *OpenHAB*. 2022. URL: <https://www.openhab.org/> (cit. 21.04.2022).
- [33] Vercel. *Next.js*. Vercel. 2022. URL: <https://nextjs.org/> (cit. 21.04.2022).

A1 Prohledávání parametrů konstrukce a trénování NS

V tabulkách A1.1 a A1.2 jsou uvedeny parametry pro konstrukci a trénování neuronových sítí (rekurentní a dopředné), které byly prohledávány za účelem získání co nejlepšího modelu v úloze automatického ovládání žaluzie na základě 15 příznaků, a průměrná (ze 3, resp. z 10 realizací) střední kvadratická odchylka na datové sadě, která nebyla využita při jejich učení.

	Struktura	E	BS	MSE	Var
1	[64, 64]	200	16	0,00510	4,95E-07
2	[64, 16, 16]	100	16	0,00629	1,12E-06
3	[64, 32]	200	16	0,00662	8,48E-07
4	[64, 64]	100	16	0,00709	7,97E-07
5	[64, 32]	100	16	0,00714	4,78E-07
6	[64, 64]	200	64	0,00740	6,88E-07
7	[64, 32]	200	64	0,00770	2,87E-07
8	[64, 16, 16]	200	16	0,00770	5,33E-07
9	[64, 64]	100	64	0,00793	1,12E-07
10	[64, 16, 16]	100	64	0,00900	3,48E-07
11	[64, 32]	100	64	0,00939	2,35E-06
12	[64, 16, 16]	200	64	0,00985	3,17E-07

TABULKA A1.1: Výsledky prohledávání parametrů rekurentní neuronové sítě a jejího učení pro použití v systému automatického ovládání žaluzí. Sloupce: Struktura sítě, E - počet epoch trénování, BS - velikost dávky (batch size), MSE - střední kvadratická odchylka na testovací datové sadě.

	Struktura	E	BS	MSE	Var
1	[20, 15, 10]	500	16	0,01889	6,554E-06
2	[20, 15, 10]	200	16	0,02030	3,371E-06
3	[20, 15, 10]	700	32	0,02223	5,928E-06
4	[20, 15, 10]	700	16	0,02228	2,329E-05
5	[20, 15, 10]	200	32	0,02380	8,069E-06
6	[20, 15, 10]	500	32	0,02407	1,181E-05
7	[10, 15, 20]	500	16	0,02480	1,479E-05
8	[10, 15, 10]	500	16	0,02488	9,936E-06
9	[10, 15, 20]	200	16	0,02503	6,469E-06
10	[10, 15, 20]	700	16	0,02630	4,822E-05
11	[10, 15, 20]	500	32	0,02690	4,218E-05
12	[10, 15, 20]	200	32	0,02783	8,611E-06
13	[15, 10]	200	16	0,03088	7,698E-05
14	[15, 10]	500	16	0,03132	4,879E-05
15	[10, 15, 10]	700	16	0,03162	8,363E-05
16	[10, 15, 10]	500	32	0,03206	1,151E-04
17	[10, 15, 10]	200	16	0,03430	1,274E-04
18	[15, 10]	700	16	0,03564	1,248E-04
19	[10, 15, 10]	200	32	0,03611	9,818E-05
20	[10, 15, 20]	700	32	0,03637	3,047E-05
21	[15, 10]	500	32	0,03780	9,898E-05
22	[10, 15, 10]	700	32	0,03887	1,048E-04
23	[15, 10]	700	32	0,03952	9,978E-05
24	[15, 10]	200	32	0,04309	1,854E-04

TABULKA A1.2: Výsledky prohledávání parametrů dopředné neuronové sítě a jejího učení pro použití v systému automatického ovládání žaluzí. Sloupce: Struktura sítě, E - počet epoch trénování, BS - velikost dávky (batch size), MSE - střední kvadratická odchylka na testovací datové sadě.