Modelowanie memrystorów SDC za pomocą sztucznych sieci neuronowych

Karol Bednarz

Rok akademicki 2024–2025

Spis treści

1 Wstęp

W naukowej literaturze pojawiło się wiele modeli memrystorów po odkryciu ich zachowań memrystywnych na nanoskalę. Oryginalny model, zaproponowany w [8], konceptualizuje memrystor jako połączenie szeregowe dwóch zmiennych rezystancji odpowiadających warstwom przewodzącym i izolacyjnym filmu. W celu lepszego modelowania memrystorów SDC (ang. Self-Directed Channel), M. Nugent i T. Molter zaproponowali uogólniony model Mean Metastable Switch, który jest modelem półempirycznym. Pochodna zmiennej stanu jest tu określana jako funkcja prawdopodobieństw przejścia do kolejnych stanów i aktualnej wartości zmiennej stanu [6]. W pracy [5] autorzy podjęli próbę modelowania memrystora z wykorzystaniem podejścia opartych na Physics-Informed Neural Networks (PINNs). Mimo że przedstawione tam wyniki wskazują na potencjał tej metody, badania nie opierają się na rzeczywistych pomiarach fizycznych memrystorów, lecz jedynie na porównaniach z wynikami uzyskanymi z istniejących modeli symulacyjnych. Co więcej, dane treningowe użyte w procesie uczenia sieci neuronowej zostały wygenerowane na podstawie wcześniej opracowanych matematycznych modeli teoretycznych, co ogranicza możliwość weryfikacji skuteczności podejścia w kontekście rzeczywistych układów fizycznych.

W pracy [3] autorzy przedstawiają koncepcję głębokich modeli neuronowych, w których dynamika ukrytego stanu opisywana jest za pomocą równania różniczkowego zwyczajnego (ODE, ang. Ordinary Differential Equation). Uczenie modelu przebiega w sposób end-to-end, co oznacza optymalizację wszystkich parametrów jednocześnie w ramach jednego procesu treningowego. Kluczowym

elementem proponowanego podejścia jest nowa metoda propagacji wstecznej błędu, oparta na rozwiązywaniu równania różniczkowego wstecz w czasie z wykorzystaniem metod sprzężonych (ang. adjoint methods). Rozwiązanie to znajduje zastosowanie m.in. w ciągłych odpowiednikach sieci rezydualnych (continuous-depth residual networks) oraz generatywnych modelach przepływowych (generative flow-based models).

Z kolei praca [2] rozszerza klasyczne modele Neural ODE, umożliwiając modelowanie nieciągłych zdarzeń w czasie ciągłym — bez uprzedniej wiedzy na temat ich liczby ani momentu wystąpienia. Wprowadzone przez autorów różniczkowalne funkcje zdarzeń pozwalają na efektywną symulację systemów hybrydowych, takich jak układy z kolizjami, przełącznikami stanów czy procesami punktowymi, co otwiera nowe możliwości w zakresie modelowania i trenowania układów z dyskretnym sterowaniem.

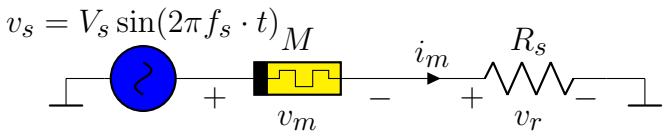
TODO: Dodać więcej informacji o pracy, w tym o zastosowaniach i wynikach.

Niniejsza praca ma na celu zbadanie możliwości modelowania memrystorów SDC przy użyciu sztucznych sieci neuronowych, oraz porównanie skuteczności tego podejścia z istniejącymi modelami teoretycznymi. W szczególności, skupimy się na zastosowaniu modeli Neural ODE do symulacji zachowania memrystorów SDC, wykorzystując dane eksperymentalne uzyskane z rzeczywistych układów fizycznych. Naszym celem jest ocena, czy modele oparte na sztucznych sieciach neuronowych mogą skutecznie odwzorować dynamikę memrystorów SDC i czy oferują przewagę w porównaniu do tradycyjnych modeli teoretycznych.

2 Materiały i metody

2.1 Memrystory, oraz pomiary

W niniejszej pracy wykorzystano dane eksperymentalne pozyskane z rzeczywistych memrystorów typu SDC (ang. Self Directed Channels) domieszkowanych wolframem. Struktura oraz właściwości tych elementów zostały szczegółowo opisane w literaturze, m.in. w [4, 1]. Pomiary przeprowadzono na układzie przedstawionym na Rys. 1, który składa się z memrystora połączonego szeregowo z rezystorem. Napięcie zasilania było generowane przez generator arbitralny, a pomiary napięcia były dokonywane za karty pomiarowej. Pomiary zostały przeprowadzone dla różnych kombinacji częstotliwości oraz amplitud napięcia zasialania, gdzie amplituda niępiacia zasilania $V_s \in \{0.5, 1, 1.5\}$[V] oraz częstotliwość $f \in \{1, 5, 10, 20, 50, 100\}$ [Hz].



Rysunek 1: Schemat układu pomiarowego memrystora SDC.

2.2 Modelowanie za pomocą sieci neuronowej

Analizowany model opiera się na ogólnym opisie elementu memrystywnego, wyrażonym za pomocą równań (1), (2)

$$v(t) = \mathcal{M}\Big(x(t), v(t)\Big) i(t), \tag{1}$$

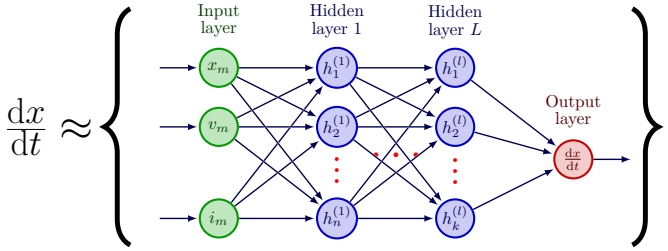
$$\frac{\mathrm{d}x}{\mathrm{d}t} = f\Big(x(t), v(t)\Big) \tag{2}$$

gdzie $x \in \langle 0, 1 \rangle$ oznacza wewnętrzną zmienną stanu, której ewolucja przebiega zgodnie z określoną dynamiką f, natomiast $\mathcal{M}$ reprezentuje memrystancję urządzenia. Kluczowym zagadnieniem modelowania memrystora jest odpowiednie wyznaczenie funkcji $\mathcal{M}(x, v)$ oraz $f(x, v)$.

W przeprowadzonych badaniach przyjęto powszechnie stosowaną postać funkcji memrystancji:

$$\mathcal{M}(x) = xR_{\mathrm{ON}} + (1 - x)R_{\mathrm{OFF}}, \tag{3}$$

gdzie R_{ON} oraz R_{OFF} odpowiadają wartościom rezystancji w stanach niskiej i wysokiej przewodności. Funkcja $f(x(t), v(t))$ została zaimplementowana w postaci sztucznej sieci neuronowej, której ideowa architektura przedstawiona została na Rys. 2. W trakcie procesu optymalizacji sieci, równocześnie dostrajane są jej parametry (wagi i biasy) oraz wartości R_{ON} i R_{OFF}.



Rysunek 2: Diagram ideowy zastosowania sieci neuronowej do symulacji dynamiki memrystora.

2.3 Model MMS - Mean Metastable Switch

W celu porównania skuteczności modelu, porównano go z deterministycznym modelem Mean Metastable Switch (MMS) zaproponowanym w pracach [6, 7]. Model ten opisuje dynamikę memrystora za pomocą równań (4), (5).

$$\frac{\mathrm{d}x}{\mathrm{d}t} = \frac{1}{\tau}\left(\frac{1}{1 + e^{-\beta\left(v(t) - V_{\mathrm{ON}}\right)}}(1 - x) - \left(1 - \frac{1}{1 + e^{-\beta\left(v(t) - V_{\mathrm{OFF}}\right)}}\right)x\right) \tag{4}$$

$$\beta = \frac{q}{kT} \tag{5}$$

gdzie, β jest parametrem temperaturowym, V_{ON} napięciem przełączania w stan LRS, V_{OFF} napięciem przełączania w stan HRS, k stałą Boltzmanna, T temperaturą, q ładunkiem elementarnym.

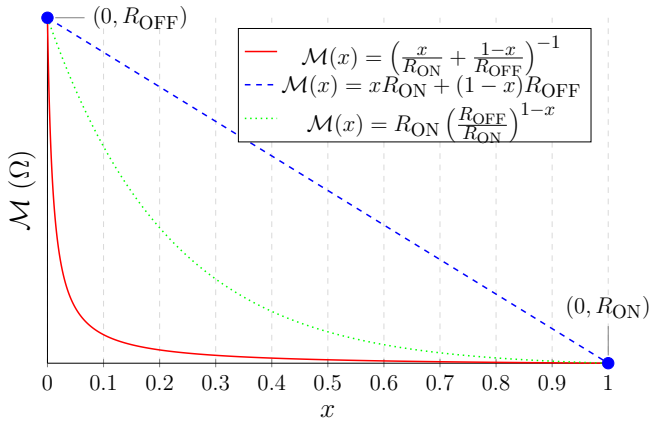
Chwilowa konduktancja memrystora jest określona jako:

$$\mathcal{G}_m(x) = \frac{x}{R_{\mathrm{ON}}} + \frac{1 - x}{R_{\mathrm{OFF}}} \tag{6}$$

Gdzie R_{ON} odpowiada rezystancji w stanie LRS, a R_{OFF} rezystancji w stanie HRS. Inne funkcje opisywane w literaturze, to między innymi:

- $\mathcal{G}_m(x) = (xR_{\mathrm{ON}} + (1 - x)R_{\mathrm{OFF}})^{-1}$
- $\mathcal{G}_m(x) = \frac{1}{R_{\mathrm{ON}}}\left(\frac{R_{\mathrm{OFF}}}{R_{\mathrm{ON}}}\right)^{x-1}$

Wpływ tych funkcji na rezystancje został przedstawiony na Rys. 3.



Rysunek 3: Porównanie funcki rezystancji $R_m(x)$ dla różnych modeli memrystora.

W pracy [6] autorzy również model GMMS (Generalized Mean Metastable Switch), który opisuje memrystor jako równoległe połączenie dwóch memrystora wraz z diodą Schottky'ego. Model ten jest bardziej złożony i uwzględnia nieliniowość charakterystyki prądowo-napięciowej, co pozwala na lepsze odwzorowanie rzeczywistych zachowań memrystora w różnych warunkach pracy.Wprowadzone jednak dodatkowe parametry oraz nieliniowość sprawiają, że model ten jest bardziej skomplikowany i trudniejszy do optymalizacji. Pochodna zmiennej stanu koresponduje z rów. (4), jednak zależność prądowo napięciowa jest opisana równaniami:

$$i(t) = \phi i_m(v,t) + (1-\phi) i_d(v) \tag{7}$$

$$i_d(v) = \alpha_f e^{\beta_f v} - \alpha_r e^{-\beta_r v} \tag{8}$$

$$i_m(v,t) = \mathcal{G}_m(x) v \tag{9}$$

gdzie $\phi \in [0,1]$, jest to parametr określający udział prądu płynącego przez memrystor w całkowitym prądzie płynącym przez element. Parametry α_f, β_f, α_r oraz β_r są parametrami dodatnimi, które określają zachowanie prądów w kierunku przewodzenia i zaporowym, wzdłuż bariery Schottky'ego.

W przypadku modelu GMMS, wyzwaniem jest również, ze względu na efekt Schottky'ego, konieczność uwzględnienia nieliniowości charakterystyki prądowo-napięciowej, co przy szeregowym połaczeniu memrystora z rezystorem, daje konieczność rozwiązania numerycznie nieliniowego równania w postaci:

$$v_s(t) = v_r(t) + v_m(t) \tag{10}$$

gdzie, $v_s(t)$ to napięcie zasilania, $v_r(t) = R_s i(t)$ to napięcie na rezystorze, a $v_m(t)$ to napięcie na memrystorze, czyni to jednak układ sztywnym, oraz nie pozwala na automatyczne różniczkowanie względem parametrów, dla funckji celu. Drugą metodą jest zastosowanie różniczkowania implikowanego. Niech F będzie funkcją określoną równaniem:

$$F(v_m, t) = v_s(t) - v_r(v_m, t) - v_m(t) = 0 \tag{11}$$

Korzystając z regułu łańcuchowej:

$$\frac{\mathrm{d}F}{\mathrm{d}t} = \frac{\partial F}{\partial v}\frac{\mathrm{d}v}{\mathrm{d}t} + \frac{\partial F}{\partial t} = 0 \tag{12}$$

Zatem:

$$\frac{\mathrm{d}v}{\mathrm{d}t} = -\frac{\partial F/\partial t}{\partial F/\partial v} \tag{13}$$

Podstawiajac równania:

$$\frac{\mathrm{d}v_m}{\mathrm{d}t} = -\frac{\mathrm{d}v_s}{\mathrm{d}t}\frac{1}{-R_s\left(\mathcal{G}_m\phi + (1-\phi)\left(\alpha_f\beta_f e^{\beta_f v_m} + \alpha_r\beta_r e^{-\beta_r v_m}\right)\right) - 1} \tag{14}$$

Tak więc układ wymaga dodatkowej zmiennej stanu, lecz nie jest konieczne rozwiązywanie nieliniowego układu równań z każdym krokiem czasowym. Tak więc układ równań opsisujący układ memrystora z rezystorem, przy zastosowaniu modelu GMMS, jest następujący:

$$\begin{cases} \dfrac{\mathrm{d}x}{\mathrm{d}t} = \dfrac{1}{\tau}\left(\dfrac{1}{1+e^{-\beta\left(v_m(t)-V_{\mathrm{ON}}\right)}}(1-x) - \left(1 - \dfrac{1}{1+e^{-\beta\left(v_m(t)-V_{\mathrm{OFF}}\right)}}\right)x\right) \\ \dfrac{\mathrm{d}v_m}{\mathrm{d}t} = \dfrac{\mathrm{d}v_s}{\mathrm{d}t}\dfrac{1}{R_s\left(\mathcal{G}_m\phi + (1-\phi)\left(\alpha_f\beta_f e^{\beta_f v_m} + \alpha_r\beta_r e^{-\beta_r v_m}\right)\right) - 1} \end{cases} \tag{15}$$

2.4 Funkcja celu

Funkcja straty oparta na portrecie fazowym została zaprojektowana w taki sposób, aby jednocześnie uwzględniać zarówno wielkości bezwzględne sygnału, jak i jego strukturę dynamiczną. W tym celu definiuje się ją jako sumę czterech składników, z których każdy pełni odmienną, komplementarną rolę w procesie optymalizacji. Całkowita funkcja straty przyjmuje postać, jak w równaniu (16):

$$\mathcal{L}_{\mathrm{total}} = \mathcal{L}_{\mathrm{base}} + \lambda_1 \cdot \mathcal{L}_v + \lambda_2 \cdot \mathcal{L}_c + \lambda_{\mathrm{con}} \cdot \mathcal{L}_{\mathrm{con}}(x) \tag{16}$$

gdzie $\mathcal{L}_{\mathrm{base}}$ stanowi podstawowy składnik błędu, zazwyczaj odnoszący się do różnicy między wartościami przewidywanymi a referencyjnymi (np. błąd średniokwadratowy – MSE), $\mathcal{L}_v$ odpowiada za zachowanie poprawnej charakterystyki prędkości zmian, tj. pochodnych trajektorii w przestrzeni fazowej, $\mathcal{L}_c$ odpowiada za zgodność trajektorii w kontekście zakrzywień (np. poprzez analizę krzywizny lub zmian kierunku trajektorii), $\mathcal{L}_{\mathrm{con}}(x)$ wprowadza dodatkowe ograniczenia fizyczne lub strukturalne, które powinny być spełnione przez model, np. warunki stabilności, nieliniowości bądź zgodności z modelem fizycznym. Współczynniki wagowe λ_1, λ_2 oraz λ_{con} służą do odpowiedniego zbalansowania wkładu poszczególnych składników w całkowitej funkcji straty, w zależności od priorytetów modelowania. Tak skonstruowana funkcja pozwala na lepsze odwzorowanie zarówno ilościowych, jak i jakościowych aspektów dynamiki badanego systemu.

Dla trajektorii obwodu oznaczonych indeksem $k = 1, \ldots, N_{\mathrm{traj}}$, każda zmienna $z \in \{v, i\}$, reprezentująca odpowiednio napięcie lub prąd, jest normalizowana względem odchylenia standardowego wyznaczonego osobno dla każdej trajektorii:

$$\sigma_{z,k} = \sqrt{\frac{1}{T-1}\sum_{t=1}^{T}\left(z_{k,t}^{\mathrm{true}} - \bar{z}_k^{\mathrm{true}}\right)^2} \tag{17}$$

gdzie średnia trajektorii wynosi:

$$\bar{z}_k^{\mathrm{true}} = \frac{1}{T}\sum_{t=1}^{T} z_{k,t}^{\mathrm{true}}, \tag{18}$$

a T oznacza długość danej trajektorii (liczbę punktów czasowych).

Tak przeprowadzona normalizacja zmiennych w obrębie każdej trajektorii pozwala na ujednolicenie skali sygnałów niezależnie od ich amplitudy, co ma istotne znaczenie w kontekście porównywania dynamicznych kształtów przebiegów oraz zapewnienia stabilności procesu uczenia.

Podstawowy składnik funkcji straty odpowiada za dopasowanie modelu do rzeczywistych danych napięciowo-prądowych, przy uwzględnieniu normalizacji względem trajektorii. Ma on na celu

minimalizację błędu średniokwadratowego (MSE) między przewidywanymi a rzeczywistymi przebiegami napięcia i prądu, znormalizowanymi względem odchylenia standardowego każdej trajektorii. Formalnie wyraża się to jako:

$$\mathcal{L}_{\text{base}} = \frac{1}{N_{\text{traj}}} \sum_{k=1}^{N_{\text{traj}}} \left[\text{MSE}\left(\frac{v_{\text{pred},k}}{\sigma_{v,k}}, \frac{v_{\text{true},k}}{\sigma_{v,k}} \right) + \text{MSE}\left(\frac{i_{\text{pred},k}}{\sigma_{i,k}}, \frac{i_{\text{true},k}}{\sigma_{i,k}} \right) \right]$$

Porównanie dynamiki pierwszego rzędu pozwala na ocenę zgodności prędkości zmian napięcia i prądu między przewidywaniami modelu a rzeczywistymi danymi, zwłaszcza w kontekście szybkiego przełączania stanów memrystora. Dzięki temu model jest uczony nie tylko na podstawie wartości bezwzględnych sygnałów, ale również ich tempa zmian, co jest szczególnie istotne w przypadku systemów dynamicznych. Jest definiowany jako:

$$\mathcal{L}_v = \text{MSE}\left(\frac{\mathrm{d}v}{\mathrm{d}t}_{\text{pred}}, \frac{\mathrm{d}v}{\mathrm{d}t}_{\text{true}} \right) + \text{MSE}\left(\frac{\mathrm{d}i}{\mathrm{d}t}_{\text{pred}}, \frac{\mathrm{d}i}{\mathrm{d}t}_{\text{true}} \right) \tag{19}$$

Porównanie drugiego rzędu, czyli krzywizny trajektorii, jest kluczowe dla uchwycenia subtelnych zmian w dynamice systemu, które mogą być istotne w kontekście nieliniowych zachowań memrystora. Składnik ten pozwala na lepsze odwzorowanie właściwości krzywizny trajektorii w przestrzeni fazowej, co jest istotne przy modelowaniu systemów nieliniowych i silnie dynamicznych. Jest definiowany jako:

$$\mathcal{L}_c = \text{MSE}\left(\frac{\mathrm{d}^2 v}{\mathrm{d}t^2}_{\text{pred}}, \frac{\mathrm{d}^2 v}{\mathrm{d}t^2}_{\text{true}} \right) \tag{20}$$

W celu wymuszenia przynależności wartości parametrów do określonych przedziałów zastosowano następującą funkcję kosztu:

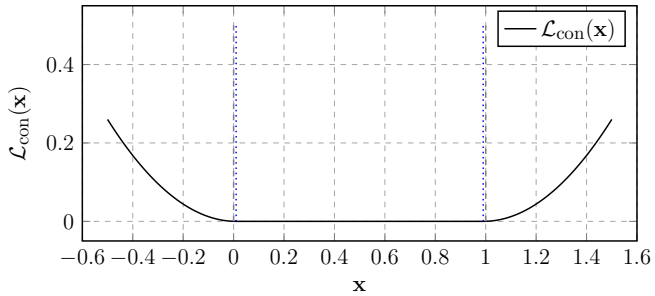
$$\mathcal{L}_{\text{con}}(\mathbf{x}) = \mathbb{E}\left[\left(\sigma\left(-\frac{\mathbf{x} - (a+\delta)}{\delta/4} \right) \cdot ((a+\delta) - \mathbf{x}) + \sigma\left(\frac{\mathbf{x} - (b-\delta)}{\delta/4} \right) \cdot (\mathbf{x} - (b-\delta)) \right)^2 \right] \tag{21}$$

gdzie: $\sigma(z) = \frac{1}{1+e^{-z}}$ oznacza funkcję sigmoidalną, a i b oznaczają odpowiednio dolne i górne ograniczenie zakresu dopuszczalnych wartości parametru, δ określa margines przejścia (szerokość strefy przejściowej między dopuszczalnym a karanym zakresem), λ_{con} to współczynnik wagowy odpowiadający za siłę działania ograniczenia, $\mathbb{E}$ oznacza wartość oczekiwaną (średnią po zbiorze parametrów lub trajektorii).

Podczas trenowania sieci neuronowej zastosowano następujące ograniczenia przedziałowe, aby zapewnić fizycznie sensowne i numerycznie stabilne wartości parametrów:

- $R_{\text{ON}} \in (0, 100)$ kΩ oraz $R_{\text{OFF}} \in (0, 100)$ kΩ – w celu ograniczenia możliwych stanów rezystancyjnych,
- $x_0 \in (0, 1)$ – aby ograniczyć początkowy stan wewnętrzny do fizycznie dopuszczalnych wartości,
- $x(t) \in (0, 1)$ – w celu utrzymania zmiennej stanu wewnętrznego w znormalizowanej dziedzinie w trakcie całej symulacji.

Przykładowe działanie funkcji ograniczającej $\mathcal{L}_{\text{con}}(\mathbf{x})$ przedstawiono na Rys. 4, dla wartości: $a = 0$, $b = 1$, $\delta = 0.01$ w zakresie $\mathbf{x} \in (-0.5, 1.5)$.



Rysunek 4: Przykładowe działanie funkcji ograniczającej $\mathcal{L}_{\text{con}}(\mathbf{x})$ dla wartości: $a = 0, b = 1, \delta = 0.01$ w zakresie $\mathbf{x} \in (-0.5, 1.5)$.

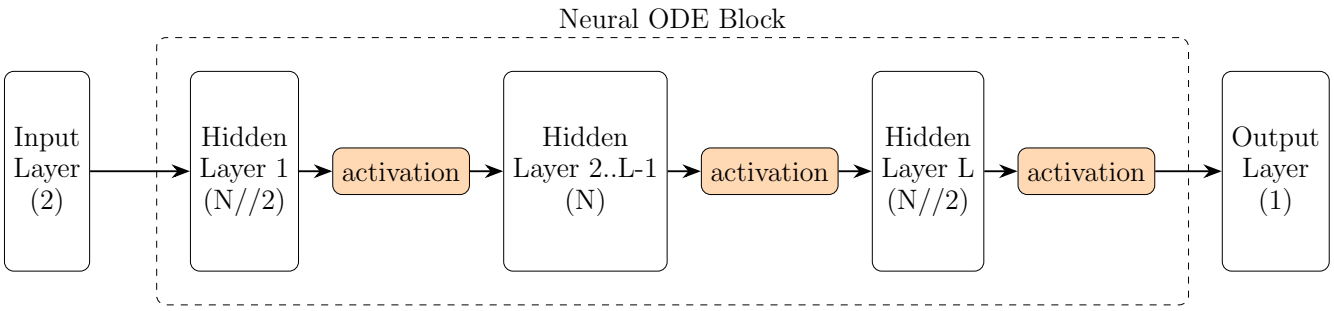
2.5 Detale implementacji

Proces tworzenia oraz trenowania modelu sieci neuronowej został zrealizowany przy wykorzystaniu frameworka `PyTorch`, zaimplementowanego w języku `Python`.

W celu umożliwienia integracji równań różniczkowych w trakcie procesu uczenia, zastosowano bibliotekę `torchdiffeq`. Biblioteka ta udostępnia różniczkowalne rozwiązywacze równań różniczkowych zwyczajnych (ODE), co pozwala na efektywne wyliczanie gradientów względem trajektorii rozwiązania poprzez zastosowanie metody czułości sprzężonej (ang. adjoint sensitivity method). Dzięki temu możliwe jest przeprowadzenie end-to-end treningu modeli opisywanych dynamiką wyrażoną za pomocą równań różniczkowych. W szczególności wykorzystano solver `dopri5`. Architektura sieci neuronowej została przedstawiona na Rys. 5. Redukcja wymiaru w pierwszej i ostatniej warstwie ukrytej wynika z kilku istotnych aspektów:

- Zmniejszenie liczby parametrów sieci, co przyczynia się do szybszego uczenia oraz redukcji ryzyka przeuczenia (overfitting).
- Efektywna ekstrakcja istotnych cech z danych wejściowych, co sprzyja poprawie zdolności modelu do generalizacji.
- Stabilizacja procesu uczenia poprzez ograniczenie liczby parametrów, co może prowadzić do bardziej stabilnej i szybszej konwergencji.
- Wprowadzenie mechanizmu regularyzacji za pomocą konstrukcji bottleneck, sprzyjającej wypracowaniu bardziej skoncentrowanej reprezentacji wewnętrznej, korzystnej dla modelowania dynamiki systemów.
- Redukcja ryzyka zaniku lub eksplozji gradientu, co ma szczególne znaczenie przy modelowaniu długich sekwencji czasowych.

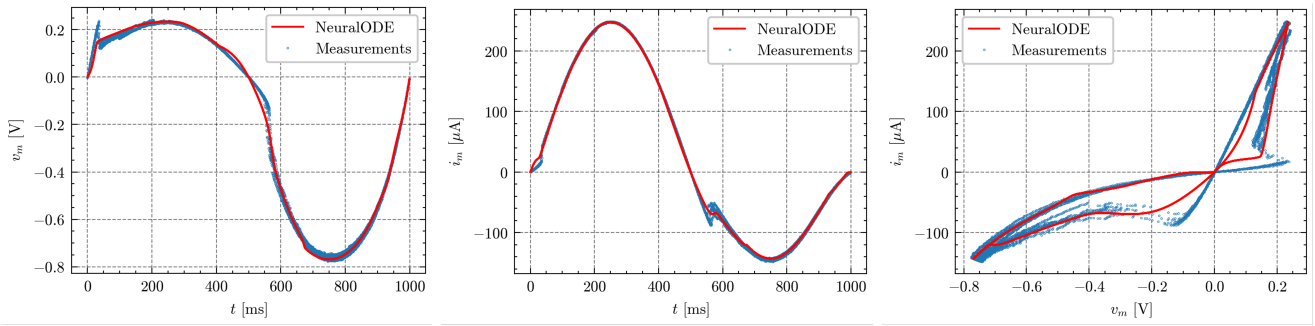
Do optymalizacji wag sieci wykorzystano algorytm `Adam`, który jest jedną z najpopularniejszych metod optymalizacji w kontekście głębokiego uczenia. Charakteryzuje się on adaptacyjnym dostosowaniem współczynników uczenia dla każdego parametru, co pozwala na efektywniejsze i szybsze zbieżności w porównaniu do tradycyjnych metod, takich jak `SGD` (Stochastic Gradient Descent). W celu adaptacyjnej kontroli współczynnika uczenia podczas treningu zastosowano metodę adaptacyjnego harmonogramowania współczynnika uczenia, znaną jako `ReduceLROnPlateau`. Ta oparta na wydajności strategia dynamicznie dostosowuje wartość współczynnika uczenia w odpowiedzi na zmiany obserwowanego wskaźnika walidacyjnego (dokładności), co przyczynia się do zwiększenia efektywności zbieżności oraz poprawy zdolności uogólniania modelu.



Rysunek 5: Schemat działania solvera ODE w trakcie procesu uczenia.

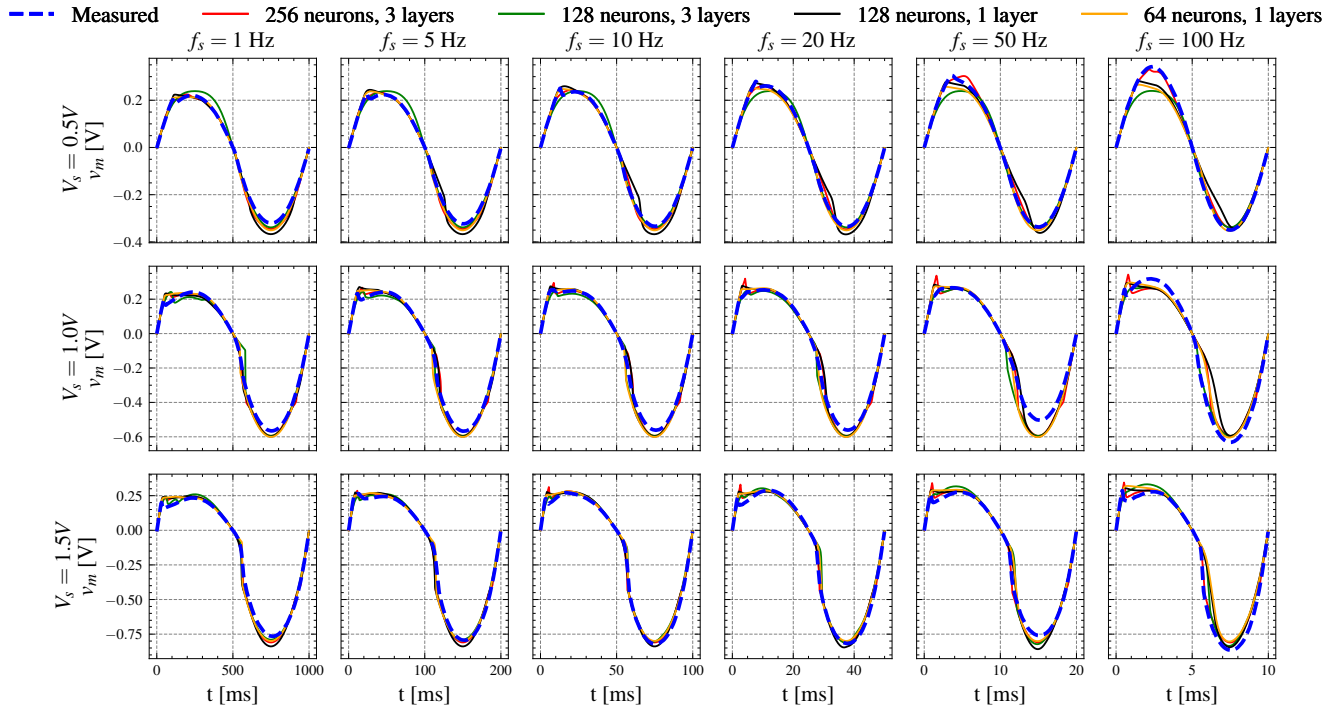
3 Wyniki

Przykładowe wyniki symulacji przedstawiono na Rys. 6 w formie wykresów porównawczych dla memrystora domieszkowanego wolframem, poddanego sinusoidalnemu pobudzeniu napięciowemu o amplitudzie 1V oraz częstotliwości 1Hz. Analiza obejmuje trzy podstawowe charakterystyki dynamiczne przebieg prądu memrystora w czasie, przebieg napięcia przyłożonego do memrystora, pętle histerezy napięcie–prąd ($v_m - i_m$), ilustrujące nieliniową zależność między tymi wielkościami oraz właściwości pamięciowe elementu.

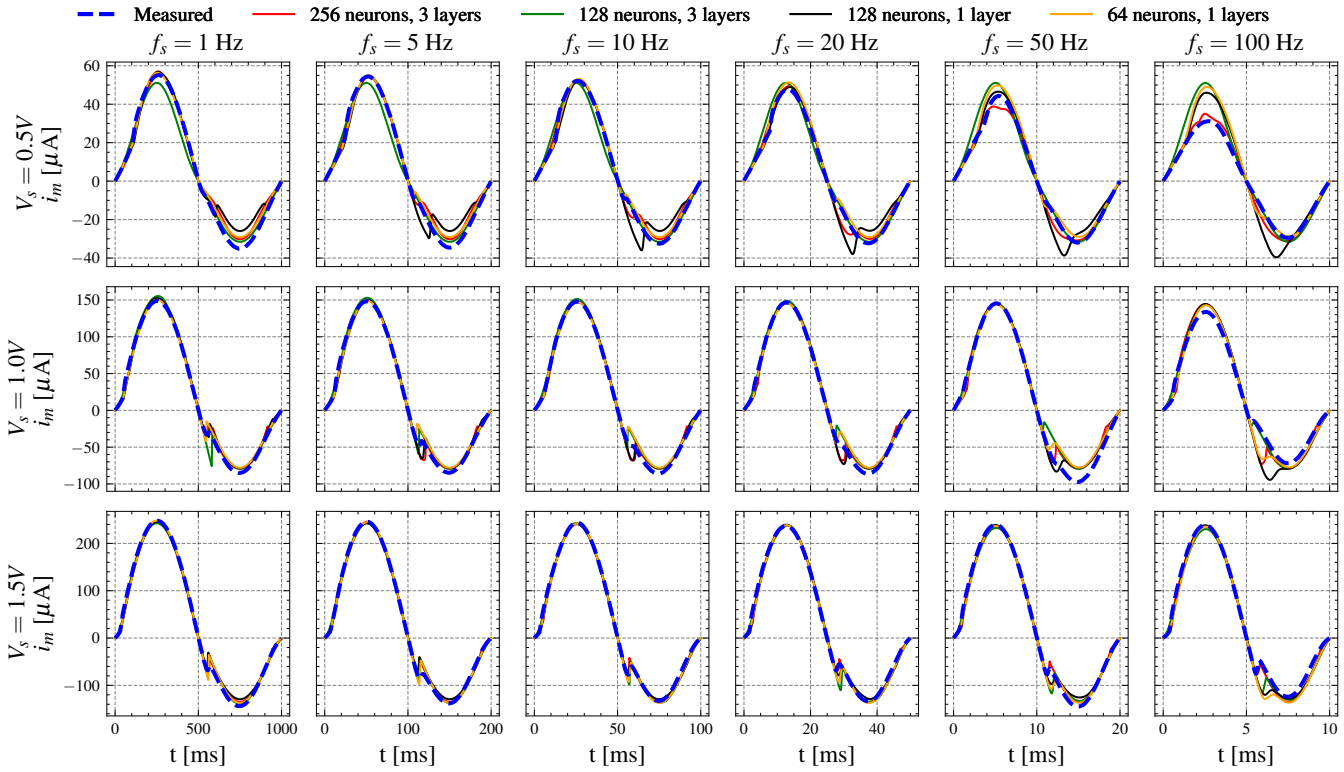


(a) (b) (c)

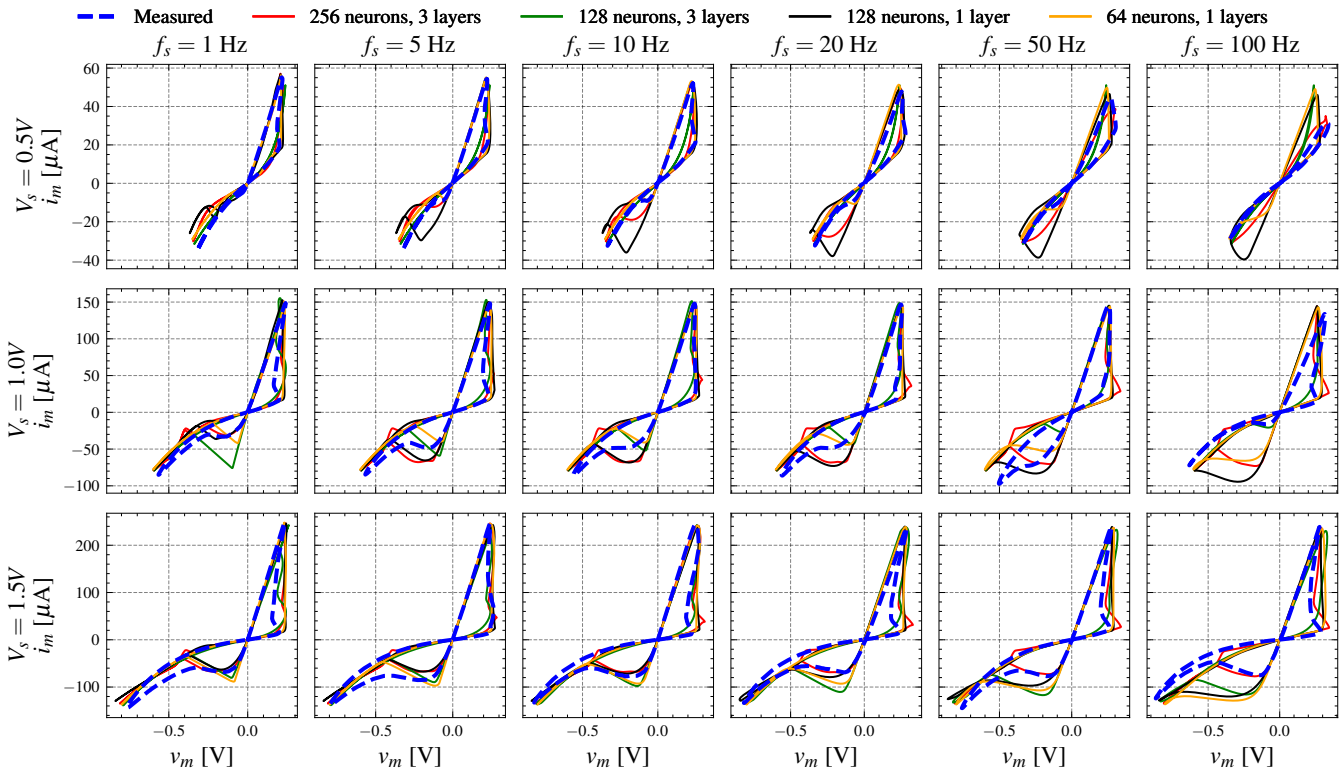
Rysunek 6: Przykładowe wyniki symulacji memrystora domieszkowanego wolframem, poddanego sinusoidalnemu pobudzeniu napięciowemu o amplitudzie 1V oraz częstotliwości 1Hz. (a) Przebieg napięcia przyłożonego do memrystora, (b) Przebieg prądu memrystora, (c) Pętla histerezy napięcie–prąd ($v_m - i_m$).



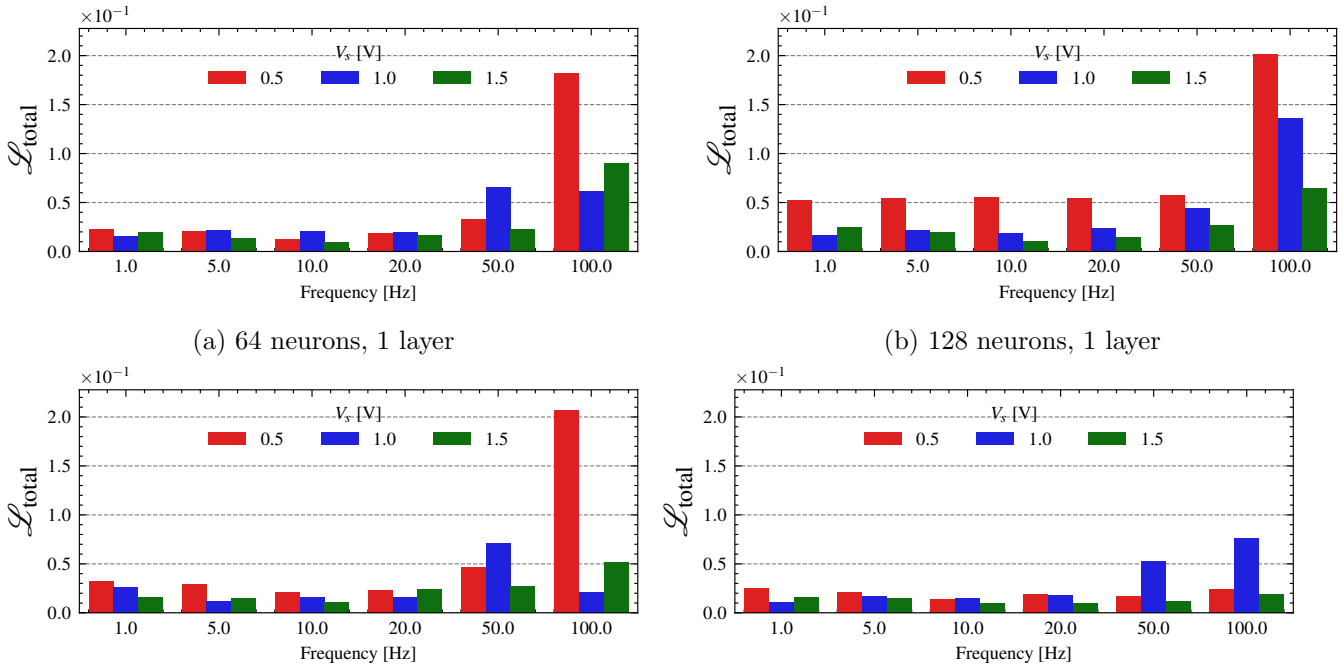
Rysunek 7: Porównanie uśrednionych do jednego okresu , przebiegów napięcia na memrystorze v_m dla różnych architektur sieci neuronowej.



Rysunek 8: Porównanie uśrednionych do jednego okresu, przebiegów prądu memrystora i_m dla różnych architektur sieci neuronowej.



Rysunek 9: Porównanie uśrednionych do jednego okresu, pętli histerezy $v_m - i_m$ memrystora dla różnych architektur sieci neuronowej.



(a) 64 neurons, 1 layer

(b) 128 neurons, 1 layer

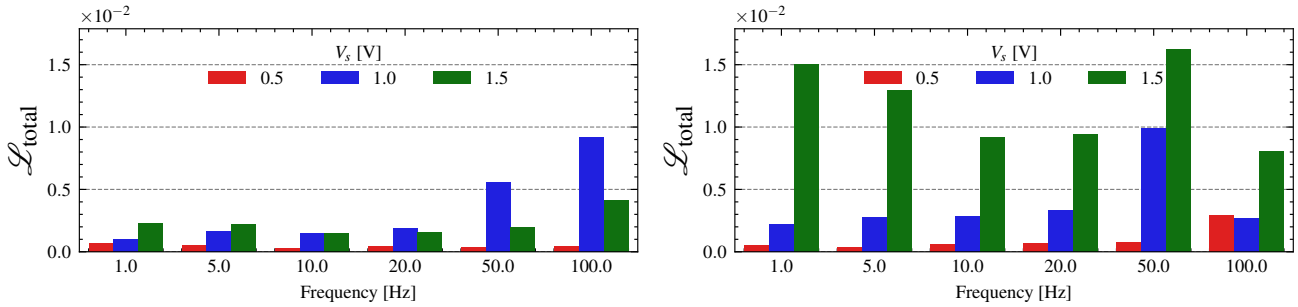
(c) 128 neurons, 3 layers

(d) 256 neurons, 3 layers

Rysunek 10: Porównanie funkcji straty dla różnych architektur sieci neuronowej.

3.1 Porównanie z modelem MMS

Porównano wyniki symulacji uzyskane z modelu sieci neuronowej z wynikami uzyskanymi z modelu Mean Metastable Switch (MMS). Analiza obejmuje przebiegi napięcia i prądu memrystora, a także pętle histerezy. Przykładowe wyniki przedstawiono na Rys. 11. Średnia warotść funkcji straty dla modelu sieci neuronowej wynosiła 0.002052, natomiast dla modelu MMS wynosiła 0.005589. Wartości te wskazują na lepszą zgodność modelu sieci neuronowej z danymi eksperymentalnymi w porównaniu do modelu MMS.

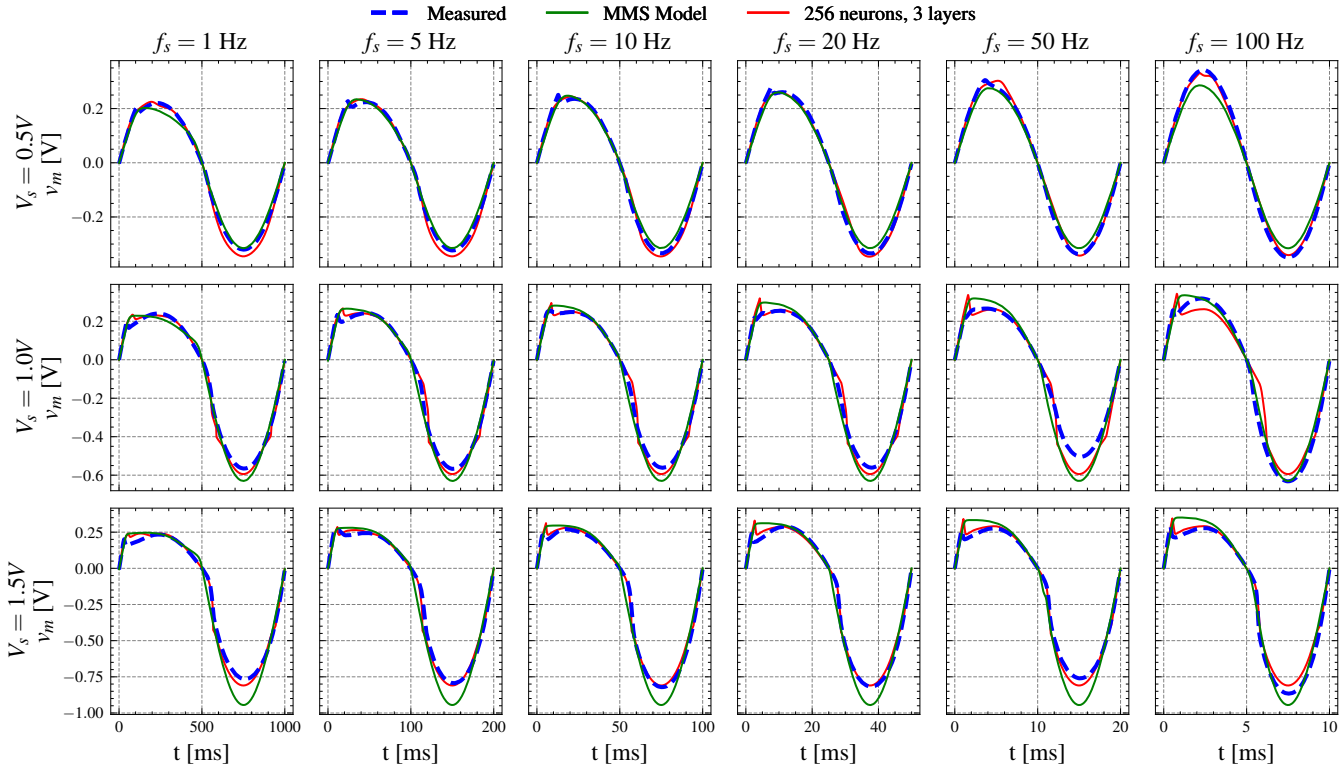


(a) 256 neurons, 3 layers

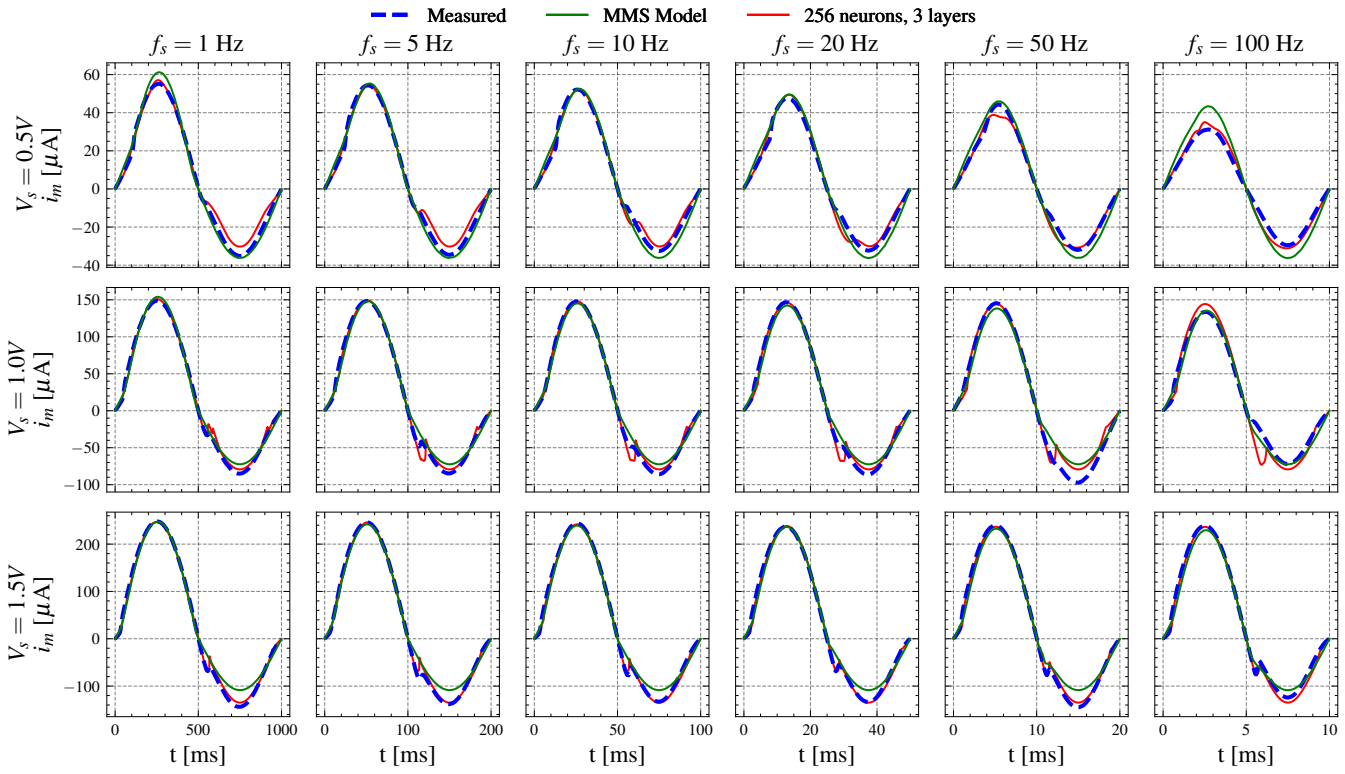
(b) MMS Model

Rysunek 11: Porównanie funkcji straty dla sieci neuronowej o 3 warstwach ukrytych oraz 256 neuronach (a), oraz modelu MMS (b).

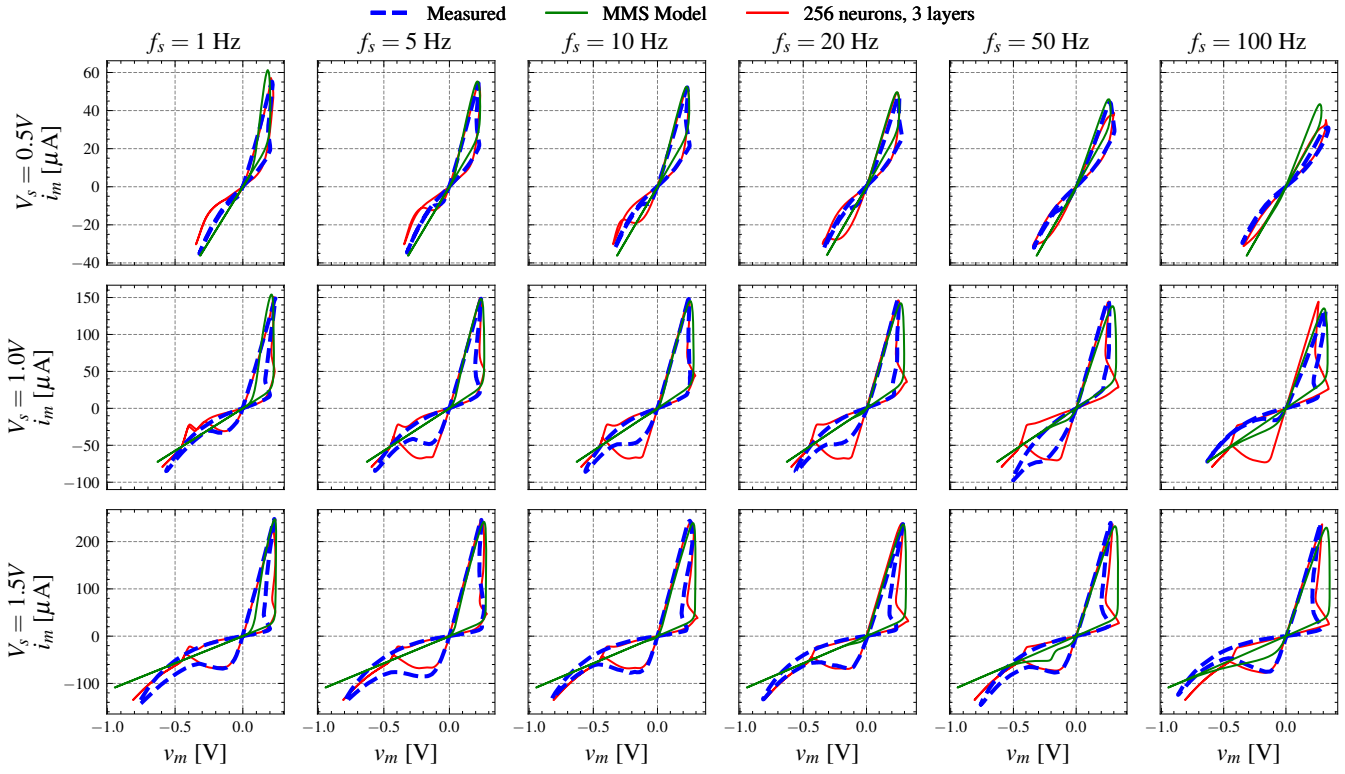
Przykładowe przebiegi napięcia i prądu memrystora, a także pętle histerezy dla modelu sieci neuronowej oraz modelu MMS przedstawiono na Rys. 12, 13, 14.



Rysunek 12: Porównanie uśrednionych do jednego okresu, przebiegów napięcia na memrystorze v_m dla sieci neuronowej oraz modelu MMS.



Rysunek 13: Porównanie uśrednionych do jednego okresu, przebiegów prądu memrystora i_m dla sieci neuronowej oraz modelu MMS.



Rysunek 14: Porównanie uśrednionych do jednego okresu, pętli histerezy $v_m - i_m$ memrystora dla sieci neuronowej oraz modelu MMS.

3.2 Analiza wpływu hiperparametrów sieci neuronowej

TODO: Analiza zostanie przeprowadzona po zakończeniu procesu trenowania sieci neuronowej.

Bibliografia

[1] Kristy A. Campbell. "Self-directed channel memristor for high temperature operation". In: *Microelectronics Journal* 59 (Jan. 2017), pp. 10–14. ISSN: 1879-2391. DOI: 10.1016/j.mejo.2016.11.006. URL: http://dx.doi.org/10.1016/j.mejo.2016.11.006.

[2] Ricky T. Q. Chen, Brandon Amos, and Maximilian Nickel. "Learning Neural Event Functions for Ordinary Differential Equations". In: *International Conference on Learning Representations* (2021).

[3] Ricky T. Q. Chen et al. "Neural Ordinary Differential Equations". In: 2018.

[4] Bartłomiej Garda and Karol Bednarz. "Comprehensive Study of SDC Memristors for Resistive RAM Applications". In: *Energies* 17.2 (Jan. 2024), p. 467. ISSN: 1996-1073. DOI: 10.3390/en17020467. URL: http://dx.doi.org/10.3390/en17020467.

[5] Younghyun Lee, Kyeongmin Kim, and Jonghwan Lee. "A Compact Memristor Model Based on Physics-Informed Neural Networks". In: *Micromachines* 15.2 (2024). ISSN: 2072-666X. DOI: 10.3390/mi15020253. URL: https://www.mdpi.com/2072-666X/15/2/253.

[6] Timothy W. Molter and M. Alexander Nugent. "The Generalized Metastable Switch Memristor Model". In: *CNNA 2016; 15th International Workshop on Cellular Nanoscale Networks and their Applications*. 2016, pp. 1–2.

[7] Valerii Ostrovskii et al. "Structural and Parametric Identification of Knowm Memristors". In: *Nanomaterials* 12.1 (Dec. 2021), p. 63. ISSN: 2079-4991. DOI: 10.3390/nano12010063. URL: http://dx.doi.org/10.3390/nano12010063.

[8] Dmitri B. Strukov et al. "The missing memristor found". In: *Nature* 453.7191 (May 2008), pp. 80–83. ISSN: 1476-4687. DOI: 10.1038/nature06932. URL: http://dx.doi.org/10.1038/nature06932.