

bokie uczenie maszyngreen6 a^{cz}onagreen7

Wykrywanie wdechu i wydechu na podstawie dźwięku z mikrofonu

Dominik Lau, Mateusz Kowalczyk, Michał Tarnacki

19 maja 2023

Spis treści

1 Wstęp	3
2 Teoria	3
2.1 Dyskretna transformata Fouriera (DFT)	3
2.2 Bramka szumów	4
2.3 Filtr Savitzky'ego-Golaya	5
2.4 Ocena modeli	6
2.5 Standaryzacja	7
2.6 SVM	7
2.7 Sieć neuronowa	7
3 Narzędzia i inne	7
3.1 Nagrywanie danych oddechowych	7
3.2 Oddychanie na żywo	8
3.3 Przeliczanie częstotliwości z DFT	9
4 Początkowo przyjęty model oddechu	9
5 Pierwotne podejście – średnia częstotliwość w czasie	10
5.1 Metoda	10
5.2 Problemy	11
6 Wykorzystanie spektrogramu	12
6.1 Założenie	12
6.2 Metoda	12
6.3 Kluczowe częstotliwości	12
6.4 Poprzedni stan	14
6.5 Problemy	15

7 Sieć neuronowa	16
7.1 Założenie	16
7.2 Metoda	16
7.3 Wykorzystywane próbki	17
7.4 Problemy	17
7.5 Rozwiązańe	17
8 Przejście na inny model oddychania	17
8.1 Wydech nosem	17
8.2 Kluczowe częstotliwości	18
8.3 Usuwanie ciszy	20
8.4 Sieć neuronowa	21
9 Maszyna stanów do zwiększenia inercji	21
9.1 Cel	21
9.2 Schemat	22
9.3 Rezultat	23
10 Wnioski	23
11 Źródła	23

1 Wstęp

Celem projektu było określanie chwil na nagraniu, w których osoba bierze wdech i wydech. Dokonano oceny jakościowej za pomocą detekcji oddechu na żywo jak i ilościowej (przy wykorzystaniu dalej wymienionej miar).

2 Teoria

2.1 Dyskretna transformata Fouriera (DFT)

W celu przejścia z dziedziny natężenia od czasu do dziedziny częstotliwości stosujemy dyskretną transformatę Fouriera. DFT przekształca ciąg skończonych próbek sygnału $a_0, a_1, a_2, \dots, a_{N-1}$ w ciąg $A_1, \dots, A_{N-1} \in \mathbb{C}$

$$A_k = \sum_{n=0}^{N-1} a_n e^{-\frac{kn\pi}{N}}, 0 \leq k \leq N-1$$

W naszych zastosowaniach będziemy brali pod uwagę tylko część rzeczywistą tego wielomianu

$$R_k = \operatorname{Re}(A_k), 0 \leq k \leq N-1$$

Skorzystamy z odmiany dyskretnej transformaty Fouriera - STFT (krótkokocza-sowa transformata Fouriera). Jest to zasadniczo transformacja Fouriera jednak

wykonywana w mniejszych oknach czasowych. Dzięki temu otrzymujemy informację o zmianie widma w czasie, czyli jednocześnie posiadamy informację o właściwościach zarówno w dziedzinie czasu, jak i częstotliwości. Najistotniejszym parametrem tej metody jest rozmiar okna. Ponieważ iloczyn jego szerokości w dziedzinie czasu i szerokości w dziedzinie częstotliwości jest stały dla danego okna, poprawiając rozdzielcość czasu, pogarszamy rozdzielcość częstotliwości. Naszą funkcję natężenia w czasie dla pewnego okresu $t \in [t_0, t_0 + B]$, gdzie B - rozmiar okna (przyjmujemy $B = 1024 \approx \frac{1}{44}s$), przedstawiamy zatem jako

$$I(t) = \sum_f I_f(t) \sin(2\pi f t)$$

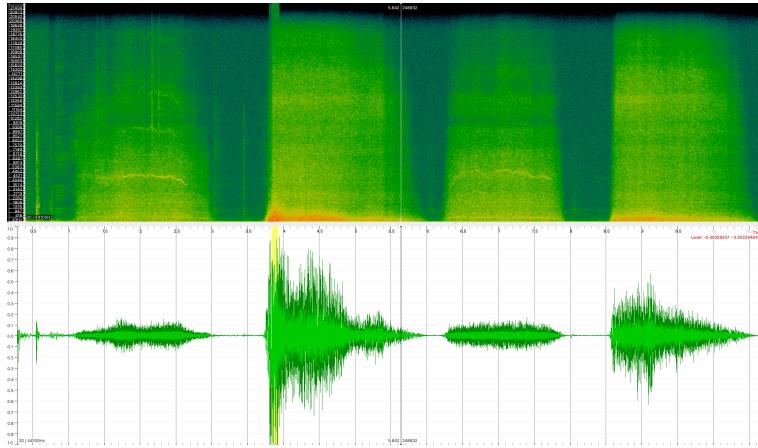
stąd dla danego przedziału w czasie jesteśmy w stanie stworzyć wektor natężeń

$$\mathbf{I} = [I_{f_1}, I_{f_2}, \dots, I_{f_n}]$$

W dalszych rozważaniach wykorzystujemy także średnią częstotliwość, którą liczymy ze wzoru

$$\bar{f} = \frac{\sum_f f I_f}{I}$$

Wykorzystujemy implementację transformaty z bibliotek *numpy* oraz *tensorflow*.



Rysunek 1: Przykład sygnału wdechu i wydechu nosem poddanego transformacji Fouriera dla okien czasowych – otrzymujemy nowy wymiar informacji, czyli nie tylko natężenie dźwięku, ale też natężenie poszczególnych częstotliwości w dźwięku. Obraz z programu Sonic Visualizer, który realizuje STFT.

2.2 Bramka szumów

Główna metoda odszumiana, z której korzystamy, to *Spectral Gating* (rodzaj bramki szumów). Pierw wyznaczany jest spektrogram sygnału za pomocą *DFT* i tworzony jest próg szumu dla każdej jego częstotliwości. Częstotliwości progowe

służą do stworzenia maski, której potem używamy do usunięcia niechcianych dźwięków. Następnie ze spektrogramu tworzymy z powrotem natężenie $I(t)$. W projekcie wykorzystujemy gotową implementację bramki szumów z biblioteki *noisereduce*¹. Zdefiniujmy

$$SNR = \frac{P_s}{P_n}$$

gdzie P_n – moc szumu, $P_s = P - P_n$ – moc sygnału użytecznego (moc sygnału pomniejszona o moc szumu). Porównanie przykładowych SNR dla algorytmów usuwania szumu²:

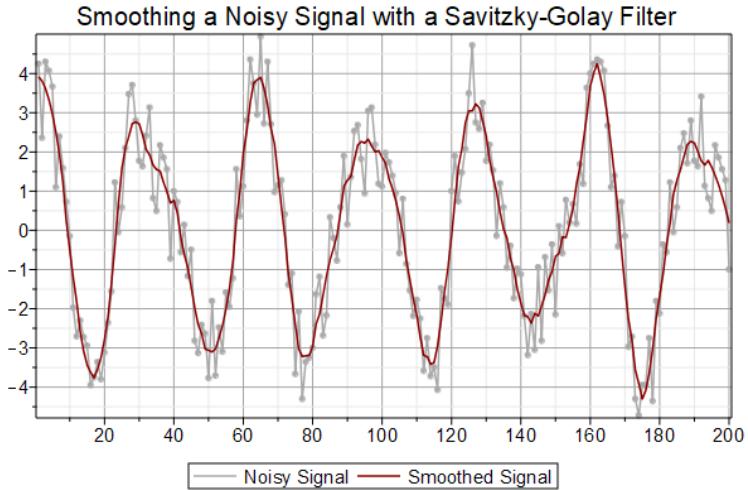
algorytm	SNR
<i>LMS</i>	2
<i>Kalman</i>	6
<i>Spectral Gating</i>	14

2.3 Filtr Savitzky'ego-Golaya

Filtru Savitzky'ego-Golaya używamy do wygładzenia spektrogramu po przeprowadzeniu DFT. Polega on na dopasowywaniu zbioru sąsiadujących punktów do wielomianu niskiego stopnia za pomocą metody najmniejszych kwadratów. Te fragmenty wielomianów są potem łączone w wygładzoną funkcję. Korzystamy z gotowej implementacji z biblioteki *scipy*. W przypadku sieci neuronowej korzystamy natomiast z warstw konwolucyjnych, nakładających odpowiednie filtry na obraz.

¹<https://pypi.org/project/noisereduce/>

²Kumar, E. and Surya, K. and Varma, K. and Akash, A. and Kurapati, Nithish Reddy (2023) *Noise Reduction in Audio File Using Spectral Gating and FFT by Python Modules*



Rysunek 2: Wizualizacja działania filtra Savitzky'ego-Golaya dla przykładowego zbioru próbek, <https://www.maplesoft.com/Applications/Detail.aspx?id=154593>

2.4 Ocena modeli

Zastosowaliśmy następujące miary oceny ilościowej modelu:

$$\begin{aligned} \text{accuracy} &= \frac{|TP_{wdech}| + |TN_{wdech}|}{|TP_{wdech}| + |FP_{wdech}| + |TN_{wdech}| + |FN_{wdech}|} \\ \text{precision}_{wdech} &= \frac{|TP_{wdech}|}{|TP_{wdech}| + |FP_{wdech}|} \\ \text{recall}_{wdech} &= \frac{|TP_{wdech}|}{|TP_{wdech}| + |FN_{wdech}|} \\ F_{wdech} &= \frac{2 \cdot \text{precision}_{wdech} \cdot \text{recall}_{wdech}}{\text{precision}_{wdech} + \text{recall}_{wdech}} \end{aligned}$$

gdzie TP_{wdech} – liczba próbek oznaczonych jako wdech, które sklasyfikowane zostały jako wdech, TN_{wdech} – liczba próbek nieoznaczonych jako wdech, które nie zostały sklasyfikowane jako wdech, FP_{wdech} – liczba próbek nieoznaczonych jako wdech, które zostały sklasyfikowane jako wdech, FN_{wdech} – liczba próbek oznaczonych jako wdech, które nie zostały sklasyfikowane jako wdech. Analogicznie definiujemy $\text{precision}_{wydech}$, recall_{wydech} , F_{wydech} .

2.5 Standaryzacja

Dane wejściowe do modelu zawsze standaryzujemy, wykonując operację daną wzorem:

$$\tilde{X} = \frac{X - EX}{\sigma_X}.$$

Stosujemy gotową implementację z biblioteki *scikit-learn* lub warstwę normalizacyjną w przypadku sieci neuronowej.

2.6 SVM

Pierwszym modelem, jakim dokonujemy klasyfikacji, jest SVM. Metoda ta polega na szukaniu optymalnych wag \mathbf{w}, b . Następnie będziemy klasyfikować według funkcji maszyny uczącej

$$f(x) = \text{sgn}(\mathbf{w}^T \mathbf{x} + b).$$

Jeśli f zwraca 1, traktujemy to jako jedną z binarnych decyzji (w naszym przypadku np. wdech) a jeśli zwraca -1 , to drugą (czyli np. wydech). \mathbf{x} jest wektorem cech. Znalezienie optymalnych wag będzie polegało na minimalizacji funkcji kosztu

$$J(\mathbf{w}) = \frac{1}{2} \|\mathbf{w}\|^2 + \frac{C}{N} \sum_x \max\{0, 1 - y(\mathbf{w}^T \mathbf{x} + b)\},$$

gdzie C jest parametrem uregulowania. Funkcję będziemy minimalizować metodą stochastycznego spadku po gradiencie. W projekcie użyliśmy własnej implementacji SVM-a.

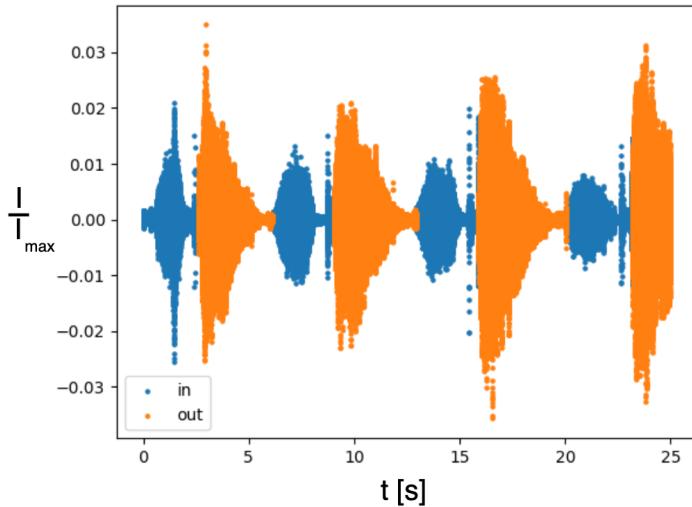
2.7 Sieć neuronowa

Drugim modelem, jakim dokonujemy klasyfikacji, jest sieć neuronowa. Ze względu na trudność w samodzielnym zaprojektowaniu dobrej sieci neuronowej postanowiliśmy użyć stworzonego przez firmę Google Tensorflowa wraz z interfejsem Keras.

3 Narzędzia i inne

3.1 Nagrywanie danych oddechowych

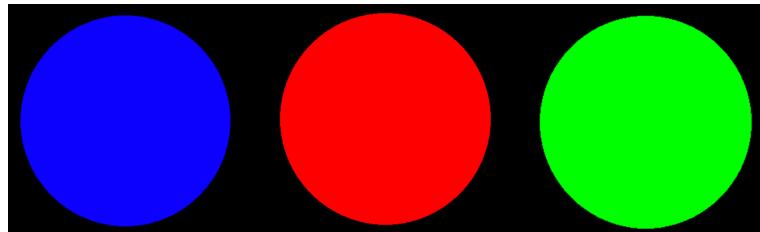
Żeby zapewnić dobre oznaczenie danych, etykietujemy je jeszcze w trakcie nagrywania dźwięku. Osoba nagrywana naciska przycisk, żeby zasygnalizować, że przestała brać wdech i zaczyna wydychać powietrze lub na odwrót. Momenty przejścia z wdechu na wydech i w drugą stronę zapisywane są w pliku .csv, a dźwięk w pliku .wav. Częstotliwość próbkowania ustalamy na 44,1 kHz.



Rysunek 3: Przykładowe nagrane dane, in-wdech, out-wydech. Jak widać, dane oznaczone na żywo są bardzo dokładne, być może nie bylibyśmy w stanie osiągnąć takiej dokładności oznaczając ręcznie (lub byłoby to bardzo żmudne).

3.2 Oddychanie na żywo

Testowanie modeli do predykcji na żywo odbywało się za pomocą programu wizualizującego aktualnie stwierdzany stan.



Rysunek 4: Jeżeli koło zwiększa się i jest niebieskie, model wykrywa wdech, jeśli zmniejsza się i jest czerwone, jest to wydech, natomiast kolor zielony oznacza, że natężenie dźwięku nie przekracza pewnego progu dobieranego eksperymentalnie w zależności od mikrofonu.

Dalej klasyfikację na żywo nazywamy też testem jakościowym.

3.3 Przeliczanie częstotliwości z DFT

W celu wyznaczenia sposobu zamiany częstotliwości otrzymanej za pomocą DFT na znaną jednostkę (Hz) wykonaliśmy kilkanaście nagrań z ustaloną (w Hz) dominującą częstotliwością i odczytaliśmy, której wartości częstotliwości otrzymanej za pomocą DFT odpowiada maksymalne natężenie. Wyniki przedstawia rysunek 5.



Rysunek 5: na osi poziomej ustalona w Hz dominująca częstotliwość z nagrania, na osi pionowej – wartość częstotliwości otrzymanej w wyniku DFT, której odpowiada maksymalne natężenie.

Po zaobserwowaniu w przybliżeniu liniowego charakteru tej relacji zastosowaliśmy metodę najmniejszych kwadratów do wyznaczenia najlepiej dopasowanej prostej. Otrzymaliśmy następujące wyniki:

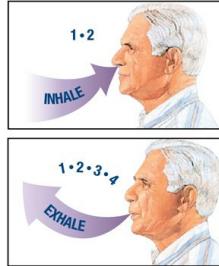
$$f_{Hz} \approx 43,0789 f_{DFT} - 2,8174,$$
$$f_{DFT} \approx 0,0232 f_{Hz} + 0,068,$$

gdzie f_{DFT} to wartość liczbową częstotliwości w jednostkach, w której otrzymuje się wynik DFT, natomiast f_{Hz} to wartość liczbową częstotliwości wyrażonej w Hz.

4 Początkowo przyjęty model oddechu

Na początku przyjmujemy model „sportowego” oddychania, czyli wdech nosem i wydech ustami. Uproszczenie to polega na tym, że wydawane dźwięki są dosyć

różne, dopiero potem sprawdzamy jak nasze podejście będzie się sprawować przy innych metodach oddychania.



Rysunek 6: Ilustracja przyjętego modelu, źródło

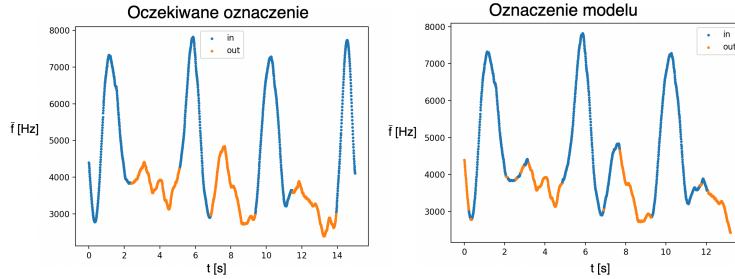
5 Pierwotne podejście – średnia częstotliwość w czasie

5.1 Metoda

Pierwotnie przyjętym założeniem było, że podczas wdechu średnia częstotliwość dźwięku jest wyższa niż gdy osoba wydycha. Z pliku w formacie .wav pobieramy natężenie, które następnie **odszumiamy** za pomocą *noisereduce* (przy testach jakościowych stosujemy bramkę szumów dla 3 ostatnich sekund). **Dzielimy próbki na bloki, dla których tworzymy spektrogramy** i obliczamy średnią ważoną częstotliwość. Cechami, na podstawie których dokonywana jest predykcja, są

$$\mathbf{x}(t_n) = [\bar{f}(t_1), \dots, \bar{f}(t_n)],$$

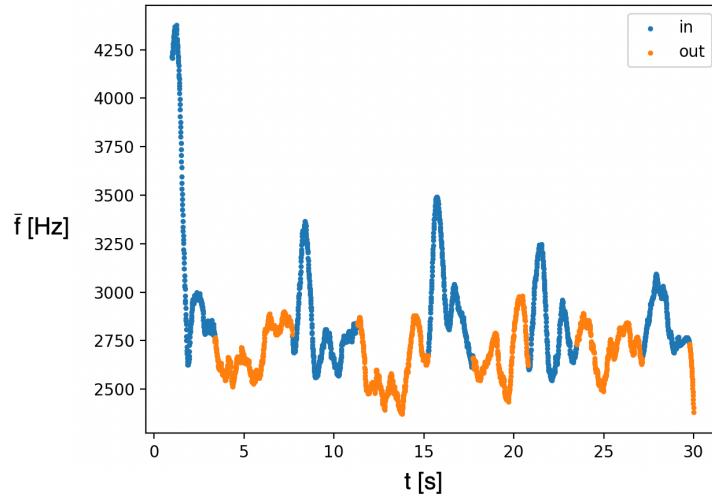
czyli średnie częstotliwości z kilku przeszłych chwil. Dla danych testowych spełniających powyższe założenie (predykcja na podstawie ostatnich 100 i modelu wytrenowanego SVM-em) dawało nam to dokładność $\sim 60\%$. (Inne miary nie zostały zmierzone ze względu na problemy, o których mowa w kolejnym podpunkcie).



Rysunek 7: Przykładowe działanie modelu dla danych spełniających założenie: wyższy wdech, niższy wydech. W pewnych momentach podobnie wyglądające fragmenty krzywej powinny być wdechem, w innych wydechem, mamy więc możliwy underfitting.

5.2 Problemy

Podejście to jednak mocno ogranicza nasze dalsze pole do rozwoju. Zmniejszenie całego spektrogramu do pojedynczej wartości częstotliwości daje nam mniej informacji, przez co być może ograniczylibyśmy się tylko do przyjętego przez nas modelu (tj. wdech – nos, wydech – usta) – w innych modelach różnica średnich częstotliwości między wdechem a wydechem może nie być taka wyraźna. Ponadto, dla niektórych osób zaobserwowałyśmy, że zależność między wdechem a wydechem jest niekoniecznie tak prosta jak założyliśmy. Obserwowana metoda była również bardzo niestabilna jeżeli chodzi o testy jakościowe (klasyfikację na żywo).

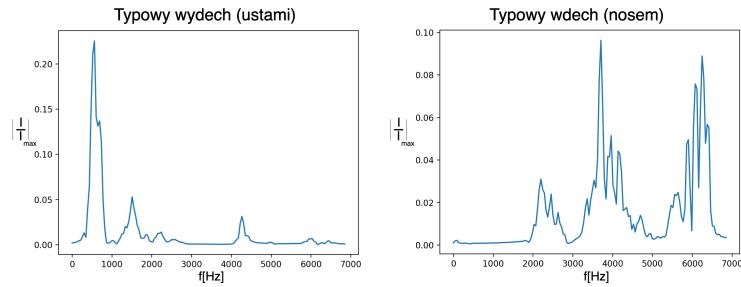


Rysunek 8: Nieoczywista zależność między wdechem a wydechem.

6 Wykorzystanie spektrogramu

6.1 Założenie

Kolejnym przyjętym przez nas podejściem było podanie całego spektrogramu (a przynajmniej jego części) jako dane wejściowe do SVM-a. Metoda pochodzi od przypuszczenia, że człowiek rozpoznaje i rozróżnia wdech/wydech na podstawie barwy dźwięku.



Rysunek 9: Istotnym jest fakt, że w przyjętym modelu oddychania wydech jest pojedynczym pikiem $\in (0, 1)$ kHz a wdech częstotliwościami z natężeniem o mniejszym odchyleniu $\in (2, 7)$ kHz. Zbliżoną zależność zaobserwowano dla kilku badanych osób.

6.2 Metoda

Podobnie jak ostatnio pierw odszumiamy sygnał z pliku za pomocą *noisereduce*. **Dzielimy dane na bloki rozmiaru 1024 próbek, tworzymy dla każdego bloku spektrogram** ale tym razem nie liczymy średniej tylko zostawiamy całą taką klatkę. Wektor cech ma więc postać

$$\mathbf{x} = [I_{f_1}, I_{f_2}, \dots, I_{f_n}]$$

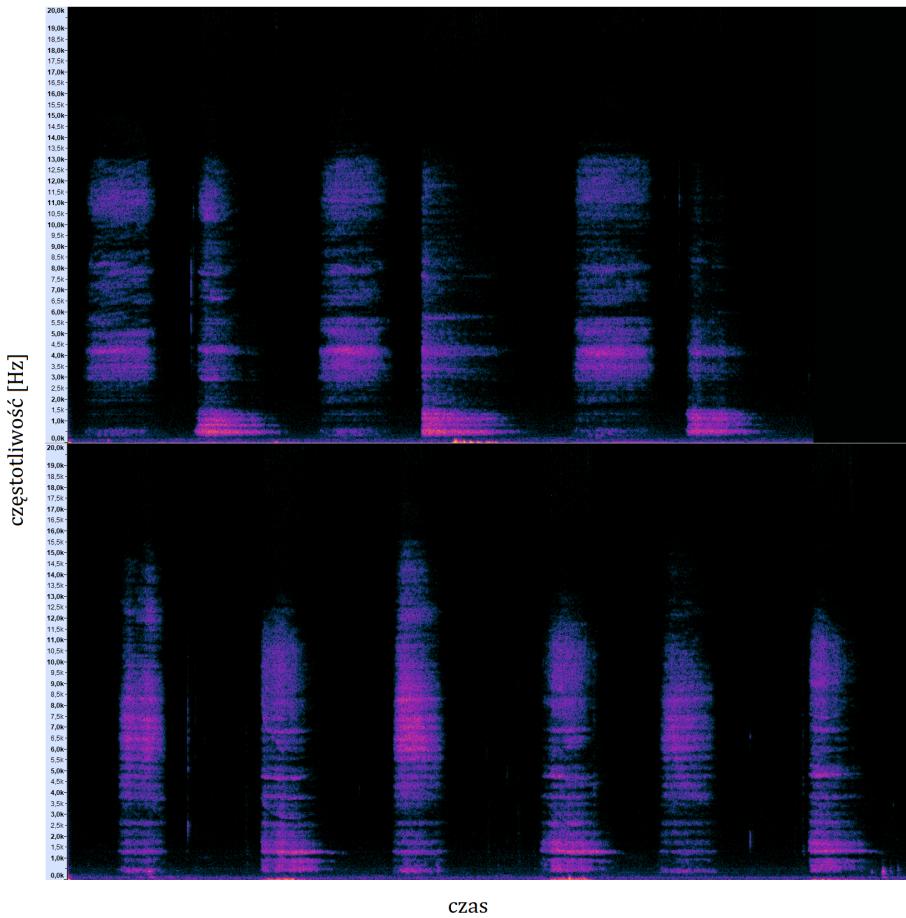
6.3 Kluczowe częstotliwości

Postanowiliśmy dokładniej przeanalizować, które częstotliwości są kluczowe dla poprawnej oceny fazy oddechu. W oparciu o obserwację spektrogramów wybraliśmy kilka możliwych zakresów, w których różnice między wdechem i wydechem są dobrze widoczne:

- 0 – 6 900 Hz,
- 0 – 1 300 Hz i 3 400 – 22 000 Hz,
- 0 – 14 000 Hz,
- 0 – 1 300 Hz i 3 400 – 14 000 Hz,
- 0 – 8 500 Hz i 10 000 – 14 000 Hz,
- 0 – 16 000 Hz,

- 0–1 300 Hz i 3 400–16 000 Hz,
- 0–1 300 Hz, 3 400–8 500 Hz i 10 000 Hz–16 000 Hz,
- 0–8 500 Hz i 10 000–16 000 Hz.

Spektrogram dźwięku oddechu

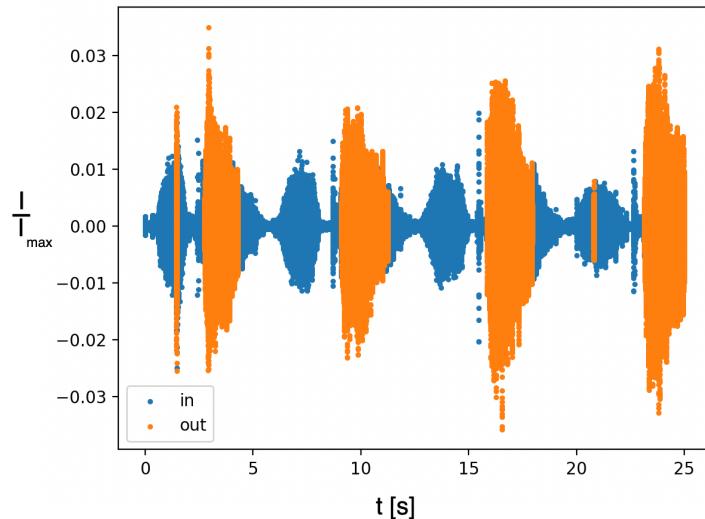


Rysunek 10: Spektrogramy dwóch nagrań oddechu z wdechem nosem i wydechem ustami uzyskane przy użyciu programu Audacity.

Dla każdego wyboru przeprowadziliśmy testy. W przypadku ostatniego uzyskaliśmy najlepsze wartości miar i dobre działanie przy przewidywaniu na żywo. Taki wybór częstotliwości przyjęliśmy zatem dla modelu rozpoznajającego wdech nosem i wydech ustami. **Używamy zatem częstotliwości z przedziału od 0 do 8 500 Hz i od 10 000 do 16 000 Hz.** Stosując to podejście dla danych testowych, otrzymaliśmy następujące wyniki:

miara	wartość (%)
$accuracy$	75
$precision_{wdech}$	68
$precision_{wydech}$	92
$recall_{wdech}$	88
$recall_{wydech}$	66
F_{wdech}	74
F_{wydech}	74

W testach jakościowych jednak zaobserwować było można „przeskakiwanie” z jednej wartości na drugą i z powrotem np. w połowie brania wdechu, wydychania.



Rysunek 11: Dane z rysunku 1. oznaczone, jak widać, występują przeskoki z jednej wartości na drugą.

6.4 Poprzedni stan

W celu pozbycia się „przeskakiwania” (zwiększenia inercji) i ogólnej poprawy działania dane z poprzedniego podpunktu rozszerzyliśmy o informację o **wczesniej przewidzianym stanie**, czyli wektor cech ma teraz postać

$$\mathbf{x} = [I_{f_1}, I_{f_2}, \dots, I_{f_{160}}, s]$$

gdzie $s = 1$ jeśli poprzednio stwierdzono wdech, $s = -1$ jeśli poprzednio stwierdzono wydech. Dodatkowo przeprowadziliśmy jeszcze **wygiadzenie spektrogramu filtrem Savitzky'ego-Golaya**. Następnie przetestowaliśmy program

w sposób uproszczony – jako wartość s stanu poprzedniego stosowaliśmy tę odczytaną z danych testowych, czyli zakładaliśmy za każdym razem, że nasz model przewidzi dobrze poprzedni stan. Otrzymaliśmy następujące wyniki:

Tabela 1: Wartości miar uzyskane podczas uproszczonych testów modelu z **wydechem ustami** przy wyborze częstotliwości 0–8 500 Hz i 10 000–16 000.

miara	wartość (%)
$accuracy$	99
$precision_{wdech}$	99
$precision_{wydech}$	99
$recall_{wdech}$	99
$recall_{wydech}$	99
F_{wdech}	99
F_{wydech}	99

Przypadek ten nie jest realny, ponieważ model nie będzie znał poprawnej poprzedniej klasyfikacji, lecz zakładał, że to co sam stwierdził jest poprawne. Odczytując poprzedni stan nie z danych testowych, a z poprzedniej klasyfikacji modelu, otrzymujemy gorsze, choć bardziej miarodajne wyniki:

Tabela 2: Wartości miar uzyskane podczas testów modelu z **wydechem ustami** przy wyborze częstotliwości 0–8 500 Hz i 10 000–16 000.

miara	wartość (%)
$accuracy$	87
$precision_{wdech}$	88
$precision_{wydech}$	87
$recall_{wdech}$	78
$recall_{wydech}$	65
F_{wdech}	83
F_{wydech}	90

Wniosek z tego jest taki, że model z wysokim prawdopodobieństwem przewidzi dobrze kolejny stan, jeżeli dobrze przewidział poprzedni. Warto dodać, że rzeczywiście metoda z poprzednim stanem pozwoliła nam ograniczyć „przeskakiwanie” w testach jakościowych, jednak całkowicie go nie wyeliminowała.

6.5 Problemy

W poprzednim podpunkcie wyszedł główny mankament tej metody – konieczność założenia, że poprzednio przewidziany stan jest dobry, co nie zawsze jest

prawdą. W testach jakościowych zaobserwować można było „wariowanie” modelu, czyli sekwencję źle przewidzianych stanów.

7 Sieć neuronowa

7.1 Założenie

Do przetestowania działania sieci neuronowej przyjęliśmy nieco odmienne podejście. Jako dane wejściowe przyjmujemy (podobnie jak w przypadku SVM) spektrogram, jednak w tym przypadku stworzony przy użyciu STFT (zarówno podczas uczenia jak i predykcji). Dzięki temu model bierze pod uwagę także zmianę widma w czasie.

7.2 Metoda

Stosując poszczególne warstwy, nasz spektrogram jest odpowiednio przetwarzany, tak aby uwidoczyć istotne cechy. Wykorzystywane przez nas warstwy (wraz z parametrami) to:

1. **Input** - Punkt wejścia do sieci neuronowej
2. **Resizing** - Zmiana rozmiaru obrazu (32x32)
3. **Normalization** - Normalizacja cech
4. **Conv2D** - Nałożenie odpowiedniego filtra o rozmiarze $n \times m$ (lub $n \times n$) na obraz tak aby uwydątnić jego cechy (32, 3, 'relu'). Wykorzystuje funkcję aktywacji.
5. **Conv2D** - Tak jak wyżej (64, 3, 'relu')
6. **MaxPooling2D** - Downsampling czyli wybranie maksymalnej wartości spośród okna o zadanych wymiarach (2x2)
7. **Dropout** - Ustawienie losowych wejść na 0 z częstotliwością n , przeskakowanie pozostałych cech przez $\frac{1}{1-n}$, pozwala na zredukowanie overfittingu (0.25)
8. **Flatten** - Spłaszczenie wejścia do 1 wymiaru
9. **Dense** - Łączy wszystkie neurony z warstwy poprzedniej ze wszystkimi z warstwy kolejnej, identyfikuje do jakiej klasy przynależy obiekt wejściowy. Wykorzystuje funkcję aktywacji. (128, 'relu')
10. **Dropout** - Tak jak wyżej (0.5)
11. **Dense** - Tak jak wyżej (64, 'relu')
12. **Dropout** - Tak jak wyżej (0.5)

13. **Dense** - Tak jak wyżej (32, 'softmax')
14. **Dropout** - Tak jak wyżej (0.5)
15. **Dense** - Tak jak wyżej (2)

Warstwy Dense i Dropout zostały dobrane doświadczyliście. Ze względu na ich dużą liczbę model uczy się stosunkowo wolno ale też dokładnie. W przypadku mniejszej liczby tychże warstw szybko następuje przeuczenie (które udało nam się zwalczyć) i rozpoznawanie na żywo praktycznie nie działało.

7.3 Wykorzystywane próbki

W przeciwnieństwie do SVM aktualną predykcję określamy nie na podstawie poprzedniego stanu, lecz dla stanu sprzed pewnej chwili czasowej. W naszym przypadku aktualnie wyświetlany stan pochodzi sprzed około 0,25 s. Oczywiście powoduje to pewne opóźnienie, jednak praktycznie niezauważalne przez użytkownika. Podejście to nie ulega jednak propagacji błędów, którą widzimy w przypadku SVM.

7.4 Problemy

Przy rozpoznawaniu próbki zakładamy przynależność do jednej z dwóch klas. W przypadku gdy jako dane wejściowe mamy jedynie wdechy i wydechy nie stanowi to problemu. Gdy jednak chcemy rozpoznać brak oddychania musimy założyć pewien próg pewności co otrzymanej klasyfikacji. Eksperymentalnie przyjęliśmy iż wdech jest wdechem gdy pewność sieci neuronowej wynosi 99% natomiast co do wydechów przyjęliśmy pewność na poziomie 70%.

7.5 Rozwiążanie

Założenie co do pewności predykcji sprawdziło się dla pewnych danych, jednak wymagało specjalnego wytrenowania sieci tak, aby predykcja była na tyle nie-dokładna aby tło klasyfikowane było w mniej niż 70% jako wydech lub mniej niż 99% jako wdech. Przygotowując metodę do potencjalnego douczania, postanowiliśmy dodać więc trzeci stan - stan tła. Choć próbki wdechów i wydechów nie są idealnie przycięte a więc zawierają dźwięki tła, dodanie trzeciego stanu pozwoliło znacznie poprawić nam dokładność predykcji co zaowocowało lepszym rozpoznawaniem na żywo. Model reaguje także lepiej na douczanie (nie wymaga dostosowywania parametrów do konkretnego nauczania).

8 Przejście na inny model oddychania

8.1 Wydech nosem

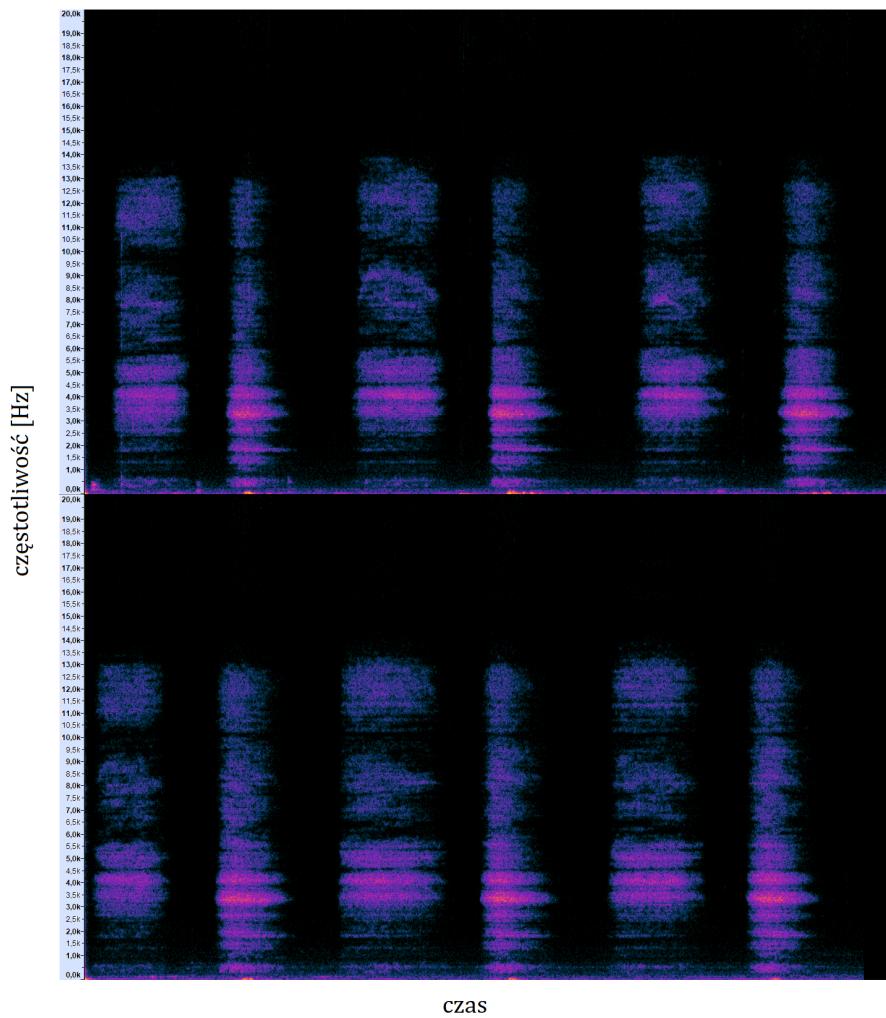
Kolejnym krokiem było przeniesienie wypracowanej metody uczenia na rozpoznawanie wdechu oraz wydechu nosem. W takim przypadku różnice między fazami oddechu są mniej zauważalne, co wymagało kolejnych udoskonaleń.

8.2 Kluczowe częstotliwości

Ponownie dokonaliśmy obserwacji spektrogramów, tym razem wyznaczonych dla wdechów i wydechów nosem. Zauważymy kilka zakresów, w których różnice między fazami oddechu są zauważalne:

- 0 – 8 500 Hz i 10 000 – 16 000 Hz,
- 0 – 3 000 Hz i 6 000 – 16 000 Hz,
- 0 – 16 000 Hz.

Spektrogram dźwięku oddechu



Rysunek 12: Spektrogramy dwóch nagrań oddechu z wdechem i wydechem nosem uzyskane przy użyciu programu Audacity.

Testy pokazały, iż ostatni z powyższych wybór częstotliwości spowodował największą poprawę większości wyników. Przedstawia je poniższa tabela:

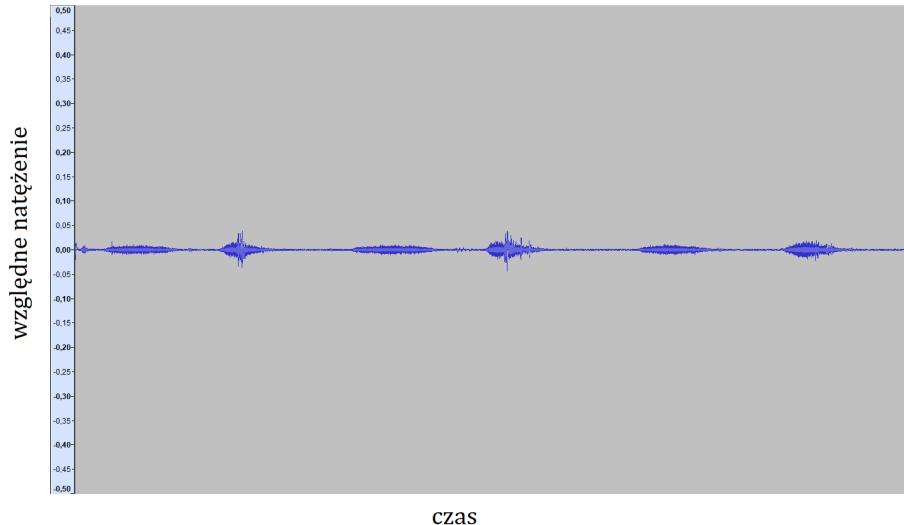
Tabela 3: Wartości miar uzyskane podczas testów modelu z **wydechem nosem** przy poprzednim (0–8 500 Hz i 10 000–16 000 Hz) oraz nowym (0–16 000 Hz) wyborze częstotliwości.

miara	wartość (%) poprzednia	wartość (%) nowa
$accuracy$	88	89
$precision_{wdech}$	89	92
$precision_{wydech}$	88	85
$recall_{wdech}$	92	89
$recall_{wydech}$	84	90
F_{wdech}	90	90
F_{wydech}	86	87

8.3 Usuwanie ciszy

Nietrudno było zaobserwować, iż znaczą część nagrani zajmuje cisza lub dźwięk bliski ciszy.

Natężenie dźwięku oddechu



Rysunek 13: Wykres natężenia dźwięku w nagraniu wdechu i wydechu nosem uzyskany przy użyciu programu Audacity.

Te przerwy między wyraźnymi dźwiękami oddechów pozostawały jednak oznaczone jako wdech lub wydech. Z tego powodu model trenowaliśmy w jednym przypadku informacją, iż cisza występuje podczas wdechu, a w innym – że

w trakcie wydechu. Podczas testów natomiast w jednym przypadku za poprawną klasyfikację ciszy przyjmowaliśmy wdech, a w innym – wydech.

Postanowiliśmy usunąć tę niedoskonałość z przypuszczeniem, iż poprawi to wyniki działania programu. Obliczyliśmy w tym celu sumę natężeń dźwięku w poszczególnych klatkach nagrania. Za próg ciszy przyjęliśmy wartość nieco większą niż maksymalne natężeń w klatkach pochodzących z tych fragmentów nagrania, które na podstawie własnych doświadczeń słuchowych oceniliśmy jako zbyt ciche. Następnie podczas trenowania oraz testowania usuwaliśmy z danych klatki o sumie natężeń nieprzekraczającej tego progu. Otrzymaliśmy poprawę wyników do następujących wartości:

Tabela 4: Wartości miar uzyskane podczas testów modelu z **wydechem nosem** przy wyborze częstotliwości 0–16 000 Hz i po usunięciu ciszy.

miara	wartość (%) poprzednia
<i>accuracy</i>	94
<i>precision_{wdech}</i>	96
<i>precision_{wydech}</i>	92
<i>recall_{wdech}</i>	95
<i>recall_{wydech}</i>	92
<i>F_{wdech}</i>	96
<i>F_{wydech}</i>	92

Postanowiliśmy także sprawdzić, jak podejście z usuwaniem ciszy sprawdzi się w poprzednim modelu – wdechu nosem i wydechu ustami. Okazało się, iż również nastąpiła poprawa.

8.4 Sieć neuronowa

W przypadku sieci neuronowej skupiliśmy się głównie na nagraniu dobrej jakości próbek oraz na dobrym wytrenowaniu tejże sieci. Choć wyniki nie są rewelacyjne, sieć zdaje się rozpoznawać zarówno wdech jak i wydech nosem jako osobne klasy. Dużym usprawnieniem okazało się natomiast wprowadzenie trzeciego stanu, o czym mowa w punkcie 7.5. Sieci zdarza się jednak mylić.

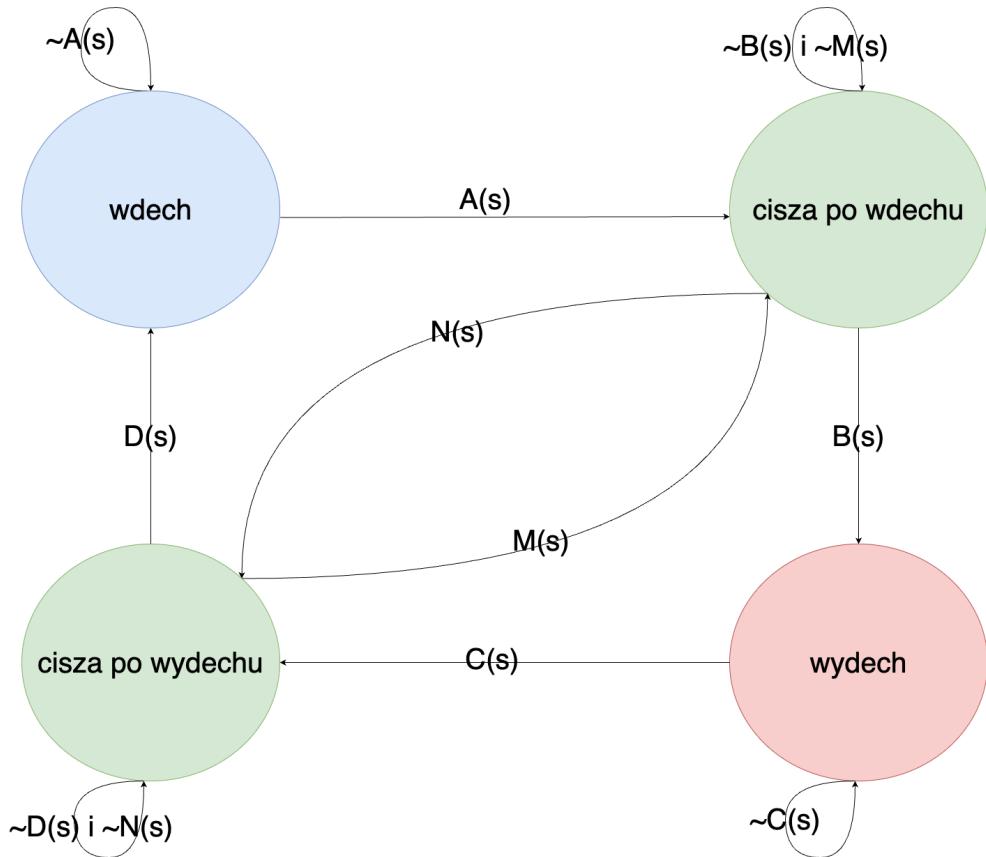
9 Maszyna stanów do zwiększenia inertji

9.1 Cel

Powyższe rozważania miały na celu głównie poprawę jakości klasyfikacji pojedynczej próbki, natomiast w praktyce, jaką jest klasyfikacja na żywo, nawet dokładne modele mogą powodować niechciane przeskakiwanie. Przypadek, w którym ktoś bierze wdech na ułamek sekundy w połowie wykonywania wydechu

jest nierealny, zatem możemy nadać wynikom naszego modelu pewnej bezwładności, co czynimy za pomocą przedstawionej w tej sekcji maszyny stanów dobudowanej na modelu i korzystającym z jego klasyfikacji do przechodzenia między kolejnymi stanami.

9.2 Schemat



Rysunek 14: Diagram stanów S , $s \in S$ oznacza stan diagramu a A, B, C, D, M, N są predykatami określającymi przejścia między poszczególnymi stanami

Diagram działa jak na schemacie powyżej, ponadto ma bufor stanowiący historię klasyfikacji modelu, np. $bufor = [in, out, out, out, cisza]$. Niech n - rozmiar bufora, i - ilość wdechów w buforze, o - ilość wydechów w buforze, t - ilość cisz w buforze. Ponadto zdefiniujemy zdarzenie X , takie, że $P(X) = P$ dla pewnego ustalonego P oraz wagę W będące parametrami maszyny stanów wpływającymi na inercję (dobrałyśmy $W = 60\%$, $P = 70\%$, $n = 10$). Dla tych parametrów

definiujemy predykaty

$$\begin{aligned}
 A(s) &= (\frac{t}{n} > W) \vee (\frac{o}{n} > W \wedge X) \\
 B(s) &= \frac{o}{n} > W \\
 C(s) &= (\frac{t}{n} > W) \vee (\frac{i}{n} > W \wedge X) \\
 D(s) &= \frac{i}{n} > W \\
 M(s) &= \frac{o}{n} > W \wedge X \\
 N(s) &= \frac{i}{n} > W \wedge X
 \end{aligned}$$

przykładowo dla stanu bufora [*in, cisza, cisza, cisza, cisza*] oraz aktualnego stanu $s = \text{wdech}$ i reszty parametrów przyjętych tak jak powyżej nastąpi przejście ze stanu wdech do stanu cisza po wdechu. Zdarzenie X tak samo jak predykaty M, N uwzględniono, by maszyna nie była cyklem, czyli uwzględniała też możliwość np. sekwencji stanów wdech, cisza po wdechu, wdech, gdyby zaszedł jakiś błąd.

9.3 Rezultat

Wyżej zdefiniowana maszyna stanów spełnia swój rezultat - przy jej zastosowaniu przeskakiwanie stanów w klasyfikacji na żywo zostaje zmniejszone. Warto jednak wspomnieć, że czynimy to kosztem wprowadzenia opóźnienia proporcjonalnego do wielkości bufora, teoretycznie jeśli ktoś będzie bardzo szybko oddychał to może nie zaobserwować zmian stanu. Brak zmiany stanów może też wystąpić przy nieregularnym oddechu np. kilka razy z rzędu wdech bez wydechu, jednak zapobiegają temu predykaty M, N .

10 Wnioski

Wniosek jeden: dużo zależy od obróbki danych, różnica między sieciami w tym wypadku nie była tak zauważalna

Wniosek dwa: trudno uzyskać inercję stanów oddechu i pozbyć się efektu przeskakiwania

Wniosek trzy: na wdech i wydech nosem wpływa bardziej zaszumione otoczenie

Wniosek cztery: dodać tabelkę z końcowymi wartościami miar dla obu modeli oddychania (SVM dla nosa i ust, Tensorflow dla ust)

11 Źródła

- [1] Cormen Thomas H., Leiserson Charles E., Rivest Roland L., Stein Clifford (1989) *Wprowadzenie do algorytmów*,

- [2] https://pl.wikipedia.org/wiki/Dyskretna_transformata_Fouriera
- [3] <https://esezam.okno.edu.pl/mod/book/view.php?id=9&chapterid=77>
- [4] Kumar, E. and Surya, K. and Varma, K. and Akash, A. and Kurapati, Nithish Reddy (2023) *Noise Reduction in Audio File Using Spectral Gattting and FFT by Python Modules*
- [5] <https://www.maplesoft.com/Applications/Detail.aspx?id=154593>
- [6] https://www.deltami.edu.pl/temat/informatyka/sztuczna_inteligencja/2017/12/28/Glebokie_uczenie_maszyn/
- [7] <http://sciadaprogramisty.blogspot.com/2018/03/fully-connected-layer-fc-warstwa-w-peni.html>
- [8] <https://www.tutorialspoint.com/keras>