

Rozpoznawanie Mówcy za Pomocą Sieci Neuronowych

Marcin Chwedczuk Albert Skłodowski

5 Styczeń 2012

Spis treści

| | | |
|----------|---|----------|
| 1 | Zadanie Rozpoznawania Mówcy | 1 |
| 2 | Zbiór Danych Uczących i Walidacyjnych | 1 |
| 3 | Opis Zastosowanych Algorytmów | 1 |
| 3.1 | Opis Wykorzystywanej Sieci Neuronowej | 1 |
| 3.2 | Opis Algorytmu Ekstrakcji Cech | 2 |
| 3.3 | Opis Algorytmu Rozpoznawania Mówcy | 3 |
| 4 | Omówienie Uzyskanych Wyników | 4 |
| 4.1 | Metodologia testowania | 4 |
| 4.2 | Wpływ Parametrów na Rezultaty | 4 |
| 4.3 | Uzyskane Wyniki | 4 |
| 5 | Podsumowanie | 4 |

1 Zadanie Rozpoznawania Mówcy

Celem naszego projektu była ocena przydatności sieci neuronowych w zadaniu rozpoznawania mówcy. TODO!

2 Zbiór Danych Uczących i Walidacyjnych

3 Opis Zastosowanych Algorytmów

3.1 Opis Wykorzystywanej Sieci Neuronowej

W stworzonej przez nas aplikacji przyjęliśmy następującą architekturę systemu: Dla każdej osoby (oznaczymy ją P) która ma być rozpoznawana przez aplikację zostanie stworzona osobna sieć neuronowa net_P . Sieć ta będzie uczona odpowiedzi 1 dla próbek głosu osoby P oraz odpowiedzi 0 dla próbek głosu pozostałych osób. Wszystkie sieci będą uczone i testowane w ten sam sposób.

Każda z sieci net_i będzie siecią typu perceptron wielowarstwowy oraz będzie posiadać dwie warstwy ukryte. O wielkości warstw ukrytych będą decydować parametry D_1 oraz D_2 w następujący sposób:

$$size(hidden_1) = \frac{size(input)}{D_1}$$

$$size(hidden_2) = \frac{size(input)}{D_2}$$

gdzie $size(hidden_1)$, $size(hidden_2)$ to odpowiednio rozmiary I i II warstwy ukrytej a $size(input)$ to rozmiar wejścia. Zazwyczaj będziemy przyjmować $1 \leq D_1 < D_2$. Każda z sieci będzie posiadała dokładnie jedno wyjście zwracające wartości z przedziału $[0, 1]$.

W opisanych wyżej sieciach będziemy stosować neurony o sigmoidalnej funkcji aktywacji ($f(x) = \frac{1}{1+\exp(-x)}$), dodatkowo będziemy stosować wzmocnienie(bias).

Każda z sieci będzie uczona za pomocą standardowego algorytmu wstecznej propagacji błędów (backpropagation). W algorytmie backpropagation będziemy stosować stopniowy rozpad współczynnika nauki oraz momentu, zgodnie z wzorami:

$$\eta_i = \frac{\eta}{\sqrt{(1+i)}}$$

$$\mu_i = \frac{\mu}{\sqrt{(1+i)}}$$

gdzie η oznacza współczynnik nauki, μ współczynnik momentu, a i to oczywiście numer iteracji. Zazwyczaj będziemy przyjmować $\mu = \frac{\eta}{2}$.

W dalszej części dokumentu gdy nie będzie powiedziane inaczej będziemy przyjmować że $D_1 = 4$, $D_2 = 8$, $\eta = 0.21$ a $\mu = 0.1$.

Sieci neuronowe w naszej aplikacji zostały zaimplementowane za pomocą darmowej do zastosowań niekomercyjnych biblioteki Encog w wersji 3.0.1.

Algorytm obliczania błędu sieci na zbiorze walidacyjnym

Przedstawiony na listingu 1 sposób obliczania błędu sieci neuronowej opiera się na standardowym błędzie średnio kwadratowym, z tym że odpowiedzi sieci które znajdują się zbyt blisko wartości 0.5 nie są uwzględniane przy liczeniu błędu. Co ma sens ponieważ nie są one uwzględniane również przy typowaniu najbardziej prawdopodobnego mówcy.

W wielu miejscach będziemy prezentować błąd aplikacji(czyli całego zestawu sieci) na zbiorze uczącym lub walidacyjnym, błąd ten będzie obliczany jako średnia arytmetyczna z błędów poszczególnych sieci na zadanym zbiorze.

3.2 Opis Algorytmu Ekstrakcji Cech

Opis Danych Wejściowych Algorytmu

Algorytm ekstrakcji cech operuje na pojedynczych próbkach dźwięku. Każda próbka niezależnie do długości jest reprezentowana jako tablica liczb zmiennoprzecinkowych o wartościach z przedziału $[-1, 1]$. Algorytm zakłada ponadto że

Algorytm 1 Algorytm obliczający błąd sieci neuronowej na zadanym zbiorze testowym

```
1: input testedPerson
2:
3: error  $\leftarrow$  0
4: count  $\leftarrow$   $\epsilon$  { Epsilon is a small number e.g. 0.0001}
5: for all person  $\in$  People do
6:   answer  $\leftarrow$  0
7:   if person = testedPerson then
8:     answer  $\leftarrow$  1
9:   end if
10:
11:   {testSet - Array containing mfcc coefficients for consecutive voice frames}
12:   testSet  $\leftarrow$  GetPersonTestSet(person)
13:   for  $0 \leq start < length(testSet) - \text{FRAMES-COUNT}$  do
14:     netInput  $\leftarrow$  testSet[start..(start + FRAMES-COUNT-1)]
15:     netAnswer =  $network_{person}$ (netInput)
16:     if netAnswer > T or netAnswer < (1 - T) then
17:       error  $\leftarrow$  error + (netAnswer - answer)2
18:       count  $\leftarrow$  count + 1
19:     end if
20:   end for
21: end for
22:
23: return error / count
```

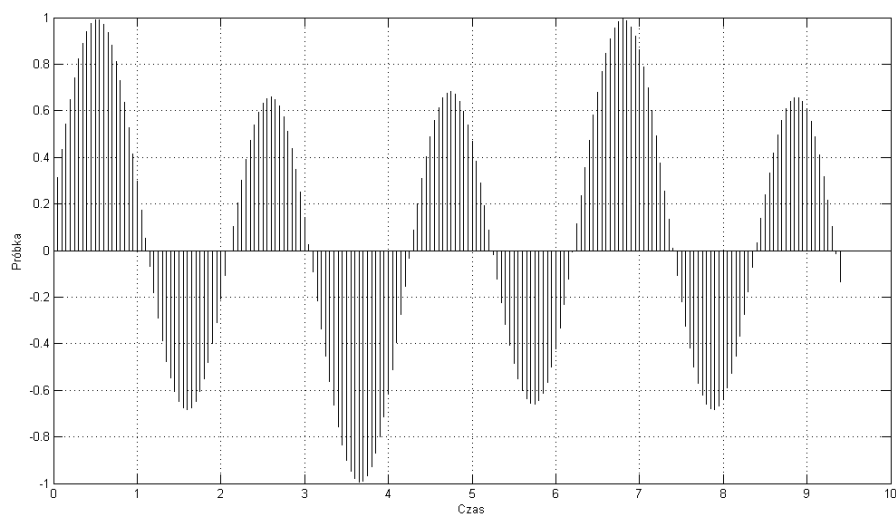
próbki zostały otrzymane przez kwantyzację sygnału ciągłego ze stałą częstotliwością próbkowania równą F . Konwersji plików WAVE na wyżej opisany format dokonaliśmy za pomocą darmowej biblioteki NAudio.

Normowanie Sygnału Wejściowego

Okazuje się że w zależności od mówcy (a także jego nastroju, czy zmęczenia), oddalenia od mikrofonu oraz innych czynników można otrzymać dość znaczne różnice w amplitudzie nagrywanych sygnałów dźwiękowych. Aby wyeliminować wpływ amplitudy sygnału na jakość klasyfikacji wszystkie przetwarzane sygnały przed ekstrakcją cech są poddawane normowaniu. Algorytm normujący sygnał wejściowy został przedstawiony na listingu 2.

Algorytm Ekstrakcji Cech

W tej części dokumentu przedstawimy algorytmy służące do wyodrębniania charakterystycznych cech mowy, omówimy również towarzyszące im parametry. Tabel 1 zawiera opis parametrów algorytmów, listingi 3, 4 oraz 5 prezentują pseudokody wykorzystywanych algorytmów.



Rysunek 1: Reprezentacja sygnału ciągłego za pomocą tablicy liczb zmienno-przecinkowych. Kolejne elementy tablicy zawierają wartości kolejnych próbek sygnału, i tak element o indeksie 0 zawiera wartość 0.0, element o indeksie 1 wartość 0.31, element o indeksie 2 wartość 0.43 itd.

3.3 Opis Algorytmu Rozpoznawania Mówcy

Listing 6 zawiera pseudokod algorytmu wykorzystywanego do wytypowania najbardziej prawdopodobnego mówcy, tabela 2 zawiera opis parametrów występujących w algorytmie.

4 Omówienie Uzyskanych Wyników

4.1 Metodologia testowania

4.2 Wpływ Parametrów na Rezultaty

4.3 Uzyskane Wyniki

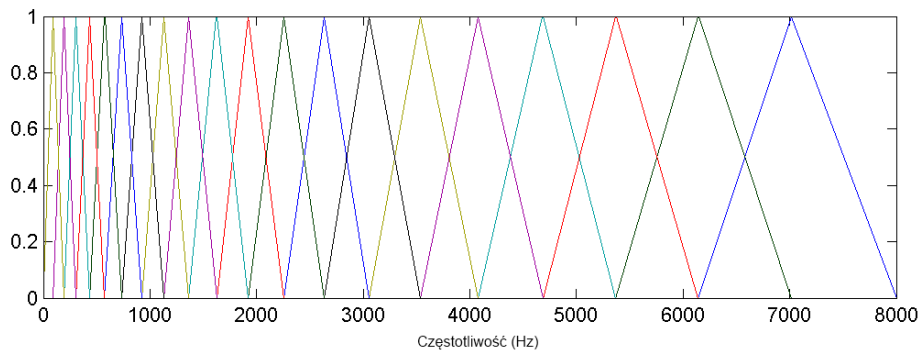
5 Podsumowanie

Algorytm 2 Algorytm wykorzystywany do unormowania wartości sygnału wejściowego

```

1:  $\{samples_i$  - Array containing voice samples $\}$ 
2:  $emph \leftarrow 1.0$ 
3:  $max \leftarrow \max_i(samples_i)$ 
4:  $min \leftarrow \min_i(samples_i)$ 
5:
6: if  $|min| > \epsilon$  then
7:    $emph \leftarrow \max(emph, \frac{1}{|min|})$ 
8: end if
9: if  $|max| > \epsilon$  then
10:   $emph \leftarrow \min(emph, \frac{1}{|max|})$ ;
11: end if
12:
13: for all  $i$  do
14:    $samples_i \leftarrow emph \cdot samples_i$ 
15: end for

```



Rysunek 2: Położenie i przepustowość banku filtrów trójkątnych rozłożonych zgodnie ze skalą MEL na przedziale częstotliwości 0Hz - 8kHz

| Parametr | Optymalna wartość | Opis parametru |
|-----------------------|-------------------|---|
| WINDOW-SIZE | 512 - 1024 | Określa ile próbek tworzy pojedynczą ramkę. Wielkość okna powinna być tak dobrana żeby czas trwania ramki wynosił od 10 do 30 milisekund. Wartość tego parametru powinna być potęgą liczby 2 |
| WINDOW-OVERLAP | 256 - 512 | Określa ile próbek współdzielią ze sobą sąsiadujące ramki |
| SPEAK-POWER-THRESHOLD | 0.02 - 0.06 | Parametr określa minimalną energię jaką musi posiadać sygnał dźwiękowy żeby został uznany za część wypowiedzi |
| MFCC-COUNT | 6 - 16 | Ilość używanych współczynników MFCC |
| $ \Delta $ | 24 - 27 | Ilość używanych filtrów trójkątnych. Filtry pokrywają pasmo częstotliwości od 0 Hz do 8 kHz zgodnie ze skalą MEL. Aby utworzyć zbiór filtrów wymagana jest znajomość częstotliwości próbkowania F , oraz rozmiar ramki WINDOW-SIZE. Rysunek 2 przedstawia przepustowość oraz rozmieszczenie przykładowego zestawu filtrów |
| Δ_i^n | | Przepustowość filtra o numerze n dla częstotliwości odpowiadającej i -temu wyjściu transformacji Furiera |

Tabela 1: Opis parametrów algorytmu ekstrakcji cech

Algorytm 3 Algorytm wykorzystywany do ekstrakcji cech z sygnału wejściowego

```

1: input samples : array [0..( $N - 1$ )] of double
2:
3: {Type mfcc is used to represent array of MFCC coefficients}
4: type mfcc = array [0..(MFCC-COUNT-1)] of double
5: features : list of mfcc
6: start  $\leftarrow$  0
7: delta  $\leftarrow$  WINDOW-SIZE - WINDOW-OVERLAP
8:
9: while start < N - WINDOW-SIZE do
10:   frame  $\leftarrow$  ExtractFrame(samples, start)
11:   power  $\leftarrow \sum_i |frame_i| / \text{WINDOW-SIZE}$ 
12:   if power > SPEAK-POWER-THRESHOLD then
13:     mfcc  $\leftarrow$  GetMFCC(frame)
14:     append mfcc to features
15:   end if
16:   start  $\leftarrow$  start + delta
17: end while
18:
19: return features

```

Algorytm 4 ExtractFrame - Procedura pomocnicza

```

1: input samples : array [0..( $N - 1$ )] of double
2: input start : integer
3:
4: frame : array [0..(WINDOW-SIZE-1)] of double
5: for  $0 \leq i < \text{WINDOW-SIZE}$  do
6:    $frame_i = samples_{start+i}$ 
7: end for
8:
9: return frame

```

Algorytm 5 GetMFCC - Algorytm obliczania współczynników MFCC dla danej ramki

```

1: input frame : array  $[0..(N-1)]$  of double
2:
3: for  $0 \leq i < N$  do
4:    $hamming_i \leftarrow 0.54 - 0.46 \cos(\frac{2\pi i}{N-1})$ 
5:    $frame_i \leftarrow hamming_i \cdot frame_i$ 
6: end for
7:
8:  $furier \leftarrow \mathbf{FFT}(frame)$ 
9: for  $0 \leq i < N$  do
10:   $furier_i \leftarrow |furier_i|$ 
11: end for
12:
13: for  $0 \leq i < |\Delta|$  do
14:   $sum_i \leftarrow 0$ 
15:  for  $0 \leq j < N$  do
16:     $sum_i \leftarrow sum_i + \Delta_j^i \cdot furier_j$ 
17:  end for
18:   $sum_i \leftarrow \log_{10}(sum_i)$ 
19: end for
20:
21: {Return only first MFCC-COUNT elements of array returned by DCT}
22: return DCT(sum)

```

| Parametr | Optymalna wartość | Opis parametru |
|--------------------|-------------------|---|
| T | unknown | Próg wiarygodności sieci neuronowej. Odpowiedz sieci będzie uznana za znaczącą wtedy gdy będzie należeć do sumy przedziałów $[0, 1 - T]$ oraz $[T, 1]$ |
| Features | | Tablica współczynników MFCC dla ramek rozpoznawanego sygnału mowy |
| People | | Zbiór osób rozpoznawanych przez klasyfikator |
| $network_{person}$ | | Sieć neuronowa nauczona rozpoznawania osoby <i>person</i> oraz odrzucania pozostałych osób |
| FRAMES-COUNT | 6 - 16 | Parametr określający ile kolejnych ramek (a raczej odpowiadających im tablic współczynników MFCC) jest podawanych na wejście sieci neuronowej. Wielkość wejścia sieci można policzyć mnożąc FRAMES-COUNT przez MFCC-COUNT |

Tabela 2: Opis parametrów algorytmu wyboru zwycięzcy

Algorytm 6 RecogniseSpeaker - Algorytm rozpoznawania mówcy

```
1: input Features : array [0.. $(N - 1)$ ] of mfcc
2:
3: total  $\leftarrow 0$ 
4: for all person  $\in$  People do
5:    $result_{person} \leftarrow 0$ 
6: end for
7:
8: for all  $0 \leq \text{start} < N - \text{FRAMES-COUNT}$  do
9:   {Skopiuj elementy tablicy features do tablicy netinput}
10:  {Kopiujemy również oba indeksy graniczne}
11:  netinput  $\leftarrow$  Features[start.. $(\text{start} + \text{FRAMES-COUNT} - 1)$ ]
12:
13:  for all person  $\in$  People do
14:    answer =  $network_{person}(\text{netinput})$ 
15:    if answer  $> T$  or answer  $< (1 - T)$  then
16:      if answer  $> T$  then
17:         $result_{person} \leftarrow result_{person} + \text{answer}$ 
18:        total  $\leftarrow \text{total} + 1$ 
19:      else
20:         $result_{person} \leftarrow result_{person} + (\text{answer} - 1)$ 
21:      end if
22:      {Notice that total is incremented only in the first branch}
23:    end if
24:  end for
25: end for
26:
27: for all person  $\in$  People do
28:    $result_{person} \leftarrow \max(0, result_{person}/\text{total})$ 
29: end for
30: sort descend person by  $result_{person}$ 
```
