Aufgabe 3: Mel-Skalierte Dreiecksfilterbank

Das FFT-Spektrum eines Signals hat eine lineare Frequenzskalierung. Um die menschliche Wahrnehmung zu imitieren, wird häufig eine gehörähnliche Frequenzskalierung bevorzugt. Diese kann man beispielsweise mit einer Mel-Filterbank berechnen. Dazu berechnet man zunächst das Kurzzeitspektrogramm wie in Aufgabe 2 und summiert dann jeweils innerhalb von einem Filterband alle Spektralanteile gewichtet auf. Dazu wird als Gewichtungsfunktion oft eine Dreiecksfunktion verwendet. Abbildung 1 zeigt das Aussehen einer typischen Dreiecksfilterbank.

Implementierung der benötigten Hilfsfunktionen

3.1 Implementieren Sie eine Funktion

```
def hz_to_mel(x):
    ...
```

in der Datei `tools.py`, die einen Frequenzwert `x` in Hz mit Hilfe der Beziehung

$$\mathrm{Mel}(f) = 2595 \log_{10}(1 + f/700)$$

als entsprechenden Wert auf der Mel-Skala zurückgibt. Die Funktion soll sowohl skalare Eingabewerte, als auch `numpy` Arrays verarbeiten können.

3.2 Implementieren Sie eine Funktion

```
def mel_to_hz(x):
    ...
```

in der Datei `tools.py`, die analog zur in Aufgabe 2.4 implementierten Funktion die Umwandlung von einer Frequenz in der Mel-Skala `x` in die lineare Frequenzskala in Hz realisiert. Die Funktion soll sowohl skalare Eingabewerte, als auch `numpy` Arrays verarbeiten können.

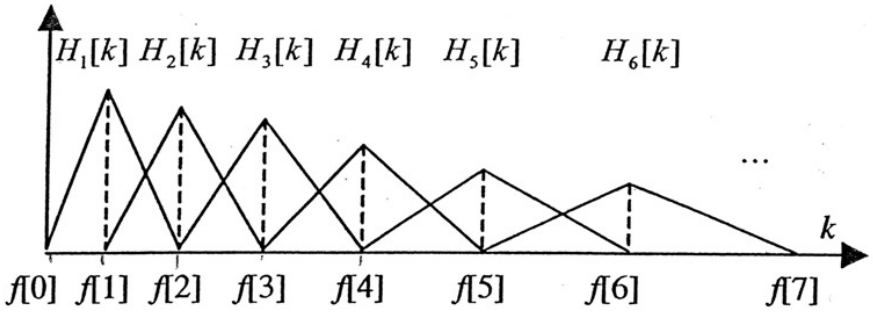

Figure 1: Dreiecksfilterbank, aus: Huang, Acero, Hon: *Spoken Language Processing*

Implementierung der Dreiecksfilterbank

3.3 Implementieren Sie eine Funktion

```
def get_mel_filters(sampling_rate, window_size_sec, n_filters,
                    f_min=0, f_max=8000):
    ...
```

in der Datei `feature_extraction.py`, die eine Dreiecksfilterbank in Form einer Matrix für die jeweiligen Eingabeparameter zurückgibt. Der Parameter `sampling_rate` bezeichnet hierbei die Abtastfrequenz in Hz, `window_size_sec` ist die Fensterbreite in Sekunden, `n_filters` beschreibt die Anzahl der Dreiecksfilter und `f_min` und `f_max` geben die minimale bzw. maximale Frequenz in Hz an, die von der Dreiecksfilterbank abgedeckt werden soll.

Wie in Abb. 1 zu sehen ist, werden für eine Dreiecksfilterbank mit M Filtern jeweils $M+2$ Frequenz-Stützstellen benötigt. Diese Stützstellen werden so berechnet, dass sie

- im Mel-Frequenzbereich angenähert äquidistant sind und
- genau mit Frequenzstützstellen im Fourierbereich übereinstimmen (M: Anzahl der Mel-Filter, im Beispiel in Abb. 1 ist $M = 6$.)

Die Gewichte der Dreiecksfilter sollen so bestimmt werden, dass sie

- bei der Mittenfrequenz maximal sind,
- bei den Mittenfrequenzen der angrenzenden Bänder Null sind und
- dass jedes von ihnen die gleiche Fläche hat.

Legen Sie hierzu bei der Implementierung der Funktion zunächst die Mittenfrequenzen aller Mel-Filter $f_{Hz}[i]$, durch Wahl der nächstgelegenen linearen Frequenzstützstelle (in Hz) fest. Verwenden Sie hierbei die zuvor implementierten Hilfsfunktionen.

Ermitteln Sie dann die zugehörigen DFT-Indizes $f[i]$, wählen Sie dabei $f[0] = 0$ und $f[M+1] = \frac{N}{2}$, wobei N der FFT-Größe entspricht. Diese beiden Filterausgänge dienen nur der Berechnung und werden bei der Ausgabe nicht mehr benötigt.

Anschließend lassen sich die Dreiecksfilter mit Hilfe der folgenden Berechnungsvorschrift bestimmen:

`Für alle m = 1 bis M und k = 0 bis N/2:`

$$H_m[k] = \begin{cases} 0 & \text{falls } k < f[m-1] \\ \dfrac{2(k - f[m-1])}{(f[m+1] - f[m-1])(f[m] - f[m-1])} & \text{falls } f[m-1] \leq k < f[m] \\ \dfrac{2(f[m+1] - k)}{(f[m+1] - f[m-1])(f[m+1] - f[m])} & \text{falls } f[m] \leq k \leq f[m+1] \\ 0 & \text{falls } k > f[m+1] \end{cases}$$

3.4 Implementieren Sie eine Funktion

```
def apply_mel_filters(abs_spectrum, filterbank):
    ...
```

in der Datei `feature_extraction.py`, die eine mit Hilfe der zuvor implementierten Funktion `get_mel_filters()` berechnete Dreiecksfilterbank $H_m[k]$ auf ein Betragsspektrum $S_{\text{LIN}}[k, \tau]$ (wie es in Aufgabe 2.1 berechnet wurde), gemäß

$$S_{\text{MEL}}[m, \tau] = \sum_{k=1}^{K} H_m[k] S_{\text{LIN}}[k, \tau]$$

anwendet. Hierbei bezeichnet τ den jeweiligen Rahmenindex.

3.5 Erweitern Sie nun die Funktion zur Merkmalsextraktion `compute_features()` in der Datei `feature_extraction.py`, so dass diese für ein gegebenes Audiosignal die

mit Hilfe von `np.log()` logarithmierten Mel-Filterbank Features zurückgibt, wenn für das Argument `feature_type` der Wert `FBANK` angegeben wird. Die optionalen Argumente `n_filters`, `fbank_fmin` und `fbank_fmax` beziehen sich hierbei auf die entsprechenden Parameter der Mel-Filterbank (s. Aufgabe 2.6). Den Eingabeparameter `num_ceps` können Sie weiterhin ignorieren.

Darstellung der Filterbank und der extrahierten Merkmale

3.6 Erweitern sie das Skript in der Datei `uebung2.py`, so dass mit Hilfe der Funktion `get_mel_filters()` berechnete Dreiecksfilterbank darstellt. Verwenden Sie für die Darstellung eine Abtastfrequenz von 16 kHz, eine Fensterlänge von 25 ms und eine Filteranzahl von 24. Verwenden Sie die Standardeinstellungen für die Grenzfrequenzen. Abb. 2 zeigt eine Darstellung des gewünschten Ergebnisses.

3,7 Erweitern sie das Skript in der Datei `uebung2.py`, so dass zusätzlich zur Darstellung der Dreiecksfilterbank mit Hilfe der Funktion `compute_features()` das Mel-Spektrum der Audiodatei `TEST-MAN-AH-3O33951A.wav` in einem separaten Plot dargestellt wird. Verwenden Sie hierbei die identischen Einstellungen wie in Aufgabe 2.9 und wählen Sie zusätzlich einen Rahmenvorschub von 10 ms. Das Ergebnis sollte in etwa wie in Abb. 3 dargestellt aussehen.

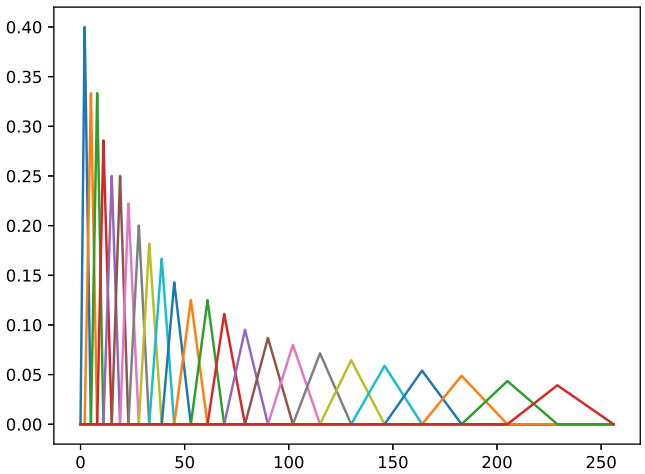

Figure 2: Darstellung der Dreiecksfilter gemäß Aufgabe 3.6.

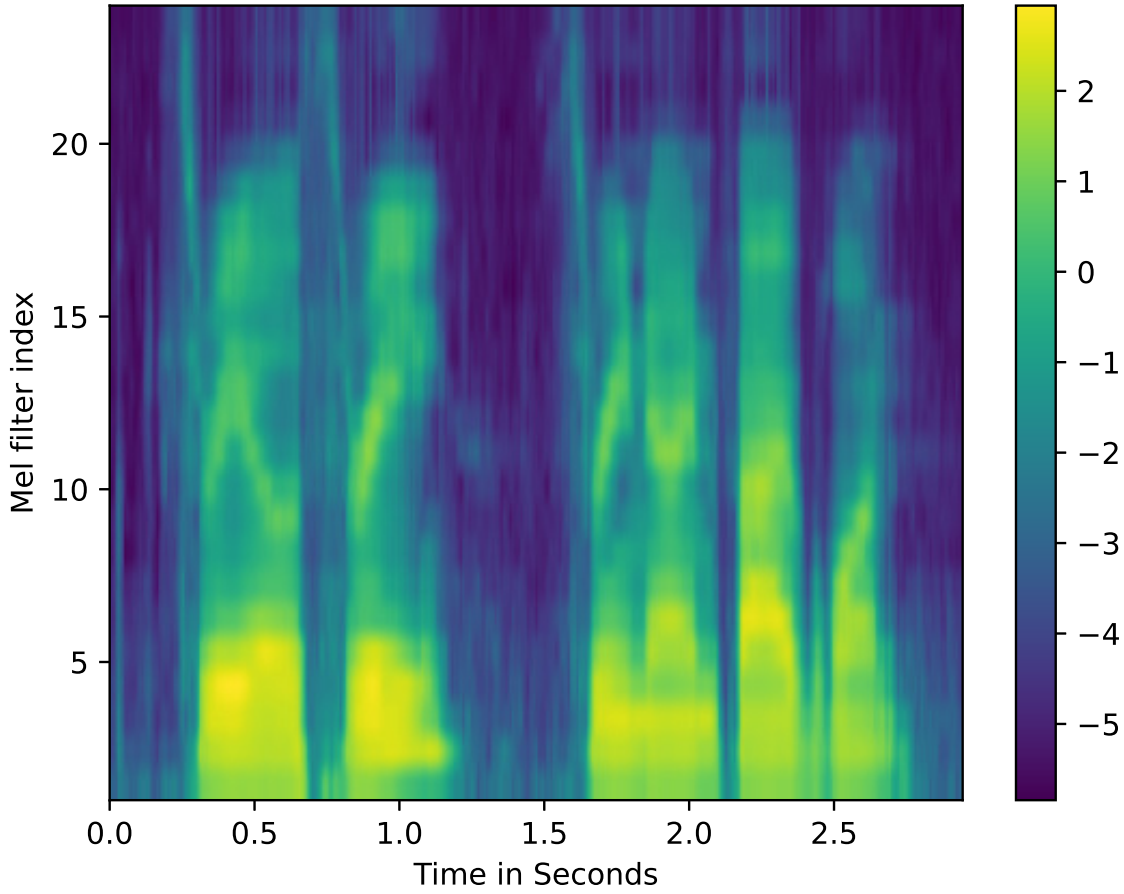

Figure 3: Darstellung des Mel-Spektrums gemäß Aufgabe 3.7.