EIN QUANTENLOGISCH MOTIVIERTER ANSATZ ZUR VERARBEITUNG VON ÄUSSERUNGS-BEDEUTUNGSPAAREN

Markus Huber-Liebl[1], *Günther Wirsching*[2]

[1]*Brandenburgische Technische Universität Cottbus-Senftenberg*
[2]*Katholische Universität Eichstätt-Ingolstadt*
markus.huber@b-tu.de

Kurzfassung: Wir repräsentieren eine Bedeutung als Liste von Mustersignalen, und unser Ziel ist es, ein weiteres ankommendes Signal damit zu vergleichen. Die Quantenlogik motiviert die Verwendung von Orthogonalprojektoren, um die gesuchte Ähnlichkeit als Projektionswahrscheinlichkeit darzustellen. Die Ergebnisse des quantenlogischen Verfahrens hängen davon ab, in welcher Weise die Signale vorverarbeitet werden. In diesem Aufsatz untersuchen und diskutieren wir vier verschiedene Möglichkeiten der Vorverarbeitung.

1 Motivation

In [1] haben wir einen Petrinetzalgorithmus vorgestellt, der auf den Ideen der Maschinensemiotik [2] beruht und eine Maschine in die Lage versetzt, die Bedeutung von gesprochenen Äußerungen im laufenden Betrieb zu erlernen. In unserem Algorithmus muss die Maschine nach einer Äußerung des Benutzers feststellen, ob eine solche Äußerung bereits gespeichert ist. Für diese Aufgabe ist es naheliegend, einen „Ende-zu-Ende“ Ansatz zu verwenden, der mit wenigen gelabelten Trainigsdaten auskommt und kein ausgearbeitetes Sprachmodell benötigt. Baevski et al. [3] beschreiben ein Verfahren, das nach einem Vor-Training mit nicht gelabelten Daten über ein „fine tuning“ mit verhältnismäßig wenigen gelabelten Daten zu einem akzeptablen Sprachmodell kommt.

Wir schlagen hier eine Methode vor, die auf Quantenlogik beruht und völlig ohne Sprachmodell auskommt. Unsere Ergebnisse sind für einen ersten Versuch durchaus vielversprechend.

2 Quantenlogik, Wahrnehmung und Schichtenmodell

Das Lehrbuch [4] enthält eine mathematische Analyse der Logik, die, ausgehend von der Begriffslogik des Aristoteles, eine Verbindung bis zur Quantenlogik herstellt. Das Resultat ist, dass jede Logik sich auf der Grundlage der mathematischen Struktur *orthomodularer Verband* beschreiben lässt. Daraus entsteht die Idee, mögliche Bedeutungen sprachlicher Äußerungen mathematisch als Elemente der Trägermenge eines orthomodularen Verbands $\mathbb{V}$ zu modellieren. Nun folgt aus den Darstellungssätzen von Piron [5] und Solèr [6], dass zu jedem (genügend reichhaltigen) orthomodularen Verband $\mathbb{V}$ ein Hilbertraum $\mathscr{H}$ derart existiert, dass jedes Element der Trägermenge von $\mathbb{V}$ zu einem Orthogonalprojektor auf einen Untervektorraum von $\mathscr{H}$ korrespondiert. Die abstrakte Konstruktion des Hilbertraums $\mathscr{H}$ beruht dabei auf dem Koordinatisierungssatz der projektiven Geometrie und ist nicht direkt algorithmisch umsetzbar.

Andererseits beginnt die menschliche Verarbeitung akustischer Signale mit dem Hören. Die Physiologie der menschlichen Ohren legt nahe, dass die Vorverarbeitung akustischer Signale im Ohr mathematisch mit einer Frequenzanalyse modelliert werden kann, der „State-of-the-art“ ist hier Mel Frequency Cepstral Coefficients (MFCC) [7]. Um die Bedeutung des akustischen

 DOI: 10.35096/othr/pub-7085, URN: urn:nbn:de:bvb:898-opus4-70857

Signals zu ermitteln, verwenden wir die in Abbildung 1 dargestellte Strukur. Diese ist motiviert durch Kahnemann [8], wobei die neuronale Verarbeitung dem „schnellen Denken“ und die logische Verarbeitung dem „langsamen Denken“ entspricht. Für die Entscheidung, welche Bedeutung dem akustischen Signal zuzuordnen ist, kann der Hörer entweder nur die neuronale Verarbeitung oder nur die logische Verarbeitung oder beides zugrunde legen.

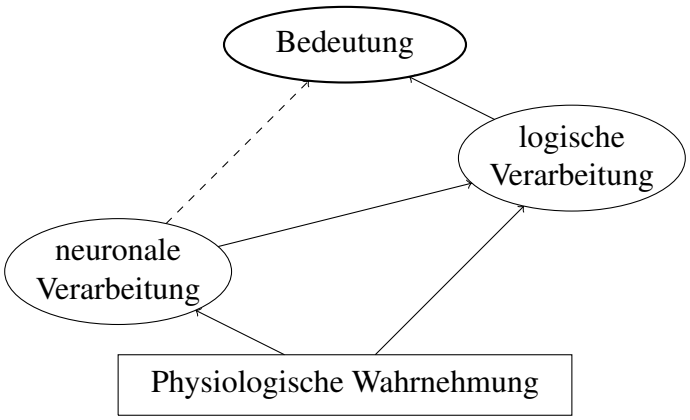

Abbildung 1 – Strukturdiagramm zur Verarbeitung physiologischer Wahrnehmungen

Der Platz eines Sprachmodells wäre in diesem Bild zwischen der neuronalen und der logischen Verarbeitung: je besser das Sprachmodell zur Ermittlung von Bedeutungen geeignet ist, umso näher liegt es an der logischen Verarbeitung. Die Ausgabe von Wortfolgen mit geringer Wortfehlerquote reicht im Allgemeinen noch nicht zur Bestimmung von Bedeutungen.

Unser Ansatz zum maschinellen Lernen verwendet für die logische Verarbeitung ein von der Quantenlogik motiviertes Modell. Dieses ermöglicht es, die Ähnlichkeit von Äußerungen durch numerisch berechenbare Projektionswahrscheinlichkeiten zu messen, was im Folgenden durchgeführt wird.

3 Projektionswahrscheinlichkeiten in $L^2([0,T])$: Definition

Die mathematische Struktur zur Beschreibung der logischen Schicht ist der Hilbert-Raum $\mathscr{H} = L^2([0,T])$ bestehend aus denjenigen komplexen Funktionen auf einem Intervall $[0,T]$ für ein geeignetes $T > 0$, deren Betragsquadrat $|u(t)|^2 = u(t)\,\overline{u(t)}$ integrierbar ist, das heißt,

$$\int_0^T u(t)\,\overline{u(t)}\,dt < \infty,$$

wobei wir für eine komplexe Zahl $z = x + iy$ deren Konjugierte mit $\bar{z} := x - iy$ bezeichnen. In dieser Struktur ist es möglich, durch n gegebene Funktionen $u_1, \ldots, u_n \in L^2([0,T])$ einen Untervektorraum als lineare Hülle zu erzeugen:

$$U_B := \operatorname{span}(u_1, \ldots, u_n) = \left\{ \sum_{i=1}^{n} \lambda_i u_i \,\middle|\, \lambda_i \in \mathbb{C} \right\} \subseteq L^2([0,T]).$$

Außerdem ist in $L^2([0,T])$ durch die Formel

$$\langle u, v \rangle := \int_0^T u(t)\,\overline{v(t)}\,dt$$

ein Skalarprodukt gegeben. Man sagt, zwei komplexe Funktionen u und v stehen aufeinander senkrecht, wenn $\langle u, v\rangle = 0$. Dadurch wird es möglich, zu einem beliebigen Vektor $\tilde{u} \in L^2([0,T])$ die orthogonale Projektion

$$P_B : L^2([0,T]) \to U_B$$

zu definieren. Die Formel für die Projektionswahrscheinlichkeit von $\tilde{u}$ auf U_B ist nun

$$p_B(\tilde{u}) := \frac{\langle P_B\tilde{u}, P_B\tilde{u}\rangle}{\langle \tilde{u}, \tilde{u}\rangle}. \tag{1}$$

4 Projektionswahrscheinlichkeiten in $L^2([0,T])$: Berechnung

Ist nun $\tilde{u} \in L^2([0,T])$ ein aus dem ankommenden Signal errechneter Vektor, so ist seine Orthogonalprojektion $P_B\tilde{u}$ durch zwei Bedingungen gegeben:

1. $P_B\tilde{u} \in U_B$, das heißt, es gibt Koeffizienten $\beta_1, \ldots, \beta_n \in \mathbb{C}$ mit der Eigenschaft

$$P_B\tilde{u} = \sum_{i=1}^{n} \beta_i u_i, \tag{2}$$

   und

2. der Differenzvektor $P_B\tilde{u} - \tilde{u}$ steht auf jedem der Mustervektoren u_i senkrecht, also

$$\forall j \in \{1, \ldots, n\} : \quad \langle u_j, P_B\tilde{u}\rangle = \langle u_j, \tilde{u}\rangle.$$

Durch diese beiden Bedingungen ist die Orthogonalprojektion als linearer Operator

$$P_B : L^2([0,T]) \to U_B$$

eindeutig festgelegt und durch Gleichung (2) berechenbar, wenn es gelingt, $\beta_1, \ldots, \beta_n$ zu bestimmen – die Koeffizienten β_i sind nur dann eindeutig festgelegt, wenn die Mustervektoren $u_1, \ldots, u_n$ linear unabhängig sind, was wir nicht voraussetzen.

Setzt man die erste Bedingung in die linke Seite der Gleichungen der zweiten Bedingung ein, so erhält man

$$\forall j \in \{1, \ldots, n\} : \quad \langle u_j, P_B\tilde{u}\rangle = \sum_{i=1}^{n} \langle u_j, \beta_i u_i\rangle = \sum_{i=1}^{n} \langle u_j, u_i\rangle \overline{\beta_i}.$$

Daraus folgt, dass der komplex konjugierte Koeffizientenvektor

$$\vec{x} := \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix} = \begin{pmatrix} \overline{\beta_1} \\ \vdots \\ \overline{\beta_n} \end{pmatrix} \in \mathbb{C}^n$$

eine Lösung eines komplexen linearen Gleichungssystems

$$A\vec{x} = \vec{b} \tag{3}$$

ist, wobei die Matrix A und die rechte Seite $\vec{b}$ wie folgt gegeben sind:

$$A := \begin{pmatrix} \langle u_1, u_1\rangle & \cdots & \langle u_1, u_n\rangle \\ \vdots & & \vdots \\ \langle u_n, u_1\rangle & \cdots & \langle u_n, u_n\rangle \end{pmatrix} \in \mathbb{C}^{n\times n} \quad \text{und} \quad \vec{b} := \begin{pmatrix} \langle u_1, \tilde{u}\rangle \\ \vdots \\ \langle u_n, \tilde{u}\rangle \end{pmatrix} \in \mathbb{C}^n.$$

Da wir nicht voraussetzen, dass die Mustervektoren $u_1, \ldots, u_n$ linear unabhängig sind, muss die Matrix A nicht invertierbar sein. Wir können also nicht ohne Weiteres einen Standardalgorithmus zur Lösung eines linearen Gleichungssystems verwenden. Aufgrund der geometrischen Vorgehensweise bei der Konstruktion der Matrix A wissen wir jedoch, dass (3) mindestens eine Lösung besitzt.

Zur Berechnung von $\vec{x}$ verwenden wir daher die QR-Zerlegung der Matrix A, das heißt, wir ermitteln eine unitäre Matrix $Q \in \mathbb{C}^{n\times n}$ und eine komplexe obere Dreiecksmatrix $R \in \mathbb{C}^{n\times n}$ mit der Eigenschaft

$$A = QR.$$

Die Matrizen Q und R lassen sich beispielsweise in Python [9] mit NumPy [10] auch dann bestimmen, wenn die Matrix A nicht invertierbar ist. Da die Matrix Q unitär ist, ist ihre inverse Matrix gleich ihrer Adjungierten: $Q^{-1} = Q^H$. Indem wir das Gleichungssystem $A\vec{x} = \vec{b}$ auf beiden Seiten von links mit Q^H multiplizieren, erhalten wir

$$R\vec{x} = Q^H\vec{b}.$$

Mit der Notation $\vec{c} := Q^H\vec{b}$ ergibt sich

$$R\vec{x} = \begin{pmatrix} r_{11} & \cdots & r_{1n} \\ & \ddots & \vdots \\ 0 & & r_{nn} \end{pmatrix} \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix} = \begin{pmatrix} c_1 \\ \vdots \\ c_n \end{pmatrix},$$

also die n Gleichungen

$$\begin{aligned} r_{11}\,x_1 &= c_1 - r_{12}x_2 - \ldots - r_{1n}x_n \\ &\vdots \\ r_{jj}\,x_j &= c_j - r_{j,j+1}x_{j+1} - \ldots - r_{jn}x_n \\ &\vdots \\ r_{nn}\,x_n &= c_n. \end{aligned}$$

Falls die Matrix A invertierbar ist, sind alle $r_{jj} \neq 0$, wodurch die Gleichungen sich durch Iteration von unten Zeile für Zeile algorithmisch lösen lassen. Da wir jedoch durch unsere geometrische Konstruktion bereits wissen, dass das lineare Gleichungssystem (3) wenigstens eine Lösung besitzt, können wir im Fall $r_{jj} = 0$ den Koeffizienten x_j beliebig wählen, also liefert die folgende Vorgehensweise eine Lösung von (3):

$$\text{für } \; j = n, \ldots, 1 \; \text{ setze } \quad x_j := \begin{cases} \dfrac{1}{r_{jj}} \left(c_j - \displaystyle\sum_{i=j+1}^{n} r_{ji}\,x_i \right) & \text{falls } \; r_{jj} \neq 0, \\ 0 & \text{falls } \; r_{jj} = 0. \end{cases}$$

Die Lösung $\vec{x}$ liefert uns die gesuchten Koeffizienten $\beta_i := \overline{x_i}$. Die Orthogonalprojektion des Vektors $\tilde{u}$ auf U_B ist jetzt durch Gleichung (2) bestimmt, und die Projektionswahrscheinlichkeit ergibt sich durch Formel (1).

5 Berechnung von Skalarprodukten mit der Trapezregel

Wir verwenden hier eine ziemlich allgemeine Methode, die bei den Funktionen weder eine konstante noch eine gemeinsame Abtastrate voraussetzt. Zur Berechnung eines Skalarprodukts

$\langle u,v\rangle$ benötigen wir für jede der gegebenen Funktionen eine geordnete Liste von Argument-Wert-Paaren

$$(t_0,u(t_0)),\ldots,(t_k,u(t_k)) \quad \text{und} \quad (s_0,v(s_0)),\ldots,(s_\ell,v(s_\ell))$$

mit den Eigenschaften

$$0=t_0<\ldots<t_k=1,\ u(t_i)\in\mathbb{C} \quad \text{und} \quad 0=s_0<\ldots<s_\ell=1,\ v(s_i)\in\mathbb{C}.$$

Die Berechnung des Skalarprodukts $\langle u,v\rangle$ erfolgt in drei Schritten:

1. Bilde die Vereinigungsmenge der Abtastpunkte und ordne diese, das ergibt

   $$\{t_0,\ldots,t_k\}\cup\{s_0,\ldots,s_\ell\}=\{r_0,\ldots,r_m\} \qquad \text{mit} \quad 0=r_0<\ldots<r_m=1.$$

   Für die Gesamtzahl der Abtastpunkte gilt damit

   $$\max\{k,\ell\}\le m\le k+\ell-2.$$

2. Zur Bestimmung der Signalwerte an den zusätzlichen Abtastpunkten verwenden wir lineare Interpolation, also die Formeln

   $$u(r_i):=\begin{cases} u(t_j) & \text{falls } r_i=t_j,\\ u(t_j)+\dfrac{(r_i-t_j)(u(t_{j+1})-u(t_j))}{t_{j+1}-t_j} & \text{falls } t_j<r_i<t_{j+1}\end{cases} \quad \text{und}$$

   $$u(r_i):=\begin{cases} v(s_j) & \text{falls } r_i=s_j,\\ v(s_j)+\dfrac{(r_i-s_j)(v(s_{j+1})-v(s_j))}{s_{j+1}-s_j} & \text{falls } s_j<r_i<s_{j+1}.\end{cases}$$

3. Das Integral wird nun mit der Trapezregel approximiert:

   $$\begin{aligned}\langle u,v\rangle &= \int_0^T u(t)\,\overline{v(t)}\,dt\\ &\approx \sum_{i=0}^{m-1}(r_{i+1}-r_i)\frac{u(r_i)\overline{v(r_i)}+u(r_{i+1})\overline{v(r_{i+1})}}{2}\\ &= \frac{1}{2}\left(u(1)\overline{v(1)}+\sum_{i=0}^{m-1}\left(r_{i+1}\,u(r_i)\overline{v(r_i)}-r_i\,u(r_{i+1})\overline{v(r_{i+1})}\right)\right).\end{aligned}$$

6 Darstellung von Bedeutungen

Wir fixieren zunächst eine mögliche Bedeutung B und bezeichnen mit $a_1,\ldots,a_n$ die bereits gespeicherten Sprachsignale mit dieser Bedeutung. Weiter bezeichne $\tilde{a}$ das aktuell neu ankommende Signal.

Es bieten sich nun mehrere Möglichkeiten, die Sprachsignale in $L^2([0,T])$ darzustellen. Allgemein ist zunächst ein geeignetes T auszuwählen, um anschließend alle Signale auf diese Länge T zu bringen. Zwei Varianten sind dabei naheliegend.

1. Bezeichne s_i die Anzahl der Samples des Signals a_i und $\tilde{s}$ entsprechend die Sampleanzahl von $\tilde{a}$. Setze

   $$T=(\min(\{s_i\mid i=0,\ldots,n\}\cup\{\tilde{s}\}))^{-1}$$

   und schneide alle Signale nach T^{-1} Samples ab.

Tabelle 1 – Die verwendeten Audiodaten (44.1 kHz Samplingrate bei 16 bit Auflösung).

Datei	Samples	Äußerung	Datei	Samples	Äußerung
a_1	107520	`ich hole gleich oma ab`	a_7	98304	`ich hole gleich oma ab`
a_2	110592	`ich hole gleich oma ab`	a_8	107520	`ich hole gleich oma ab`
a_3	115712	`ich hole gleich oma ab`	a_9*	480000	*`Rauschen`*
a_4	121856	`ich hole gleich oma ab`	a_{10}	104448	`ich hole gleich oma ab`
a_5	104448	`ich hole gleich oma ab`	a_{11}*	108544	`ich brauch nen kaffee`
a_6	102400	`ich hole gleich oma ab`	a_{12}*	88064	`nein`

2. Um hingegen die vollständigen Signale zu verwenden, wähle T beliebig, setze beispielsweise $T = 1$. Bezeichne x_i die Menge der Abtastpunkte $\{{}^0/_{s_i}, \ldots, {}^{s_i}/_{s_i}\}$ für a_i und $\tilde{x}$ entsprechend $\{{}^0/_{\tilde{s}}, \ldots, {}^{\tilde{s}}/_{\tilde{s}}\}$. Bilde $x = \bigcup_i x_i \cup \tilde{x}$ und interpoliere alle Signale auf x.

Danach können weitere Verarbeitungsschritte folgen. Diese orientieren sich sinnvollerweise an der Physiologie des menschlichen Ohres. Wir verwenden hier die MFCC aus der Implementierung [11] für Python, die noch eine zusätzliche Vor- und Nachverarbeitung erlaubt.

In Tabelle 1 sind die Äußerungen aufgeführt, die wir für unsere Experimente benutzt haben. Die zusätzlichen Äußerungen a_{13}, a_{14} und a_{15} sind Wiederholungen von a_5, a_8 bzw. a_{10}. Die mit * gekennzeichneten Äußerungen sollen nicht der Bedeutung B zugeordnet werden.

7 Ergebnisse

Wir protokollieren zwei Experimente, die Orthogonalprojektion auf den Untervektorraum U_B und den Vergleich mit dem Durchschnitt der Mustervektoren, mit jeweils vier Durchläufen:

1. Zurückschneiden auf die minimale Länge (Spalten p_1, p_1'),
2. Reskalieren auf gemeinsame Stützstellen (Spalten p_2, p_2'),
3. Zurückschneiden mit anschließender MFCC (Spalten p_3, p_3') und
4. Zurückschneiden, MFCC mit Vor- und Nachverarbeitung (Spalten p_4, p_4').

In den Tabellen 2 und 3 ist das in der betreffenden Zeile ankommende Sprachsignal in der ersten Spalte notiert. Die Spalte n enthält die Anzahl der in der betreffenden Zeile verwendeten Äußerungen mit der Bedeutung B. Die Projektionswahrscheinlichkeiten auf U_B befinden sich in den Spalten p_i, diejenigen auf den Durchschnittsvektor in den Spalten p_i'. Eine hundertprozentige Projektionswahrscheinlichkeit ergibt sich genau dann, wenn der ankommende Vektor im jeweils betrachteten Untervektorraum liegt; im Falle der Durchschnittsbildung ist dieser nur eindimensional. Die Spalte Stützstellen enthält jeweils die Anzahl der verwendeten Stützstellen zur Beschreibung der transformierten Signale im Hilbertraum $L^2([0,T])$. Die vorletzte Zeile enthält die minimale Projektionswahrscheinlichkeit für eine Äußerung mit der Bedeutung B, die letzte die maximale Projektionswahrscheinlichkeit für einen Äußerung, die nicht die Bedeutung B hat. Eine sichere Trennung ist möglich, wenn „min Treffer“ > „max Fehler“ gilt.

Bei den Zeitsignalen in Tabelle 2 war zu erwarten, dass die reskalierten Zeitsignale schlechtere Ergebnisse liefern, da eine Reskalierung die Frequenzen ändert („Der Ton macht die Musik!“). Überraschenderweise scheint gerade hier eine Trennung durch unsere Methode möglich. Tabelle 2 zeigt, dass für die Untersuchung der Bedeutung Zeitsignale nicht ideal sind, andererseits kann man die Stabilität unseres Verfahrens erkennen.

Auch bei den MFCC-Signalen in Tabelle 3 ist zu sehen, dass die Orthogonalprojektion auf den Untervektorraum U_B bessere Ergebnisse liefert als der Vergleich mit dem Durchschnitt.

Tabelle 2 – Ergebnisse der Experimente 1 und 2 mit Zeitsignalen.

Signal	n	p_1	p'_1	Stützstellen	p_2	p'_2	Stützstellen
a_1	0			107520			107520
a_2	1	0.97%	0.97%	107520	0.77%	0.77%	218110
a_3	2	2.74%	0.29%	107520	0.65%	0.10%	333820
a_4	3	6.49%	1.47%	107520	1.96%	0.37%	455674
a_5	4	4.94%	8.01%	104448	0.32%	0.15%	560120
a_6	5	2.70%	0.38%	102400	1.08%	0.32%	662518
a_7	6	7.31%	1.45%	98304	0.62%	0.04%	760820
a_8	7	5.24%	5.10%	98304	0.55%	0.00%	760820
a_9*	8	10.34%	1.17%	98304	0.02%	0.00%	1720818
a_{10}	8	2.36%	9.87%	98304	0.53%	0.00%	760820
a_{11}*	8	7.89%	0.05%	98304	0.24%	0.01%	869360
a_{12}*	8	26.84%	15.19%	88064	0.23%	0.01%	848882
a_{13} (a_5)	8	100.00%	5.08%	98304	100.00%	18.50%	760820
a_{14} (a_8)	9	100.00%	2.75%	98304	100.00%	4.62%	760820
a_{15} (a_{10})	9	2.36%	1.73%	98304	0.53%	0.01%	760820
min Treffer		2.36%	1.73%		0.53%	0.00%	
max Fehler		26.84%	15.19%		0.24%	0.01%	

Außerdem ist zu sehen, dass die Trennschärfe durch eine zusätzliche Vor- und Nachverarbeitung der MFCC zunimmt, wovon vor allem der Vergleich mit dem Durchschnitt profitiert.

8 Zusammenfassung und Ausblick

Wir haben bisher nur eine einzelne Bedeutung untersucht und damit die Eignung des Verfahrens als Detektor demonstriert. Für den Einsatz innerhalb der Maschinensemiotik wäre jedoch ein Klassifikator erforderlich, was weitere Untersuchungen nötig macht.

Literatur

[1] KLIMCZAK, P., M. HUBER, P. BEIM GRABEN, und G. WIRSCHING: *Eine maschinensemiotische Petrinetz-Architektur für ein menschenzentriertes User-Interface*. In S. HILLMANN, B. WEISS, T. MICHAEL, und S. MÖLLER (Hrsg.), *Proceedings of "Elektronische Sprachsignalverarbeitung (ESSV)"*, Bd. 99 d. Reihe *Studientexte zur Sprachkommunikation*, S. 254–263. TUDpress, Dresden, 2021.

[2] BEIM GRABEN, P., M. HUBER-LIEBL, P. KLIMCZAK, und G. WIRSCHING: *Machine semiotics*. *ArXiv*, abs/2008.10522v2, 2023. URL `https://arxiv.org/abs/2008.10522v2`. `2008.10522v2`.

[3] BAEVSKI, A., H. ZHOU, A. MOHAMED, und M. AULI: *wav2vec 2.0: A framework for self-supervised learning of speech representations*. *ArXiv*, abs/2006.11477v3, 2020. doi:10.48550/arXiv.2006.11477. URL `https://arxiv.org/abs/2006.11477v3`. `2006.11477v3`.

[4] WIRSCHING, G., M. WOLFF, und I. SCHMITT: *Quantenlogik. Eine Einführung für In-*

Tabelle 3 – Ergebnisse der Experimente 3 und 4 mit MFCC-Signalen.

Signal	n	p_3	p'_3	Stützstellen	p_4	p'_4	Stützstellen
a_1	0			3159			3159
a_2	1	88.65%	88.65%	3159	79.27%	79.27%	3159
a_3	2	91.42%	89.15%	3159	79.48%	79.43%	3159
a_4	3	87.11%	84.20%	3159	68.15%	65.78%	3159
a_5	4	88.12%	84.41%	3068	65.24%	59.96%	3068
a_6	5	92.69%	83.77%	3003	74.08%	66.56%	3003
a_7	6	78.64%	65.94%	2886	66.49%	55.65%	2886
a_8	7	87.84%	81.56%	2886	56.87%	50.59%	2886
a_9*	8	17.77%	2.60%	2886	2.01%	0.27%	2886
a_{10}	8	92.58%	87.37%	2886	60.99%	56.67%	2886
a_{11}*	8	84.87%	84.23%	2886	28.99%	27.18%	2886
a_{12}*	8	43.96%	25.50%	2587	19.39%	16.43%	2587
a_{13} (a_5)	8	100.00%	92.14%	2886	100.00%	75.36%	2886
a_{14} (a_8)	9	100.00%	84.36%	2886	100.00%	57.25%	2886
a_{15} (a_{10})	9	92.58%	88.64%	2886	60.99%	58.15%	2886
min Treffer		92.58%	84.36%		60.99%	56.67%	
max Fehler		84.87%	84.23%		28.99%	27.18%	

genieure und Informatiker. Springer Berlin Heidelberg, 2023. doi:10.1007/978-3-662-66780-4. URL http://dx.doi.org/10.1007/978-3-662-66780-4.

[5] PIRON, C.: *Axiomatique quantique*. *Helvetica Physica Acta*, S. 439–468, 1964.

[6] SOLÈR, M. P.: *Charakterisierung von Hilberträumen als spezielle orthomodulare Räume*. Dissertation. Universität Zürich, 1994.

[7] PICONE, J.: *Signal modeling techniques in speech recognition*. *Proceedings of the IEEE*, 81(9), S. 1215–1247, 1993. doi:10.1109/5.237532. URL http://dx.doi.org/10.1109/5.237532.

[8] KAHNEMANN, D.: *Thinking, fast and slow*. Farrar, Straus and Giroux, New York, 2011.

[9] VAN ROSSUM, G. und F. L. DRAKE: *The Python Language Reference Manual*. Python-Labs, Virginia, USA, 2024. URL http://www.python.org/.

[10] HARRIS, C. R., K. J. MILLMAN, S. J. VAN DER WALT, R. GOMMERS, P. VIRTANEN, D. COURNAPEAU, E. WIESER, J. TAYLOR, S. BERG, N. J. SMITH, R. KERN, M. PICUS, S. HOYER, M. H. VAN KERKWIJK, M. BRETT, A. HALDANE, J. F. DEL RÍO, M. WIEBE, P. PETERSON, P. GÉRARD-MARCHANT, K. SHEPPARD, T. REDDY, W. WECKESSER, H. ABBASI, C. GOHLKE, und T. E. OLIPHANT: *Array programming with NumPy*. *Nature*, 585(7825), S. 357–362, 2020. doi:10.1038/s41586-020-2649-2. URL https://doi.org/10.1038/s41586-020-2649-2.

[11] LYONS, J., D. Y.-B. WANG, GIANLUCA, H. SHTEINGART, E. MAVRINAC, Y. GAURKAR, W. WATCHARAWISETKUL, S. BIRCH, L. ZHIHE, J. HÖLZL, J. LESINSKIS, H. ALMÉR, C. LORD, und A. STARK: *jameslyons/python_speech_features: release v0.6.1*. 2020. doi:10.5281/ZENODO.3607820. URL https://zenodo.org/record/3607820.