TECHNISCHE UNIVERSITÄT
TUHH
HAMBURG-HARBURG

Arbeitsbereich Technische Informatik VI

Prof. Dr. F. Mayer–Lindenberg

# Modifikation der neuronalen *ART–2* Architektur

Christian Müller–Tomfelde

9. Juli 1993

# Einleitung

Die vorliegende Arbeit beschäftigt sich mit der von Stephen Grossberg 1976 vorgestellten *Adaptive Resonance Theorie*. Die zentrale Eigenschaft von neuronalen Netzen, die auf diese Theorie aufbauen, ist die des selbständigen, stabilen Lernens.

Neuronalen Netzen mit der Eigenschaft der Selbstorganisation dürfen Lernphase und Klassifizierungsphase nicht starr vorgegeben sein. Das Netz muß vielmehr aufgrund vorangegangener Musterpräsentationen jederzeit selbst entscheiden, ob das anstehende Muster ein schon bekanntes ist und einer Klasse zugeordnet werden kann (Klassifizierungsphase) oder ob es ein unbekanntes ist und eine neue Klasse für dieses eröffnet werden muß (Lernphase).

Die Übertragung dieser Fähigkeit auf neuronale Netze mündet, wie z.B. bei der *Kohonen Feature Map*, in einen Dualismus zwischen Stabilität und Plastizität der Klassifizierung. Mit dem *ART–1*-Modell hat Grossberg in [2] 1987 ein neuronales Netz vorgestellt, das diesen Dualismus aufhebt.

In der folgenden Arbeit soll versucht werden das *ART*-Modell um die Eigenschaft einer verteilten Klassifizierung, im Sinne einer orthogonalen Zerlegung eines Musters zu erweitern.

# Inhaltsverzeichnis

# Kapitel 1

## Adaptive Resonanztheorie

Die Arbeiten von Carpenter und Grossberg zeichnen sich durch einen deutlichen Bezug auf die mathematische Modellierung dynamischer elektrochemischer Prozesse bei der Informationsübertragung zwischen Neuronen aus.

Dem *ART–1*-Modell liegen zwei Differentialgleichungen zugrunde, die entscheident für sein dynamisches Verhalten sind.

$$\frac{dx_i}{dt} = -Ax_i + (1 - Bx_i)J_i^r - (C + Dx_i)J_i^h \tag{1.1}$$

Die Gl.1.1 ist die Membrangleichung. Sie beschreibt das Verhalten des Potentials bzw. der Aktivität einer Nervenzelle bei Reizungen ($J^r$) und Hemmungen ($J^h$).

$$\frac{dz_{ij}}{dt} = d\,(S_i - z_{ij})x_j \tag{1.2}$$

Gl.1.2 ist eine Nicht-*Hebb'sche* Lernregel, die mit der postsynaptische Aktivität $x_j$ und mit dem präsynaptische Signal $S_i$ über die differentielle Veränderung des Gewichts $z_{ij}$ (Größe der Synapsenmembran) entscheidet. Der Faktor $d$ stellt die Lernrate dar.

Für die zeitlichen Einordung der Gl. 1.1 und Gl. 1.2 muß immer berücksichtig bleiben, daß sich die Aktivität eines Neuron schneller ändert, als Größe der Synapsenfläche.

### 1.1 *ART–1* Struktur

Die allgemeine Struktur des *ART–1* Modells (siehe Abbildung 1.1) besteht aus zwei Neuronenfeldern (*F2* und *F1* ) sogenannte *STM* Felder (*short term memory* ).

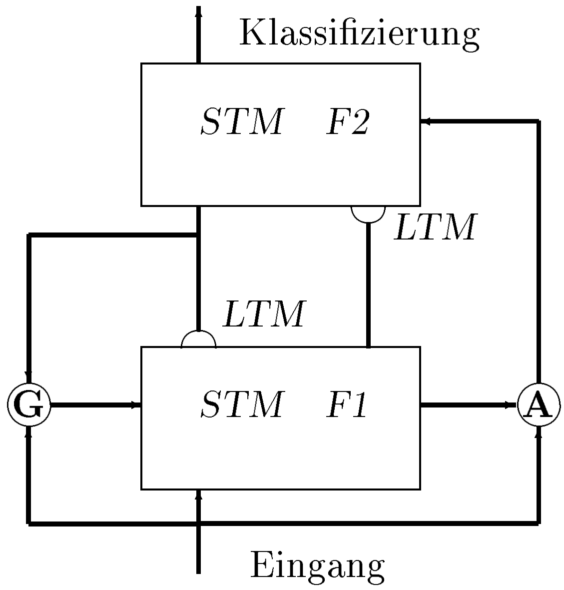


Abbildung 1.1: *ART–1*

Zwischen den Felder befinden sich gewichtete Signalpfade, die in ihrer Anordnung adaptiven Filtern ähneln (*long term memory* ). Zusätzlich besitzt das *ART–1* Modell noch eine sogenannte Aktivitätskontrolle und Aufmerksamkeitskontrolle. Die Abbildung 1.1 spiegelt das Funktionsschema wider.

**Adaptive Filter :** Von jedem Neuron aus *F1* führt ein gewichteter Signalpfad zu jedem Neuron aus *F2* und umgekehrt (siehe *LTM* in Abbildung 1.1).

*F2* **:** Neuronenfeld mit Konkurrenzeigenschaften. Jedes Neuron in diesem Feld stellt eine Klasse dar. Für eine Eingangsmuster ist dann eine Klasse gefunden, wenn das zugehörige Neuron signifikante Aktivität besitzt.

*F1* **:** Neuronenfeld mit Konkurrenz- und Vergleichseigenschaften. An diesem Feld liegen zwei Vektoren an: der Eingangsmustervektor und ein Vektor, der eine Erwartung (*top down template*) des Systems auf das Eingangsmuster darstellt. Auf die Vergleichseigenschaft wirkt die Aktivitätskontrolle. Die Aktivität des Feldes wirkt auf die Aufmerksamkeitskontrolle.

Die in Abbildung 1.1 dargestellten Kontrollen, wie Aktivitäts - und Aufmerksamkeitskontrolle sind Steuereinheiten, die den Lauf der Klassifizierung beeinflussen, ohne die das Netzwerk dem Aufbau einer *Kohonen Feature Map* ähnelt.

**Aktivitätskontrolle:** Mit einem Signal aus dem Eingangsmustervektor und dem Zustand der Erwartung wird das Verhalten des Feldes *F1* gesteuert (siehe **G** in Abbildung 1.1).

**Aufmerksamkeitskontrolle:** Entscheidet über die Güte der Resonanz, d.h über die Ähnlichkeit zwischen Erwartungsmuster und Eingabemuster. Mit einem Parameter kann hier die Aufmerksamkeit des Systems eingestellt werden (siehe **A** in Abbildung 1.1).

Die Unterscheidung zwischen *short term memory* und *long term memory* liegt in der zeitlichen Einordnung, der zugrunde liegenden Differentialgleichungen 1.1 und 1.2 begründet. Während sich die Aktivitäten $x_i$ in den Feldern *F1* und *F2* ändern können, bleiben die Gewichte $z_{ij}$ in den Signalpfaden der *LTM* 's konstant.

## 1.2 Algorithmus

Zur allgemeinen Beschreibung des Algorithmus werden Konventionen getroffen, mit denen das *ART–1* System im weiteren prinzipiell beschrieben wird.

| Variable | Beschreibung |
|---:|---|
| $M$ | Anzahl der Neuronen in Feld *F1* |
| $N$ | Anzahl der etablierten Neuronen im Feld *F2* |
| $I$ | Eingangsmuster |
| $X^{(1)}$ | Aktivität des Feldes *F1* |
| $Y^{(1)}$ | Ausgangsvektor des *F1* Feldes |
| $X^{(2)}$ | Aktivität des Feldes *F2* |
| $Y^{(2)}$ | Ausgangsvektor des *F2* Feldes |
| $z_{ij}$ | Gewicht des Signalpfades von $Y_i^{(1)}$ zu $X_j^{(2)}$ |
| $z_{ji}$ | Gewicht des Signalpfades von $Y_j^{(2)}$ zu $X_i^{(1)}$ |
| $E^{(J)}$ | Erwartungsmuster vom Neuron J aus *F2* an *F1* |
| $R$ | Register der Verfügbarkeit von *F2* Neuronen (Ausschluß) |

Bei folgender Darstellung der Abarbeitungsvorschrift eines *ART–1* Systems wird auf die detaillierte Beschreibung verzichtet und auf [2] und [5] verwiesen. Diese Beschreibung gilt für das *ART–1* und *ART–2* Modell.

1. Es liegt kein Vektor $I$ an *F1* an, bzw. $I = \mathbf{0}$. Die Aktivitätskontrolle setzt $X^{(1)}$ und $X^{(2)}$ zu $\mathbf{0}$. In $R$ werden alle Neuronen $N$ des Feldes *F2* als verfügbar erklärt. Ein neuer Eingangsvektor I wird an das System gelegt.

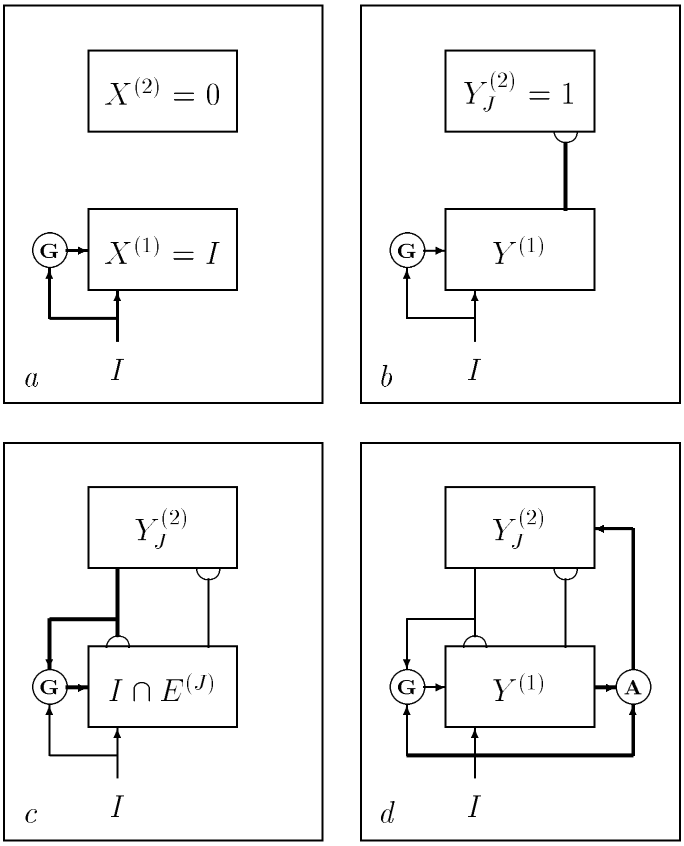


Abbildung 1.2: *Zustände im ART-System*

2. Da kein Neuron in *F2* aktiv ist, schaltet die Aktivitätskontrolle das *F1* Feld so, daß $Y^{(1)} = X^{(1)} = I$ wird (Abbildung 1.2 a).

3. Über die adaptiven Filtergewichte $z_{ij}$ werden die Aktivitäten in *F2* bzw. der Vektor $X^{(2)}$ bestimmt. Unter allen verfügbaren Neuronen aus *F2* bekommt nur dasjenige mit maximaler Aktivität den Ausganswert 1, $Y_J^{(2)} = 1$ (Abbildung 1.2 b).

4. Über die adaptiven Filtergewichte $z_{ji}$ wird die Erwartung $E^{(J)}$ an *F1* gelegt. Das System reagiert auf ein Eingangsmuster mit einer Erwartung. Die Aktivitätskontrolle schaltet nun die Vergleichseigenschaft von *F1* ein, da sowohl ein Eingangsmuster $I$ als auch eine Erwartung an *F1* anliegt. Es wird $Y^{(1)} = I \cap E^{(J)}$ gebildet (Abbildung 1.2 c).

5. Mit den Normen $\| Y^{(1)} \|$ und $\| I \|$ und einem Aufmerksamkeitsparameter $\rho$ wird über die Güte der Resonanz entschieden (Abbildung 1.2 d). Wird die Klassifizierung als ungültig erklärt, wird das aktive Neuron $J$ aus *F2* inaktiv, in $R$ als nicht mehr verfügbar erklärt, d.h. ausgeschlossen und mit Schritt 2 die Suche nach geeigneter Klassifizierung fortgesetzt.

6. Die Gewichte der adaptiven Filter werden gemäß der Lernregel (Gl.1.2) neu bestimmt d.h. integriert.
7. Das Eingangsmuster ist klassifiziert am aktiven Neuron $J$ aus $F2$ , das Muster wird vom System abgekoppelt. Der nächste Schritt ist Schritt 1.

Grossberg unterscheidet i.A. zwischen *schnellem* und *langsamem* Lernen. Diese Einteilung erfolgt durch die unterschiedliche Bewertung der Zeitbasis $\Delta t$ beim Lösen der Differentialgleichung (Lerngleichung 1.2).

**Schnelles Lernen :** Die numerische Integration der Lerngleichung erfolgt bis zu einer definierten Konvergenzschranke. Die Gewichte befinden sich danach in einem Zustand des momentanen Gleichgewichts.

**Langsames Lernen :** Die Lerngleichung wird nur eine fest Schrittweite $\Delta t$ integriert und kann von ihrer Konvergenz noch weit entfernt sein.

Bei der Integration der unterschiedlichen Differentialgleichungen für *short term memory* und *long term memory* muß lediglich darauf geachtet werden, daß die zeitliche Einstufung erhalten bleibt; Die *STM* -Gleichungen werden schneller verarbeitet als *LTM* -Gleichungen, d.h. die Veränderung der Gewichte $z_{ij}$ und $z_{ji}$ ist verhältnismäßig langsam gegenüber der Veränderung der Aktivitäten $X^{(1)}$ und $X^{(2)}$.

## 1.3 Grundlagen des Klassifizierungsverhaltens

Nach Grossberg ist die Existenz und das Zusammenspiel von drei Eigenschaften des *ART*-Systems zentral, um die Anforderung des stabilen, selbstständigen Klassifizierens zu erfüllen. Mit mehreren Theoremen wurde in [2] das Klassifizierungsverhalten bewiesen. Von den drei folgenden Gesetzen, ist das letzte Ausgangspunkt für die weiteren Betrachtungen.

- Webergesetz
- nicht- *Hebb'sche* Lernregel
- nichtlineares Vergleichsgesetz (*2/3 Regel*)

Ein Nicht-Beachten dieser Vorschriften führt i.A. zu nicht-stabilem Klassifizierungsverhalten. Im einzelnen stellt das *Webergesetz* eine Normierung der adaptiven Filtergewichte dar, um nach der Stabilisierung des Systems eine direkte Klassifizierung zu ermöglichen, d.h. das zuerst aktive Neuron von *F2* ist eine gültige Klasse (keine Suche). Mit der nicht-*Hebb'schen* Lernregel nach Gl. 1.2 wird im System die Dynamik der sich schnell ändernden Neuronenaktivität verbunden mit der Dynamik der trägen Gewichtsadaption, was die Voraussetztung zu Selbstorganisation darstellt. Diese beiden Vorschriften sind in vielen neuronalen Netzen mit der Eigenschaft der Selbstorganisation zu finden (z.B. *Counterpropagation* Netzwerke , *Kohonens Feature Map*).

Das nichtlineare Vergleichsgesetz ist in der oben beschriebenen Struktur im Feld *F1* realisiert. Das Eingangsmuster $I$ und der Ausgangsvektor $Y^{(2)}$ von *F2* bestimmten das Verhalten des Feldes *F1* .

$$Y^{(1)} = \begin{cases} \mathbf{0} & : \; I = \mathbf{0} \quad \text{und} \quad Y^{(2)} = \mathbf{0} \\ I & : \; I \neq \mathbf{0} \quad \text{und} \quad Y^{(2)} = \mathbf{0} \\ I \cap E^{(J)} & : \; I \neq \mathbf{0} \quad \text{und} \quad Y^{(2)} \neq \mathbf{0} \end{cases} \tag{1.3}$$

Der Name *2/3 Regel* kommt daher, daß die Aktivität $X_i^{(1)}$ eines Neurons $i$ nur dann einen Ausgang $Y_i^{(1)} > 0$ erhält, wenn 2 von drei Eingänge an das Neuronenfeld aktiv sind. Zu diesen Eingängen gehörten die Vektoren $I$, $E^{(J)}$ und das Ausgangssignal der Aktivitätskontrolle. Während $X_i^{(1)}$ und $E_i^{(J)}$ die Reizungen $J_i^r$ bedeuten, steuert die Aktivitätskontrolle eine nichtspezifische Hemmung $J^h$ auf alle Neuron des Feldes (Gl. 1.1).

## 1.4 Mathematische Beschreibung der Klassifizierung

Für die weitere Beschreibung wird ein Muster $I$ als ein Vektor aus dem binären Raum $B^M$ definiert.

$$I = \{I_1, I_2, \ldots, I_m, \ldots, I_M\} \quad \text{mit} \quad I_m \in \{0, 1\} \tag{1.4}$$

Die Bildung Norm eines Vektors $I$ erklärt sich aus der zugrundeliegenden Metrik des binären Raums (Hamming-Distanz). Sind $x$,$y$ binäre $M$-Tupel gilt :

$$\begin{aligned} d_{Ham}(x, y) &= \sum_{i=1}^{M} (\overline{x_i} \wedge y_i) \vee (x_i \wedge \overline{y_i}) \\ \|x\| &= d_{Ham}(x, \mathbf{0}) = \sum_{i=1}^{M} x_i \end{aligned} \tag{1.5}$$

Weiterhin wird ein Muster als Element einer Menge

$$I \in F_K \quad \text{mit} \quad F_K = \{I^{(1)}, I^{(2)}, \ldots, I^{(k)}, \ldots, I^{(K)}\} \tag{1.6}$$

definiert. $F_K$ ist ein Vorrat von $K$ Mustern. Interpretiert man die sequentielle Präsentation der Muster als zeitdiskrete Abfolge von Musterwerten einer Musterquelle,

$$f_K(n\,T_P) = f(n) = I^{(k_n)} \quad \text{mit} \quad n = 0, 1, \ldots \tag{1.7}$$

kann nach der Stabilisierungsphase das System als *eingeschwungen* erklärt werden, in Analogie zu systemtheoretischen Nomenklatur dynamischer Systeme. Die Größe $T_P$ stellt in Gleichung 1.7 die Präsentationsdauer eines Musterwertes dar.

Um nun verschiedene Systemverhaltensweisen miteinander vergleichen zu können, muß die Musterquelle $F_K$ das System immer identisch, d.h. in der gleichen Musterabfolge, erregen.

Für die Beschreibung in diesem Abschnitt, wird die Klassifizierung im *eingeschwungenen* Zustand betrachtet. Die stabile Zuordnung eines Musters zu einer Klasse entspricht dann einer Überführung von

$$F_K \to G_N \quad \text{mit} \quad G_N = \{E^{(1)}, E^{(2)}, \ldots, E^{(j)}, \ldots, E^{(N)}\} \tag{1.8}$$

Ein Erwartungsvektor $E^{(j)}$ ist ebenso ein $M$-Tupel aus $B^M$. Ein einzelnes Muster $I^{(k)}$ wird mit $E^{(j)}$ klassifiziert. d.h

$$P : F_K \longrightarrow G_N \; ; \; k \longmapsto j \tag{1.9}$$

Diese Relation $P$ ist eindeutig, aber nicht umkehrbar. Alle erzeugten Paar $(k, j)$ der Abbildung genügen einer Aufmerksamkeitsbedingung, die mit $\rho \in [0, 1]$ lautet:

$$\frac{\| I^{(k)} \cap E^{(j)} \|}{\| I^{(k)} \|} > \rho \tag{1.10}$$

Die Klassifizierung eines *ART*-Systems im *eingeschwungenen* Zustand ist durch drei Eigenschaften charkterisiert:

1. Vergleicht man die Klassifikationen zweier *ART*-Systeme, die verschieden Aufmerksamkeitswerte besitzen, wie z.B. $\rho_1$ und $\rho_2$ so gilt:

   $$\rho_1 < \rho_2 \quad \to \quad N_{\rho_1} < N_{\rho_2} \tag{1.11}$$

   Dieser Sachverhalt spiegelt die Eigenschaft, der parametrisch einstellbaren Größe von $G_N$ wider. Für die Mengen der Erwartungen gilt, daß $G_{N_{\rho_1}}$ keine wahre Teilmenge von $G_{N_{\rho_2}}$ sein muß.

2. An der oberen Grenze der Aufmerksamkeit folgt mit

$$\rho = 1 \quad \rightarrow \quad N_{\rho=1} = K \tag{1.12}$$

d.h jedes Muster erhält eine eigene Klasse und $F_K \equiv G_N$ (triviale Klassifikation).

3. Für die minimale Aufmerksamkeit wird

$$\rho = 0 \quad \rightarrow \quad N_{\rho=0} \geq 1 \tag{1.13}$$

$$E^{(1)} = \bigcap_{\nu=1}^{K} I^{(\nu)} \tag{1.14}$$

Ist $E^{(1)} = \mathbf{0}$ existieren orthogonale Musterpaare $(\nu, \mu)$ mit $I^{(\mu)} \perp I^{(\nu)}$ in der Musterquelle $F_K$. Dann müssen die Muster aus $F_K$ herausgenommen werden, die $E^{(1)} = \mathbf{0}$ nach Gl. 1.14 bewirken. Mit den isolierten Mustern wird nun eine neue Klasse gegründet und wiederum auf orthogonale Muster untersucht, wenn nicht direkt $E^{(N)} \neq \mathbf{0}$ ist. Für $N_{\rho=0} = 1$ stellt $E^{(1)}$ die Grundschnittmenge aller Muster, für $N_{\rho=0} > 1$ die der meisten Muster dar. Für $N_{\rho=0} > 1$ sind alle etablierten Erwartungen zueinander orthogonal.

Mit den Gl. 1.11,1.12 und 1.13 kann für eine Klassifizierung mit $\rho$ der Bereich angegeben werden, in dem die Anzahl der etablierten Klassen liegt

$$1 \leq N_\rho \leq K \tag{1.15}$$

Diese Aussage gilt allgemein, während Gl. 1.11 immer nur für eine bestimmte Folge der Präsentation von Mustern gilt.

In der Simulation (Abbildung 1.3) ist $\rho_1 = 0$ und es gilt $\rho_1 < \rho_2 < \rho_3$. Die binären Muster $I^{(k)}$ sind Vektoren mit 25 Elementen und bilden zusammen die Musterquelle $F_5$. Die Abfolge der Präsentation ist in der linken Spalte notiert. Die anderen Spalten zeigen die Erwartungen des Systems bei den verschiedenen Aufmerksamkeiten. Die Abbildung 1.4 zeigt die Zuordnung $P$ des Mustervorrats $F_5$ auf die verschieden großen Erwartungsmengen $G_{N_\rho}$.

- Nach spätestens 7 Präsentationen sind alle Klassifizierungen *eingeschwungen* (siehe $\star$ in Abbildung 1.3). Alle nachfolgenden Präsentationen werden ohne Suche direkt klassifiziert.

- Mit wachsendem $\rho$ wird Anzahl der etablierten Neuronen in *F2* größer, d.h. es gilt $N_{\rho_1} < N_{\rho_2} < N_{\rho_3}$. Durch Hinzufügen der Erwartung $E^{(3)}_{\rho_2}$ läßt sich $G_{N_{\rho_2}}$ aus $G_{N_{\rho_1}}$ erzeugen. Dies gilt jedoch nicht allgemein, wie man mit $G_{N_{\rho_2}}$ und $G_{N_{\rho_3}}$ sehen kann ($G_{N_{\rho_2}} \not\subset G_{N_{\rho_3}}$) (siehe auch Abbildung 1.4).

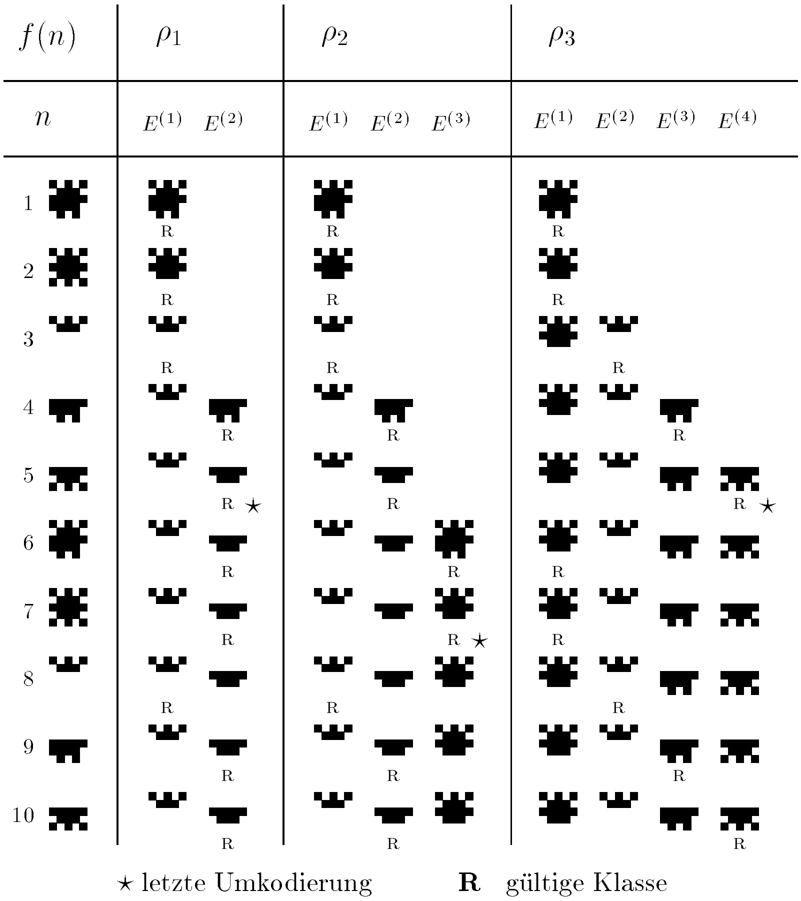


Abbildung 1.3: *Klassifizierungsverhalten*

- Für $\rho_1$ bilden sich zwei Klassen heraus, da das Muster $I^{(3)}$ nach Gl. 1.14 $E^{(1)}$ zu $\mathbf{0}$ zwingt, d.h zur Grundschnittmenge orthogonal ist. $I^{(3)}$ bildet somit eine eigene Klasse.

- Mit einer geeigneten Wahl von $\rho > \rho_3$ würde sich der triviale Fall nach Gl. 1.12 einstellen.

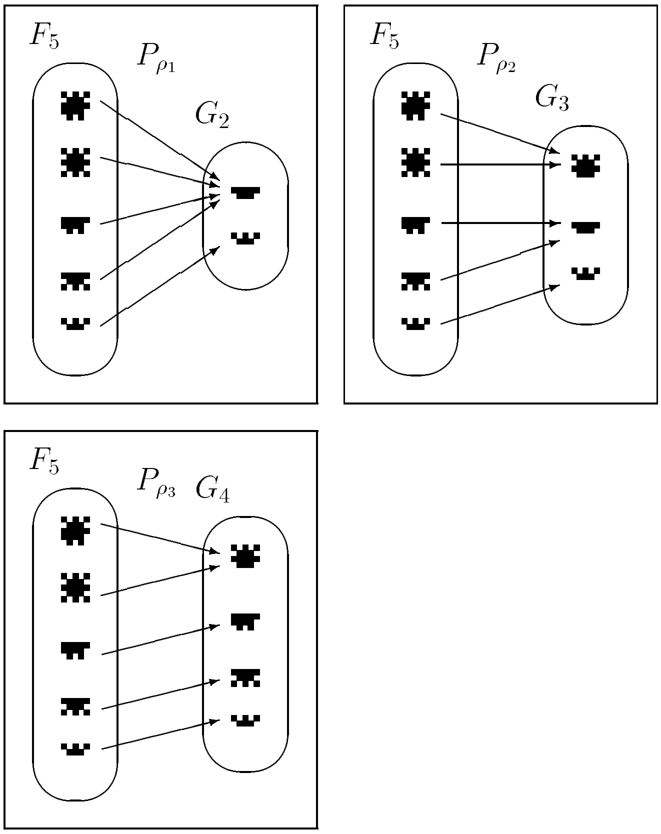


Abbildung 1.4: *Abbildung $P$ mit verschiedenem $\rho$*

# Kapitel 2

# Modifikation

Eine nähere Betrachtung der Musterquelle in Abbildung 1.3 macht deutlich, daß sich alle Muster $I \in F_5$ als Linearkombination dreier Musterkomponenten darstellen lassen (siehe Abbildung 2.1). Wie im vorhergehenden Kapitel beschrieben,

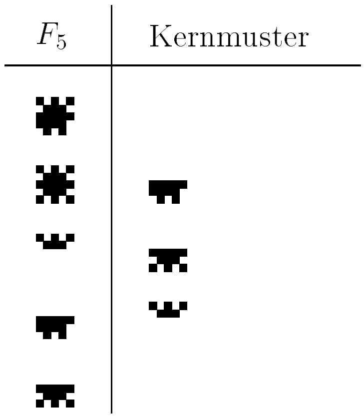


Abbildung 2.1: *Komponenten der Musterquelle*

vollzieht das *ART*-System aber eine Wahlklassifizierung, d.h. in *F2* ist immer nur ein Neuron aktiv, und ein Muster wird nur einer Klasse zugeordnet. Eine Erwartungsbildung, wie sie in Abbildung 2.1 mit den Kernmustern dargestellt ist, würde aber eine verteilte Klassifizierung voraussetzen, bei der mehrere Neuronen zu einem Muster aktiv sind. Eine Veränderung des Klassifizierungsverhalten des *ART*-Systems, die die Eigenschaft des stabilen, selbstorganisierten Lernens nicht einschränkt und außerdem eine teilweise orthogonale Zerlegung der Muster ermöglicht, wird im Folgenden beschrieben.

## 2.1 Erweiterung der *ART–1* Struktur

Werden zwei Muster $I^{(a)}$ und $I^{(b)}$ auf einem Neuron von *F2* klassifiziert, bildet sich eine Erwartung

$$E^{(J)} = I^{(a)} \cap I^{(b)} \tag{2.1}$$

als Schnittmenge der beiden Muster aus. Als Konsequenz der *2/3 Regel* verarmen die Merkmale, die nicht gleichzeitig in beiden Mustern vorhanden sind. Somit besitzt die Erwartung immer eine geringere oder gleiche Anzahl von Merkmalen als das kleinere der beiden klassifizierten Muster.

$$\| E^{(J)} \| \leq \min\{\| I^{(a)} \|, \| I^{(b)} \|\} \tag{2.2}$$

Werden nach der Bildung der Schnittmenge Gl. 2.1 die Muster $I^{(a)}$ und $I^{(b)}$ dem System nochmals vorgeführt, findet keine weitere Umkodierung der Gewichte statt, unter Voraussetzung der Gültigkeit von Gl. 1.10.

Für den besonderen Fall, daß für zwei Mustervektoren $I^{(a)}$ und $I^{(b)}$ einer Musterquelle

$$I^{(a)} \subset I^{(b)} \quad \text{mit} \quad I^{(a)} = f(n-1)\ ,\ I^{(b)} = f(n) \tag{2.3}$$

gilt, wird der Klassifizierungsverlauf im *ART–1*-Modell nun um zwei Verhalten erweitert. Als Grundlage und Voraussetzung der Erweiterung muß das *ART*-Modell eine Aussage über das Obermengenverhältnis von Erwartung und Eingangsmuster machen.

1. Wurde $I^{(a)}$ vom System angelernt, ist also $E^{(j_1)} = f(n-1) = I^{(a)}$. Nachfolgend wird dem System $I^{(b)} = f(n)$ präsentiert. Bei der Suche nach geeigneter Klassifizierung wird das Neuron $j_1$ aus *F2* mit der Erwartung $E^{(j_1)}$ d.h $I^{(a)}$ gewählt, welche eine Teilmenge von $I^{(b)} = f(n)$ ist. Mit dieser Information setzt das System für Gl. 1.10 eine geringere als die gewöhnliche Aufmerksamkeit zugrunde ($\rho_{ext} < \rho_{std}$), um über die Klassenzugehörigkeit zu entscheiden.

$$\frac{\| I^{(b)} \cap E^{(j_1)} \|}{\| I^{(b)} \|} = \frac{\| E^{(j_1)} \|}{\| I^{(b)} \|} = \frac{\| f(n-1) \|}{\| f(n) \|} > \rho(n) = \rho_{ext} \tag{2.4}$$

   Die Aufmerksamkeit $\rho$ ist also nicht mehr über die Zeit konstant, sondern schwangt situationsabhängig zwischen $\rho_{std}$ und $\rho_{ext}$ , je nachdem ob das Eingangsmuster eine Obermenge zur Erwartung darstellt oder nicht:

$$\rho(n) = \begin{cases} \rho_{std} & : \quad E^{(J)} \not\subseteq f(n) \\ \rho_{ext} & : \quad E^{(J)} \subseteq f(n) \end{cases} \tag{2.5}$$

2. Ist $E^{(j_1)} = f(n-1) = I^{(a)}$, wird $I^{(b)} = f(n)$ präsentiert und ist die Klassifizierung nach Gl. 2.4 gültig, so wird dem System noch ein Muster vorgehalten, das die Merkmale besitzt, die zusätzlich in $I^{(b)}$ gegenüber $E^{(j_1)}$ existieren.

$$\overline{E^{(j_1)}} \cap I^{(b)} = \overline{f(n-1)} \cap f(n) \tag{2.6}$$

Mit dem Muster nach Gl. 2.6 wird eine zusätzliche Erwartung $E^{(j_2)}$ etabliert. Wird $I^{(b)}$ nochmals präsentiert ist $I^{(b)}$ vollständig durch $E^{(j_1)}$ und $E^{(j_2)}$ klassifizierbar.

## 2.2 Mathematische Darstellung

Mit dem erweiterten *ART*-Modell, wie oben beschrieben, werden jetzt nicht mehr Zuordnungspaare $(k, j)$ mit $I^{(k)} \in F_K$ und $E^{(j)} \in G_N$ gebildet, sondern es wird einem Muster $I^{(k)}$ eine geordnete Reihe aus $N$ zugeordnet. Ist das System *eingeschwungen* gilt die Relation:

$$P_{ext} : F_K \longrightarrow G_N \; ; \; k \longmapsto \{j_1, j_2, \ldots, j_{n_k}\} = g(k) \tag{2.7}$$

Die geordnete Reihe $g(k)$ ist eine Folge von $n_k$, nacheinander aktivierten Neuronen aus *F2* . Für die aus $g(k)$ folgende geordnete Menge $G(k)$ gilt:

$$G(k) \; = \{E^{(j)} \mid j \in g(k)\}$$

$$G(k) \; \subseteq \; G_N \quad \text{für} \quad k \in \{0, 1, \ldots, K\} \tag{2.8}$$

Die Eigenschaften der Abbildung $P_{ext}$ lassen sich durch die folgenden Gleichungen beschreiben.

1. Ein Muster $I^{(k)}$ wird durch die Superposition aller Erwartungen der aktivierten Neuronen aus *F2* vollständig rekonstruiert.

$$I^{(k)} = \bigcup_{j \in g(k)} E^{(j)} \tag{2.9}$$

2. Alle Erwartungen zur Rekonstruktion eines Musters sind zueinander orthogonal:

$$E^{(j_a)} \cap E^{(j_b)} = \mathbf{0} \quad \text{für} \quad j_a, j_b \in g(k) \;\wedge\; a \neq b \tag{2.10}$$

3. Die Reihenfolge der Aktivierung der Neuronen aus *F2* entspricht der Größe der zugehörigen Erwartungen:

$$\| E^{(j_a)} \| \geq \| E^{(j_b)} \| \quad \text{für} \quad j_a, j_b \in g(k) \;\wedge\; a < b \tag{2.11}$$

Wie auch für das Standard-*ART–1*-Modell lassen sich auch für das Erweiterungsmodell Grenzen der Klassifizierung in Abhängigkeit der Aufmerksamkeitsparameter angeben.

1. An der oberen Grenze handelt es sich um die triviale Klassifizierung (siehe Gl. 1.12 Seite 8), mit $F_K \equiv G_N$ d.h. jedes Muster ist vollständig auf einem Neuron klassifiziert :

$$\rho_{std} = 1 \ , \ \rho_{ext} = 1 \quad \rightarrow \quad N = K \tag{2.12}$$

2. Für minimale Aufmerksamkeit wird $F_K$ in alle orthogonalen Teilmuster auf $G_N$ abgebildet. Ähnlich wie mit Gl. 1.14 auf Seite 8 werden Grundschnittmengen der Muster als Erwartungen etabliert. Zusätzlich jedoch noch alle zu diesen Grundschnittmengen orthogonalen Anteile der Muster. $G_N$ ist eine orthogonale Basis von $F_K$.

$$\rho_{std} = 0 \ , \ \rho_{ext} = 0 \quad \rightarrow \quad 1 \leq N \leq M \tag{2.13}$$

$$E^{(a)} \cap E^{(b)} = \mathbf{0} \quad \text{für} \quad E^{(a)}, E^{(b)} \in G_N \ \wedge \ a \neq b \tag{2.14}$$

Mit $N = 1$ besteht $F_K$ nur aus einem Muster. Ist $N = M$ hat sich die maximal Anzahl von orthogonalen Erwartungen in $G_N$ etabliert, was voraussetzt, daß $F_K$ aus $K = 2^M - 1$ verschiedenen Mustern besteht. Dies stellt jedoch eine triviale Klassifizierung dar.

So läßt sich der Bereich angeben, in dem sich die Anzahl der Erwartungen im *eingeschwungenen* Zustand befindet :

$$1 \leq N \leq \max\{M, K\} \tag{2.15}$$

Bei der Simulation des erweiterten Modelles (Abbildung 2.2) wurden das System mit der gleichen Musterquelle und Abfolge wie in Abbildung 1.3 erregt. In Abbildung 2.2 entspricht die Klassifizierung der ersten Spalte einem System der geringsten Aufmerksamkeit mit $\rho_{std_1} = 0$ und $\rho_{ext_1} = 0$. Für die zweite Spalte ist die Standardaufmerksamkeit des Systems $\rho_{std_2} = 1$, die Aufmerksamkeit für erkannte Obermengen jedoch sehr gering $\rho_{ext_2} = 0$.

- Beide Klassifizierungen sind nach spätestens 7 Präsentation stabil. Für alle nachfolgenden Musterpräsentationen aus $F_5$ findet keine Suche statt und, da alle Muster Obermengen bzw. identische Mengen der etablierten Erwartungen sind, herrscht ab jetzt nur noch die geringe Aufmerksamkeit $\rho_{ext}$.

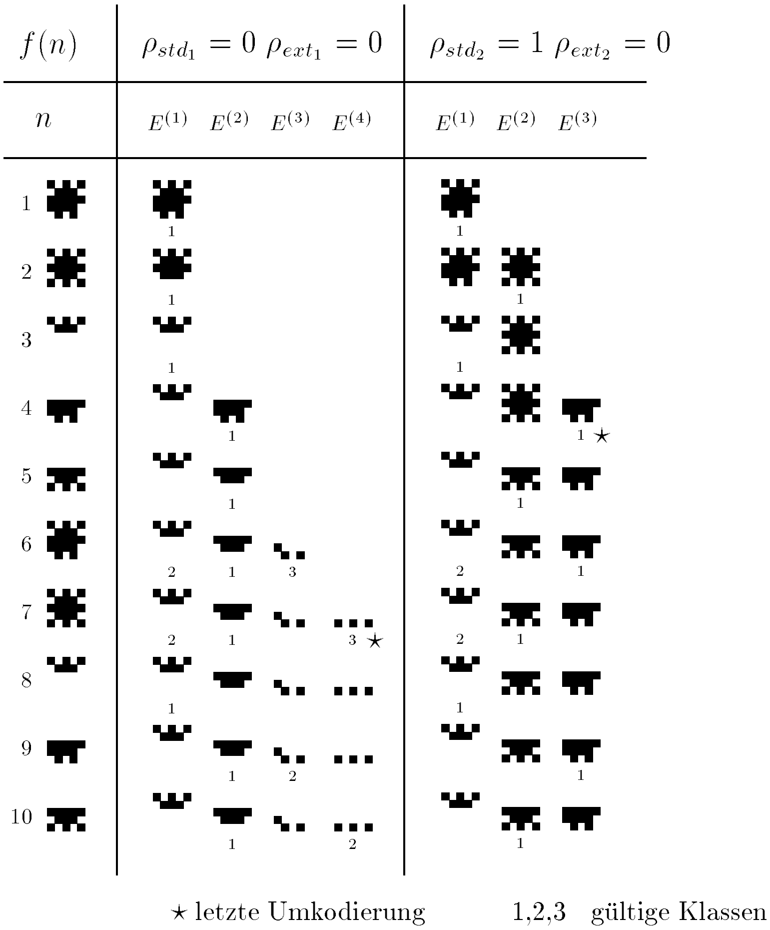


Abbildung 2.2: *Erweiterte Klassifizierung*

- Die Ziffern $1, 2, 3$ in jeder Zeile geben an welche Erwartungen dem Eingangsmuster zugeordnet sind und in welcher Reihenfolge. Die Erwartungen bilden die geordnete Reihe $G(k)$ aus $G_N$( siehe auch Abbildung 2.3).

- Alle markierten Erwartung einer Zeile sind zu einander orthogonal (siehe Gl. 2.10).

- Die Erwartungen $E^{(1)}$ und $E^{(2)}$ für $\rho_{std_1}$ und $\rho_{ext_1}$ sind die gleichen wie bei der Simulation in Abbildung 1.3. Sie sind die Grundschnittmengen, während $E^{(3)}$ und $E^{(4)}$ die jeweils fehlenden Musteranteile darstellen, die in dem erweiterten *ART*-Modell zusätzlich etabliert werden.

- Für $\rho_{std_1} = 0$ sind alle Erwartungen zueinander orthogonal, während dies für $\rho_{std_2}$ nur teilweise gilt.

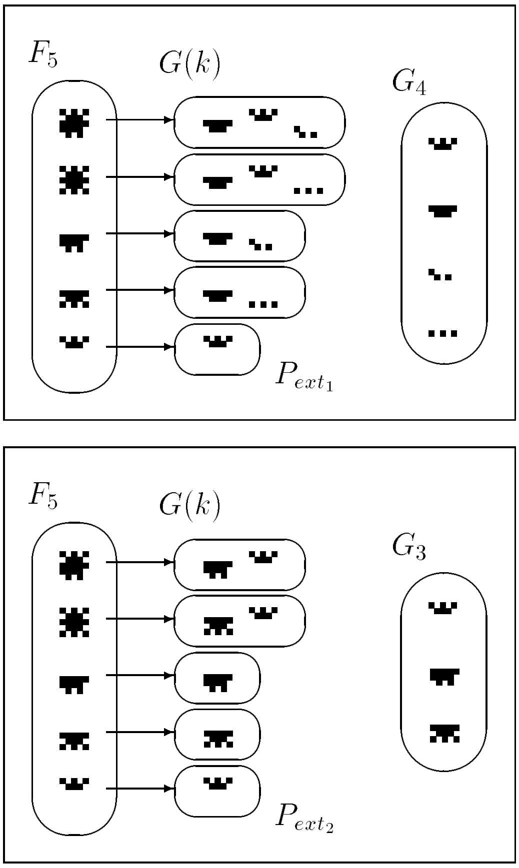


Abbildung 2.3: *Erweiterte Abbildung $P_{ext}$*

## 2.3 Parametrische Einstellbarkeit

Die oben beschriebenen Erweiterungen greifen derart massiv in den Ablauf der Klassifizierung ein, daß die Eigenschaften, die das Standard- *ART*-Modell auszeichneten für das erweiterte Modell neu untersucht werden müssen. Die Eigenschaft vom Standard- *ART–1*-Modell, mit $\rho$ die Größe von $G_N$ einstellen zu können ist im erweiterten *ART–1*-Modell verlorengegangen. Hinzugekommen ist eine verteilte Klassifizierung der Muster aus $F_K$, wobei die Teilmuster der Klassifizierung zueinander orthogonal sind (Gl. 2.14). Weiterhin bleibt auch das erweiterte *ART*-Modell in seinem Verhalten stabil, da die *2/3 Regel* in der gleichen Weise, wie im Standard-*ART*-Modell realisiert ist.

Im Zentrum der Erweiterung steht die Information, ob das Eingangsmuster zu der von ihm gewählten Erwartung eine wahre Obermenge darstellt. Diese Information wird aus einer Klassifizierungssituation heraus im System selbst generiert. Der Einfluß von außen auf die Dynamik des erweiteren *ART*-Modells erfolgt über die a-priori festgelegten Parameter $\rho_{std}$ und $\rho_{ext}$ .

Die situationsabängige Aufmerksamkeit führt dazu, daß im Verlauf der Klassifizierung immer seltener die hohe Aufmerksamkeit $\rho_{std}$ bei der Überprüfung auf Gültigkeit einer Zuordnung herrscht, bis im stabilen Zustand für die Aufmerksamkeit nur noch $\rho_{ext}$ benutzt wird. Im stabilen Zustand sind alle Eingaben Obermengen zu den Erwartungen der gewählten Klassen oder sogar identische Mengen (siehe auch Gl. 2.5).

Eine isolierte Betrachtung der Klassifizierung zweier Muster macht deutlich, wie die Anzahl der etablierten Klassen von der Beschaffenheit der Musterquelle und den Parametern $\rho_{std}$ und $\rho_{ext}$ abhängt.

- Zwei Muster $I^{(a)}$ und $I^{(b)}$, die gegenseitig keine wahren Teilmengen darstellen, prägen drei Erwartungen (siehe Abbildung 2.4 I). Die Muster werden jeweils aus zwei Erwartungen rekonstruiert. $\rho_{std}$ ist gering genug, damit sich $E^{(1)}$ als Schnittmenge beider Muster etabliert.

- Zwei Muster $I^{(a)}$ und $I^{(b)}$, von denen eins eine wahre Teilmenge des anderen ist, erzeugen zwei Erwartungen. Eine Erwartung entspricht der Teilmenge, die andere den fehlenden Merkmalen der Teilmenge zur Obermenge (siehe Abbildung 2.4 II). In diesem Fall ist $\rho_{ext}$ so gering, daß $I^{(b)}$ zu $E^{(1)}$ immer noch eine gültige Klasse darstellt.

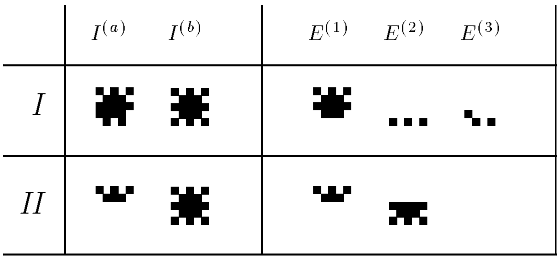


Abbildung 2.4: *Misch- und Teilmengen Präsentation*

Der Parameters $\rho_{std}$ ist also entscheidend für die Bildung der Schnittmenge zweier Mischmengen. Ein höheres $\rho_{std}$ würde die Anzahl der etablierten Klassen in Abbildung 2.4 I auf zwei sinken lassen. Eine Erhöhung von $\rho_{ext}$ in Abbildung 2.4 II würde lediglich verhindern, daß durch die Klassifizierung die Teilmengenverhältnisse erkannt werden. Die Anzahl der etablierten Klassen wäre die gleiche. Eine Überhöhung von $\rho_{ext}$ mit $\rho_{std} < \rho_{ext}$ würde zu instabilen Klassifizierungen führen; etablierte Schnittmenge von Mischmengen wie $E^{(1)}$ in Abbildung 2.4 I würden eine abermalige Klassifizierung der Muster $I^{(a)}$ und $I^{(b)}$ an der Schnittmenge $E^{(1)}$ verhindern.

## 2.4 Simulation

In einem Klassizierungsverlauf greifen die oben beschriebenen Verhaltensweisen ineinander. Diese Situationen können auch zwischen den virtuell erzeugten Mustern entstehen. Wie zum Beispiel die Erwartungen $E^{(2)}$ in der Abbildung 2.4 I und II. Statt das Verhalten der Klassifizierung formal in Abhängigkeit von $\rho_{std}$ und $\rho_{ext}$ zu beschreiben soll an dieser Stelle eine Simulation über den parametrischen Raum erfolgen, und zwei Annahmen auf ihre Gültigkeit hin untersucht werden.

Als Bewertung einer Klassifizierung soll die mittlere Anzahl der benötigten Erwartungen zur Rekonstruktion eines Musters aus der Musterquelle $F_K$ herangezogen werden.

$$\overline{n_k} = \frac{1}{K} \sum_{\nu=1}^{K} n_\nu \tag{2.16}$$

Die Klassifizierung eines Standard-*ART*-Modells wurde mit der Größe von $G_N$ also $N$ bewertet (siehe Gl. 1.11 Seite 7 ). Die Bewertung der verteilten Klassifizierung des erweiterten *ART*-Modells nach Gl. 2.16 entspricht der mittleren Größe von $G(k)$ (siehe Gl. 2.8 Seite 13 ).

Genau wie die Bewertung der Klassifizierung des Standard-*ART*-Modells, bleibt die des erweiterten Modells innerhalb eines Intervalls.

$$\overline{n_k} \in [1, \overline{n_k}_{max}] \tag{2.17}$$

Die Grenzen dieses Intervalls entsprechen denen der Abbildung $P_{ext}$ des erweiterten *ART*-Modells:

$\overline{n_k} = 1$
: In diesem Fall wird jedem Muster aus $F_K$ ein Erwartung in $G_N$ zugeordnet. Es handelt sich um die triviale Klassifizierung mit $F_K \equiv G_N$ (siehe Gl. 2.12).

$\overline{n_k} = \overline{n_k}_{max}$
: Der Wert von $\overline{n_k}_{max}$ wird durch eine Klassifizierung mit minimaler Aufmerksamkeit definiert. Für diesen Fall ist $G_N$ eine orthogonale Basis von $F_K$ (siehe Gl. 2.14).

Aus Vorüberlegungen ergaben sich zwei Annahmen, wie die Bewertung der Klassifizierung mit $\overline{n_k}$ in Verbindung mit den Parametern $\rho_{std}$ und $\rho_{ext}$ zu setzen ist.

Für ein festes $\rho_{std}$ wird angenommen :

$$\rho_{ext_1} < \rho_{ext_2} \quad \rightarrow \quad \overline{n_k}_1 > \overline{n_k}_2 \tag{2.18}$$

Für ein festes $\rho_{ext}$ wird angenommen :

$$\rho_{std_1} < \rho_{std_2} \quad \rightarrow \quad \overline{n_k}_1 > \overline{n_k}_2 \tag{2.19}$$

Als Musterquelle für die Simulationen dienen hier die ersten sechs Buchstaben des Alphabets, die mit 25 Merkmale dargestellt sind (wie auch in [2]). Für die

$I^{(a)}$ $I^{(b)}$ $I^{(c)}$ $I^{(d)}$ $I^{(e)}$ $I^{(f)}$

Abbildung 2.5: *Musterquelle der Simulation*

Simulation wurde ein *ART–2*-Modell verwendet, bei dem nach [3] die minimale Aufmerksamkeit $\rho_{min_{ART2}} = 0.7$ und nicht 0 ist.

In den Abbildungen 2.6 und 2.7 sind jeweils die *Einschwingphase* und ein Präsentationszyklus im stabilen Zustand gezeigt. Mit Abbildung 2.6 errechnet sich $\overline{n_k} = 4.667$, die Erwartungen von $G_N$ sind hier vollständig orthogonal zueinander.

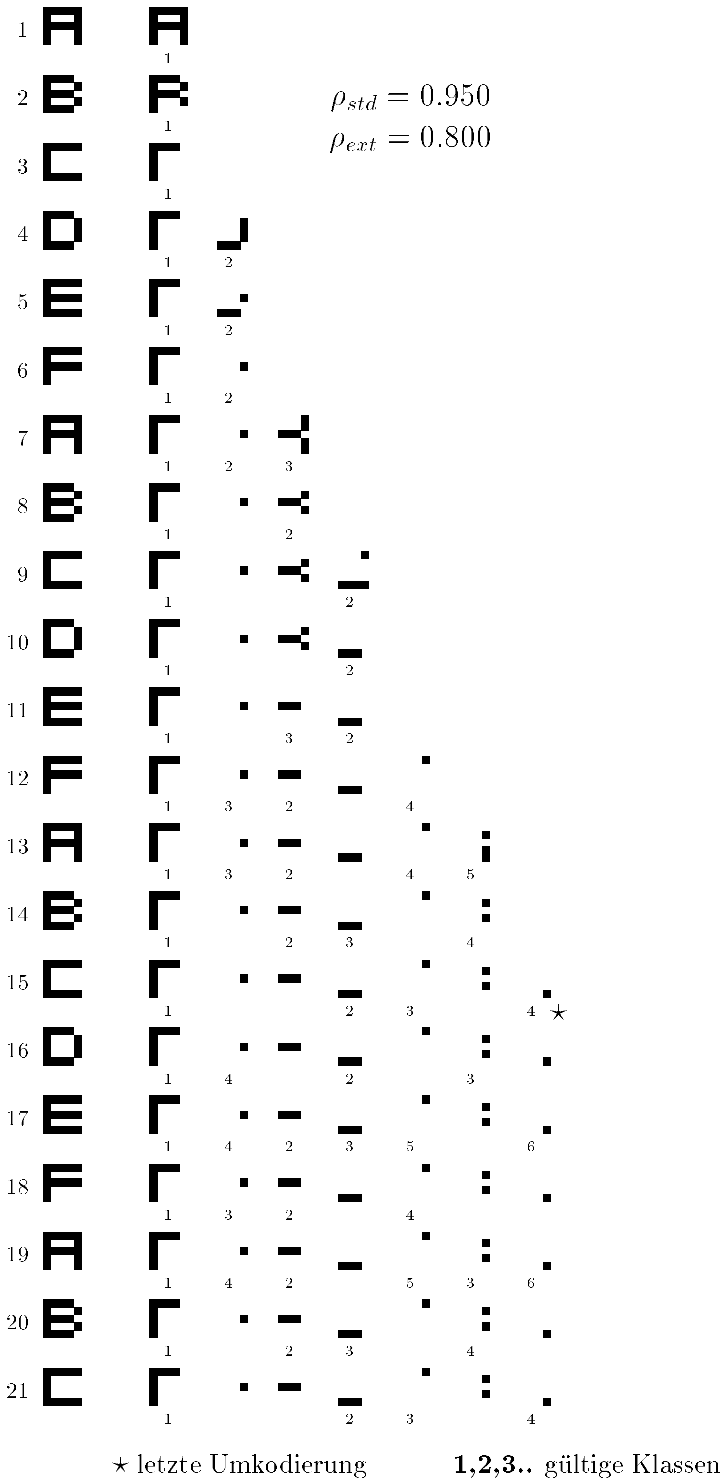


Abbildung 2.6: *Klassifizierung mit geringer Aufmerksamkeit*


Abbildung 2.7: *Beispiel einer Klassifizierung*

In der Abbildung 2.7 bilden sich für die Muster $I^{(b)}$,$I^{(d)}$ eigene Erwartungen, da $\rho_{std}$ maximal ist. Aufgrund der Teilmengenverhältnisse der übrigen Muster und der geringen Aufmerksamkeit $\rho_{ext}$ bilden sich die übrigen Erwartungen, die untereinander fast vollständig orthogonal sind. Der Bewertung bei dieser Klassifizierung liegt bei $\overline{n_k} = 2.00$.

Die Oberflächen in den Abbildungen 2.8 und 2.9 stellen die Bewertung der Klassifizierung mit $\overline{n_k}$ über den Parametern $\rho_{std}$ und $\rho_{ext}$ dar. Der Wert $\overline{n_k}$ wurde jeweils nach Erreichen des stabilen Zustandes berechnet. Beiden Simulationen liegt die gleiche Musterquelle, die sechs Buchstaben, zugrunde. Für die Simulation in Abbildung 2.8 wurden die Muster in der Abfolge $I^{(a)}$,$I^{(b)}$,$I^{(c)}$,$I^{(d)}$,$I^{(e)}$,$I^{(f)}$ zyklisch wiederholt und für die Simulation in Abbildung 2.9 umgekehrt.

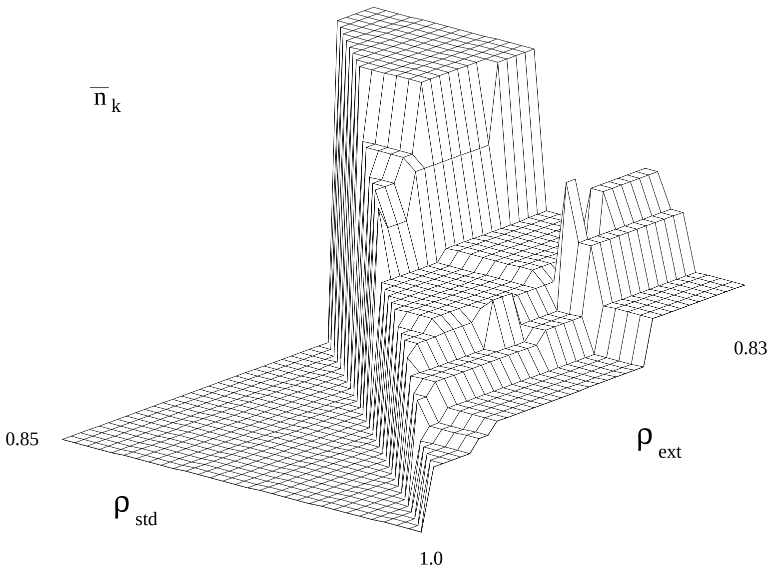


Abbildung 2.8: *Simulation 1*

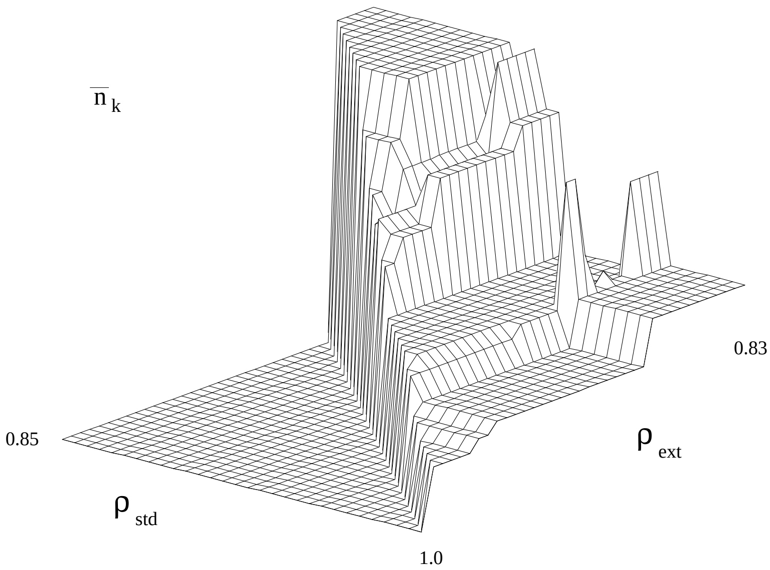


Abbildung 2.9: *Simulation 2*

Die Simulationen zeigen, daß die Annahmen 2.18 und 2.19 keine uneingeschränkte Gültigkeit haben. Es gibt Bereiche, in denen der Wert $\overline{n_k}$ wieder stark steigt. In diesen Bereichen findet man auch auch die längsten *Einschwingzeiten*. In weiten Teilen der Oberflächen findet man aber die Annahmen bestätigt und da sich die herausfallenden Bereiche in beiden Simulationen bei ähnlichem $\rho_{std}$ und $\rho_{ext}$ befinden, könnte daraus geschlossen werden, daß es sich um einen Effekt handelt, der unabhängig von der Musterabfolge ist.

Zwei Parametereinstellungen haben bei der Klassifizierung besondere Bedeutung. Diese Einstellungen liegen auf den Grenzen des parametrischen Raums und engen dadurch das dynamische Verhalten des erweiterten Systems ein.

$\rho_{std} = \rho_{ext}$
: In diesem Fall ist Situationsabhängigkeit der Aufmerksamkeit aufgehoben, Teilmengen und Mischmengen werden in der Klassifizierung mit der gleichen Aufmerksamkeit behandelt.

$\rho_{std} = 1$ und $\rho_{ext} = \rho_{min_{ART2}} = 0.7$
: Da $\rho_{std}$ maximal ist entstehen keine Erwartungen aufgrund von von Mischmengen der Musterquelle. Nur wahre Teilmengen, die in der Musterquelle vorhanden sind werden verteilt klassifiziert (siehe Abbildung 2.7).

Ein Problem, das aus dem *ART–1*-Modell hervorgeht, ist die Suche nach der maximalen Aktivität in *F2* . Wird ein Eingangsmuster dem System präsentiert, werden über die Gewicht $z_{ij}$ die Aktivitäten $X_j^{(2)}$ bestimmt. Nur das Neuron $J$, das die maximale Aktivität besitzt, erhält ein signifikantes Ausgangssignal $Y_J^{(2)} = 1$. In Klassifizierungssituationen kann es vorkommen, daß zwei Neuronen $j_a$ und $j_b$ identische Aktivitäten besitzen, $X_{j_a}^{(2)} = X_{j_b}^{(2)}$. Die Suchstrategie im *ART*-Modell bevorzugt immer das Neuron mit dem geringsten Index $J = \min\{j_a, j_b\}$.

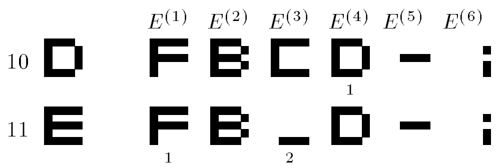


Abbildung 2.10: *Problematische Situation der Klassifizierung*

In Abbildung 2.10 taucht dieses Problem bei der Präsentation des Musters $I^{(e)}$ im Schritt 11 auf und hat entscheidenden Einfluß auf die nachfolgenden Klassifizierung.

Die Werte der Aktivitäten $X_1^{(2)}$ und $X_3^{(2)}$ sind für die Klassifizierung des ersten Teilmusters von $I^{(e)}$ identisch, da $\| I^{(e)} \cap E^{(1)} \| = \| I^{(e)} \cap E^{(3)} \|$ und $\| E^{(1)} \| = \| E^{(3)} \|$ ist. Die Suchstrategie bewirkt die Wahl von $E^{(1)}$, was zur Folge hat, daß bei der Klassifizierung des zweiten Teilmusters die Erwartung $E^{(3)}$ massiv umkodiert wird. Wäre die Wahl des ersten Teilmusters auf $E^{(3)}$ gefallen, würde mit $E^{(5)}$ das Muster $I^{(e)}$ ohne weiteres Umkodieren klassifiziert. Die Folge dieser unnötigen Umkodierung kann sein, daß die Erwartung $E^{(5)}$ im stabilen Zustand für keine Musterrekonstruktion herangezogen wird.

## 2.5 Gegenüberstellung

Hauptunterschied zwischen dem Standard- *ART*-Modell und seiner Erweiterung ist, daß die Abbildung der Musterquelle $F_K$ auf $G_N$ für beide Modelle eindeutig, aber nur für das erweiterte Modell umkehrbar ist. Weitere Unterschiede sind den folgenden Tabellen zu entnehmen.

| Abbildung | Standard | Erweiterung |
|---:|:---:|:---:|
| eindeutig | • | • |
| umkehrbar | | • |
| verteilte Klassifizierung | | • |
| Erwartungen | $K \geq N \geq 1$ | $\max\{K, M\} \geq N \geq 1$ |

Tabelle 2.1:

| *ART*-Modell | Standard | Erweiterung |
|---:|:---:|:---:|
| stabil | • | • |
| Aufmerksamkeit | $\rho$ | $\rho_{std}$ , $\rho_{ext}$ |
| Einstellgröße | $N$ | $\overline{n_k}$ |

Tabelle 2.2:

# Kapitel 3

## Übertragung auf *ART–2*

Im Gegensatz zu *ART–1* ermöglicht das von Grossberg 1987 in [3] vorgestellte *ART–2*-Modell die Klassifizierung von reellwertigen Mustervektoren mit den Eigenschaften von *ART–1*. Ein entscheidender Unterschied zu *ART–1* ist die Gestaltung des Feldes *F1* , das dem Webergestz und dem nichtlinearen Vergleichsgesetz genügen muß (siehe Abschnitt 1.3 Seite 5). Obwohl die zu klassifizierenden Muster aus einem linearen Verktorraum mit reellen Merkmalskomponenten stammen und folglich der Norm die euklidische Metrik zugrunde liegt, wird die *2/3 Regel* nach dem Prinzip einer binären Algebra realisiert.

Zwei weitere Unterschiede finden sich in der Entscheidung über die Güte der Resonanz und in der Anzahl der Neuronenfelder. Das hier benutzte *ART–2*-Modell verfügt über ein zusätzliches Feld *F0* das den Mustervektor vorverarbeitet (Abbildung 3.1).

Die Lerngleichungen für das *ART–2*-Modell ergeben sich aus der, für das *ART*-Modell grundlegenden Nicht-*Hebb'schen* oder auch kompetenzunterstützenden Lerngleichung (Gl. 1.2 Seite 1).

$$\frac{dz_{ij}}{dt} = d\,(p_i - z_{ij})Y_j^{(2)} \tag{3.1}$$

$$\frac{dz_{ji}}{dt} = d\,(p_i - z_{ji})Y_j^{(2)} \tag{3.2}$$

Das $i$-te Element des Vektors $\mathbf{p}$ aus den *F1* Feld ist die präsynaptische Aktivität, $Y^{(2)}$ die postsynaptische. Der Zusatz *kompetenzunterstützend* beruht auf der Eigenschaft des *F2* Feldes, in dem immer nur die maximale Aktivität $J$ in diesem Neuronenfeld einen signifikanten Ausgang erhält. Hierdurch werden die Gewichte $z_{ji}$ und $z_{ij}$, die nicht zu oder von dem Neuron $J$ aus *F2* verlaufen nicht geändert.

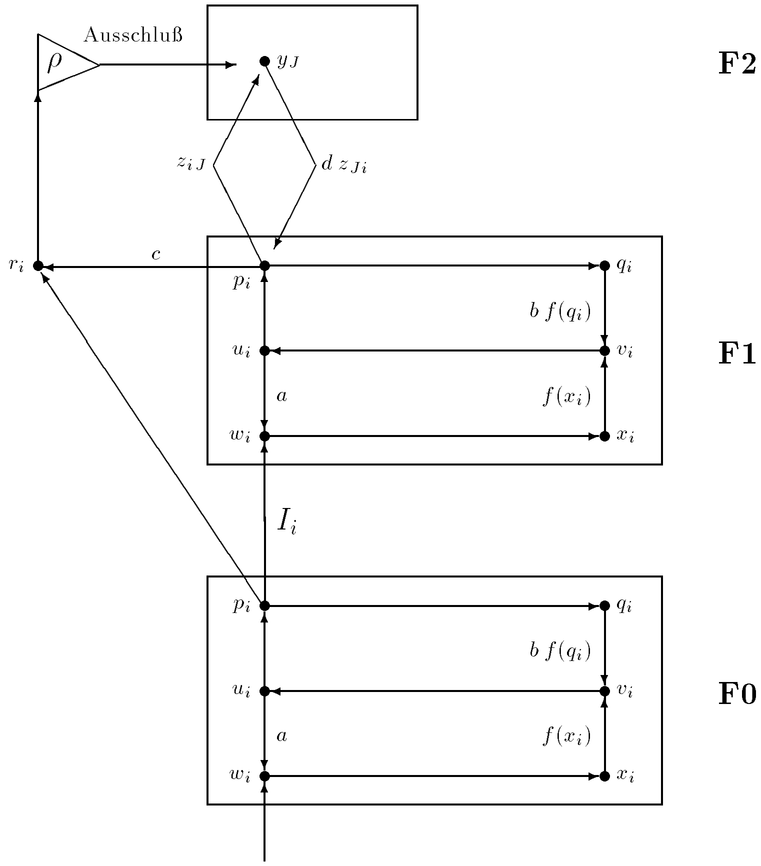


Abbildung 3.1: *ART–2 Struktur*

## 3.1 Das *F1* Feld

Das *F1* Feld besteht aus 6 Neuronen für jede $i$-te Komponente des Eingangsvektors. Die Zustände aller Neuronen werden auf den Wert 0 zurückgesetzt und dann iterativ ins Gleichgewicht gebracht.

**Iterationsschritt 1** Zuerst werden die Zustände der Neuronen $w_i$ und $p_i$ über die folgenden Gleichungen neu berechnet.

$$w_i = I_i + au_i \tag{3.3}$$

$$p_i = u_i + \sum_i dY_j^{(2)}\, z_{ji} = u_i + dE_i^{(J)} \tag{3.4}$$

$I_i$ ist die i-te Komponente des Eingangsvektors und $E_i^{(J)}$ die, der Erwartung des aktiven Neurons $J$ aus *F2* . Ist kein Neuron in *F2* aktiv ist $p_i = u_i$. Der Faktor $d$ ist die Lernrate der für das System zugrundeliegenden Lerngleichungen 3.2 und 3.1.

**Iterationsschritt 2** Mit den Normen der Vektoren $\mathbf{p}$ und $\mathbf{w}$ werden die normierten Vektoren $\mathbf{q}$ und $\mathbf{x}$ bestimmt.

$$q_i = \frac{p_i}{\|\mathbf{p}\|_2} \tag{3.5}$$

$$x_i = \frac{w_i}{\|\mathbf{w}\|_2} \tag{3.6}$$

Über eine Funktion $f$ werden die beide Vektoren addiert. Dieser Schritt stellt den eigentlichen Vergleich zwischen dem Eingangsmuster und der Erwartung dar.

$$v_i = f(x_i) + bf(q_i) \tag{3.7}$$

Die Funktion $f$ ist im Allgemeinen eine Funktion, die oberhalb des Parameters $\theta \in [0,1]$ linear verläuft, unterhalb jedoch nichtlineare ist (siehe auch Abbildung 3.2):

$$f_1(x) = \begin{cases} \dfrac{2\theta x^2}{x^2+\theta^2} & : \quad 0 \le x \le \theta \\ x & : \quad \theta < x \end{cases} \tag{3.8}$$

$$f_2(x) = \begin{cases} \dfrac{x}{1+e^{b_s(\theta - x)}} & : \quad 0 \le x \le \theta \\ x & : \quad \theta < x \end{cases} \tag{3.9}$$

$$f_3(x) = \begin{cases} 0 & : \quad 0 \le x \le \theta \\ x & : \quad \theta < x \end{cases} \tag{3.10}$$

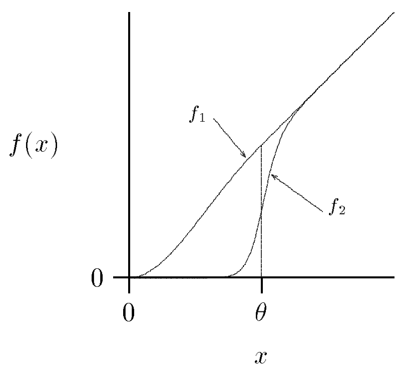


Abbildung 3.2: *Nichtlineare Kontrastfunktionen*

Auf die Bedeutung dieser Funktion wird im Folgenden weiter eingegangen.

**Iterationsschritt 3** Der Vektors **u** ist der normierte Vektor **v**. So ist **u** der Vektor, der normiert den Vergleich zwischen Eingangsmuster und Erwartung darstellt.

$$u_i = \frac{v_i}{\| \mathbf{v} \|_2} \tag{3.11}$$

**Iterationsschritt 4** Die Iteration endet, wenn die Änderungen von **p** und **w** bezogen auf den Zustand im letzten Iterationszyklus unter eine Schranke fällt. Andernfalls wird ein neuer Iterationszyklus eingeleitet.

Ist *F2* inaktiv liegt keine Erwartung an *F1* an und in Gl. 3.4 wird $E_i^{(J)}$ als 0 behandelt. In dieser Situation wird durch das *F1* Feld der Eingangsvektor $I$ normiert und nichtlinear kontrastverstärkt auf den Vektor **u** abgebildet (siehe Abbildung 3.3). Hohe Werte für $\theta$ bewirken eine große Kontrastverstärkung.

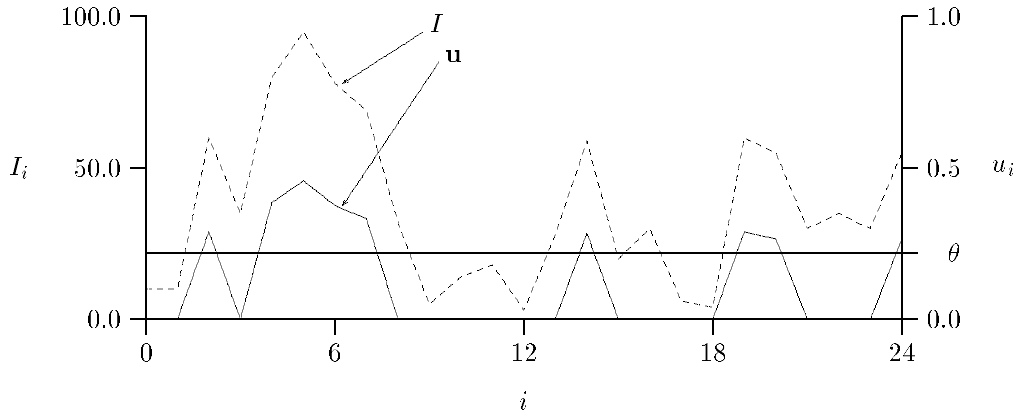


Abbildung 3.3: *Kontrastverstärkung und Normierung eines Mustervektors*

Die nichtlineare Funktion $f$ in Gl. 3.7 bewirkt, daß bei der iterativen Stabilisierung der Zustände in *F1* die geringen Merkmalsanteile im Eingangsvektor unterdrückt werden. Nach der Zusammenführung sind diese in **u** nach Gl. 3.11 schwächer bewertet, als die Merkmalsanteile, die oberhalb von $\theta$ liegen.

In dem anderen Fall, in dem *F2* aktiv ist und somit der Summand $dE_i^{(J)}$ in Gl. 3.4 bei der Iteration berücksichtigt wird, besitzt *F1* zusätzlich zu den Eigenschaften der Normierung und der Kontrastverstärkung auch noch die des Vergleichs zwischen der Erwartung und dem Eingangsvektor. Nach der iterativen Stabilisierung von *F1* bleiben die Aktivitäten $u_i$ signifikant, die sowohl in $I_i$ als auch in $E_i^{(j)}$ vorhanden sich. Die Anteile, die in $E^{(J)}$ vorhanden sind, in $I$ aber nicht, erfahren durch die Vergleichseigenschaft eine Minderung ihrer Signifikanz

(Abbildung 3.4). Die Faktoren $a > 1$ und $b > 1$ unterstützen die Dominanz von Merkmalen aus $E^{(J)}$ im Vergleich mit $I$, so daß Merkmale die zusätzlich in $I$ gegenüber $E^{(J)}$ vorhanden sind, unterdrückt werden.

Somit ist mit dem *F1* -Feld die *2/3 Regel* realisiert, die zu den Voraussetzungen der stabilen Klassifzierung gehört (siehe Abschnitt 1.3 Seite 5). Nach Grossberg (siehe [3]) stellt diese Realisierung eine schwache, aber ausreichende Form der *2/3 Regel* dar. In dem binären *ART*-Modell ist die *2/3 Regel* dort durch die Schnittmengenbildung zwischen Erwartung und Eingangsmuster repräsentiert ist (Gl. 1.3 Seite 6). Im Vergleich mit dieser Schnittmengenbildung ist die *2/3 Regel* in *ART–2* als eine schwächere zu verstehen.

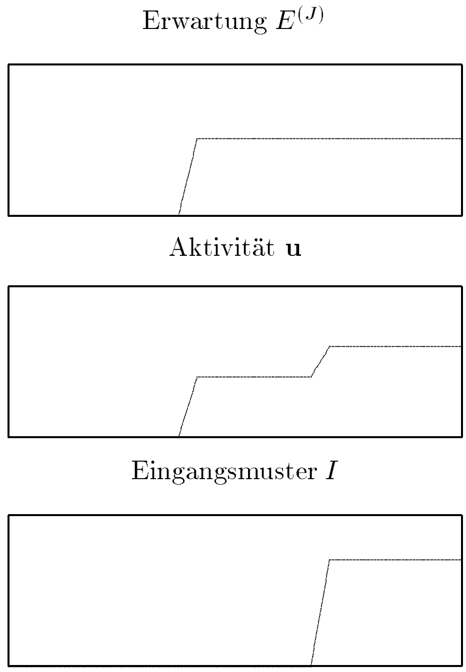


Abbildung 3.4: *Vergleich zwischen Muster und Erwartung*

Das *F1* Feld besitzt somit die Eigenschaften, die wie in *ART–1* für das stabile Verhalten des Modells nötig sind:

**2/3 Regel:** Diese Bedingung ist hier in einer schwachen Form realisiert und beruht auf einer zusätzlichen Kontrastverstärkung des Musters.

**Webergesetz:** Normierung des Mustervektors erfolgt nach der euklidischen Normdefinition reellwertiger Vektoren.

Um die Stabilisierung der Klassifizierung zu begünstigen werden die Mustervektoren $I$ der Musterquelle einer Vorverarbeitung unterworfen, die mit einem zusätzlichen Neuronenfeld *F0* geleistet wird (siehe Abbildung 3.1). Dieses Feld ist identisch zu *F1* aufgebaut, und vollzieht die Normierung und Kontrastverstärkung des Eingangsmusters $I$. Im weiteren soll für das *ART–2*-Modell $I$ jetzt immer das vorverarbeitete Eingangsmuster bezeichnen.

## 3.2 Aufmerksamkeit

Um über die Gültigkeit einer Klassenzuordnung eines Musters zu entscheiden wird der Vektor $\mathbf{r}$, wie in Abbildung 3.1 angedeutet aus dem Vektoren $\mathbf{p}$ des Feldes *F1* und dem Vektor $I$, dem vorverarbeiteten Mustervektor gebildet.

$$r_i = \frac{I_i + cp_i}{\|\mathbf{p}\|_2} \tag{3.12}$$

Die Gleichung 3.12 stellt ähnlich wie Gl. 3.7 einen Vergleich zwischen den Vektoren $\mathbf{p}$ und $I$ dar. Genügt die Norm des Vektors $\mathbf{r}$ der Relation

$$\rho \ < \|\mathbf{r}\|_2 \quad , \tag{3.13}$$

ist die Klassifizierung gültig, andernfalls wird das Neuron $J$ aus *F2* für die weitere Suche nach einer Klassifizierung ausgeschlossen. Wie in *ART–1* hat auch *ART–2* ein Register, in dem die Verfügbarkeit der Neuronen aus *F2* verwaltet wird.

Aus analytischen Betrachtungen der Gleichung 3.12 ([3]) ergibt sich, daß $\rho \in [0.7, 1]$ sein muß. Die Norm der Erwartungen konvergiert bei der Integration der Lerngleichung gegen einen festen Wert, d.h. :

$$\| E^{(J)} \|_2 \to \ \frac{1}{1-d} \tag{3.14}$$

Damit in jeder Situation die Stabilität des Modells gewährleistet ist, müssen die Parameter $d$ und $c$ folgender Ungleichung genügen:

$$\frac{cd}{1-d} \ \leq \ 1 \tag{3.15}$$

und für die Gewichte muß eine Initialisierung erfolgen mit :

$$z_{ij} \ \leq \ \frac{1}{(1-d)\sqrt{M}} \quad \text{und} \quad z_{ji} = 0 \tag{3.16}$$

## 3.3 Erweiterung von *ART–2*

Um darüber zu entscheiden, ob mit das Eingangsmuster eine Ober-, Teil- oder Mischmenge der Erwartung ist muß eine Aussage über Merkmalsanzahl von $I$ und $E^{(J)}$ gemacht werden. Für diese Aussage wird die binäre Algebra benutzt, obwohl es sich bei den zu klassifizierenden Muster um reellwertige Vektoren handelt. Die Vorgehensweise ist jedoch zulässig und zwingend, da im Kern einer jeden *ART*-Struktur die *2/3 Regel* nach einer Verknüpfung der binären Algebra realisiert ist.

Ist im *ART–1*-Modell die Norm eines Mustervektors über den Hammingabstand (Gl. 1.5 Seite 6 ) definiert, wird an dieser Stelle eine Norm eingeführt, die die Merkmalsanzahl des reellwertigen, normierten und kontrastverstärkten $M$-Tupels $I$ mit Hilfe einer Inversionsfunktion definiert.

$$f_{inv}(x) = \begin{cases} 1 & : \quad 0 \leq x \leq \theta \\ 0 & : \quad \theta < x \end{cases} \tag{3.17}$$

$$\| I \|_{art} = M - \sum_{\nu=1}^{M} f_{inv}(I_\nu) \tag{3.18}$$

Mit Hilfe dieser Normdefinition kann nun vorab gesagt werden :

- Ist $\| I \|_{art} > \| E^{(J)} \|_{art}$ , kann das Eingangsmuster eine Obermenge der Erwartung sein und es ist

$$O^{(E,I)} = I \quad \text{und} \quad T^{(E,I)} = E^{(J)} \tag{3.19}$$

- Ist $\| E^{(J)} \|_{art} > \| I \|_{art}$ , kann die Erwartung eine Obermenge des Eingangsmusters sein und es ist

$$O^{(E,I)} = E^{(J)} \quad \text{und} \quad T^{(E,I)} = I \tag{3.20}$$

Für den allgemeinen Fall, muß die Erwartung vor der Normbildung nach Gl. 3.18 noch kontrastverstärkt werden. Dies geschieht mit einem zusätzlichen *STM* -Feld *F0x* (siehe Abbildung 3.5).

Mit der Information über die Mengenverhältnisse wird nun aus den Vektoren $O^{(E,I)}$ und $T^{(E,I)}$ ein Merkmalsvektor erzeugt, der den Teil der von $O^{(E,I)}$ repräsentiert, der nicht in $T^{(E,I)}$ vorhanden ist (siehe Abbildung 3.5). Die komponentenweise Multiplikation von $T^{(E,I)}$ und $O^{(E,I)}$ führt zu einem Vektor $S$.

$$S_i = f_{inv}(T_i^{(E,I)})\, O_i^{(E,I)} \tag{3.21}$$

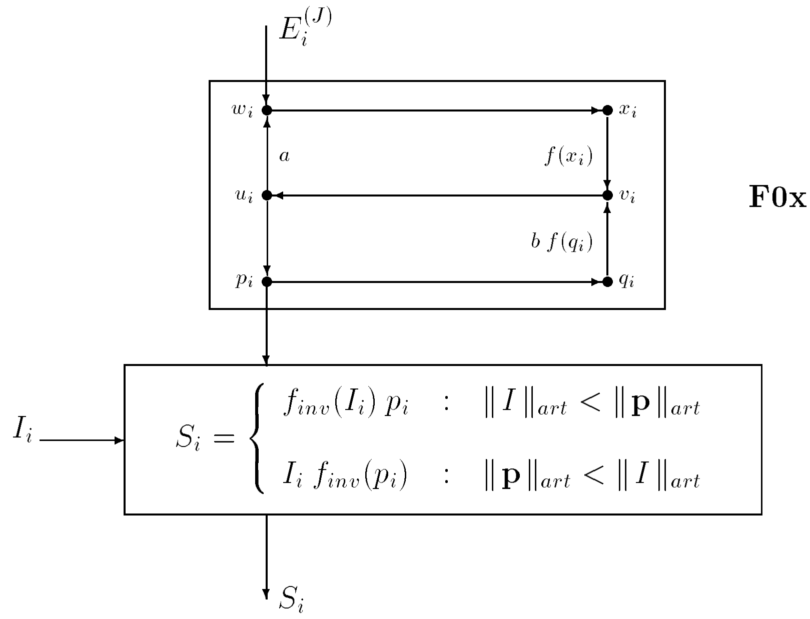


Abbildung 3.5: *Struktur der Erweiterung*

Es wird also ähnlich zu Gl. 2.6 auf Seite 12 die Schnittmenge zwischen $\overline{T^{(E,I)}}$ und $O^{(E,I)}$ gebildet.

Mit der Norm des Merkmalverktor $S$, $\| S \|_{art}$ kann nun darüber entschieden werden kann, ob mit dem Eingangsmuster $I$ eine Obermenge oder eine Teilmenge zur Erwartung $E^{(J)}$ vorliegt.

$$I = \begin{cases} \subset E^{(J)} & : \; \| S \|_{art} + \| I \|_{art} = \| E^{(J)} \|_{art} \\ \supset E^{(J)} & : \; \| S \|_{art} + \| E^{(J)} \|_{art} = \| I \|_{art} \end{cases} \tag{3.22}$$

Wie in dem *ART–1*-Modell wird die Erweiterung für das *ART–2*-Modell aus dieser Information Gl. 3.22 formuliert.

1. Ist nun Gl. 3.22 in einer der beiden Formen erfüllt, wird mit der Aufmerksamkeit $\rho_{ext}$ über die Gültigkeit der Klassifizierung von $I$ auf $E^{(J)}$ entschieden.

2. Ist zudem $I \supset E^{(J)}$ wird der Vektor $S$ als orthogonale Abspaltung von $I$ an $E^{(J)}$ dem System zur weiteren Klassifizierung vorgehalten.

Eine Simulation mit reellwertigen Mustervektoren (aus [3]) verdeutlicht die orthogonale Rekonstruktion der Mustervektoren im Sinne der binären Algebra (Abbildung 3.6 und 3.7).

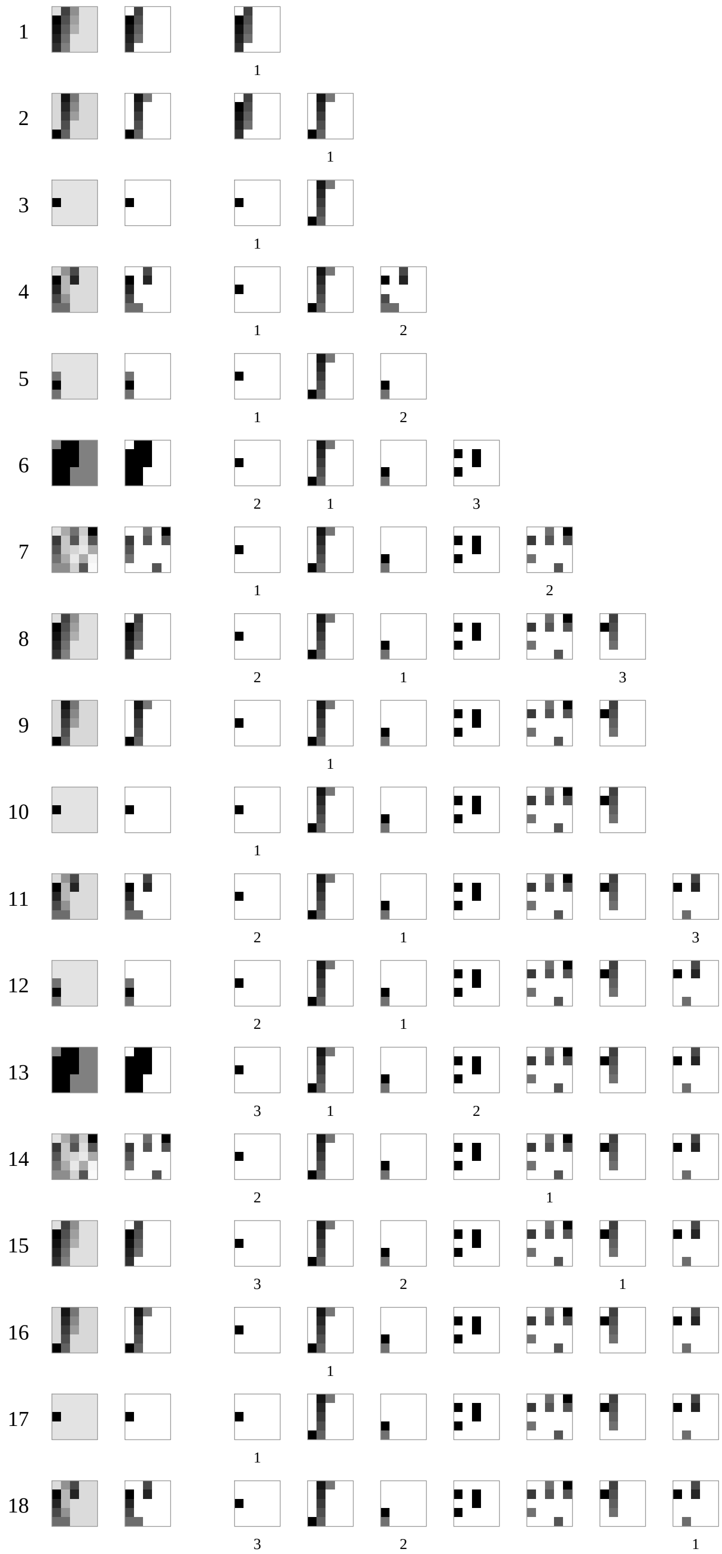

Abbildung 3.6: *Klassifizierungverlauf mit reelwertigen Mustervektoren*

Die Musterquelle besitzt hier 7 Muster, die zyklisch wiederholt dem System präsentiert werden. Die Abbildungen 3.6 und 3.7 zeigen in der ersten, linke Spalte das Eingangsmuster und in der zweiten das kontrastverstärkte. Die weiteren Spalten stellen wie in den Abbildungen 2.7 auf Seite 21 und 2.6 auf Seite 20 die zu den Neuronen aus *F2* gehörenden Erwartungen dar. Für die Graustufendarstellung wurden die Mustervektoren und Erwartungen nicht auf die euklidische Norm des Vektor $I$ bzw. $E^{(J)}$ bezogen, sondern auf das jeweils maximale Element des Vektors.

In der Abbildung 3.6 herrscht eine Auferksamkeit von $\rho_{std} = 0.99$ und $\rho_{ext} = 0.7$, ferner zeigt die Abbildung sowohl die *Einschwingphase* als auch die stabile Klassifizierung aller Muster, die nach der 12. Musterpräsentation erfolgt.

Die Abbildung 3.7 zeigt hingegen nur die stabile Klassifizierung ohne die *Einschwingphase*. Die Aufmerksamkeitsparameter lauten für diese Simulation $\rho_{std} = 0.9$ und $\rho_{ext} = 0.9$.

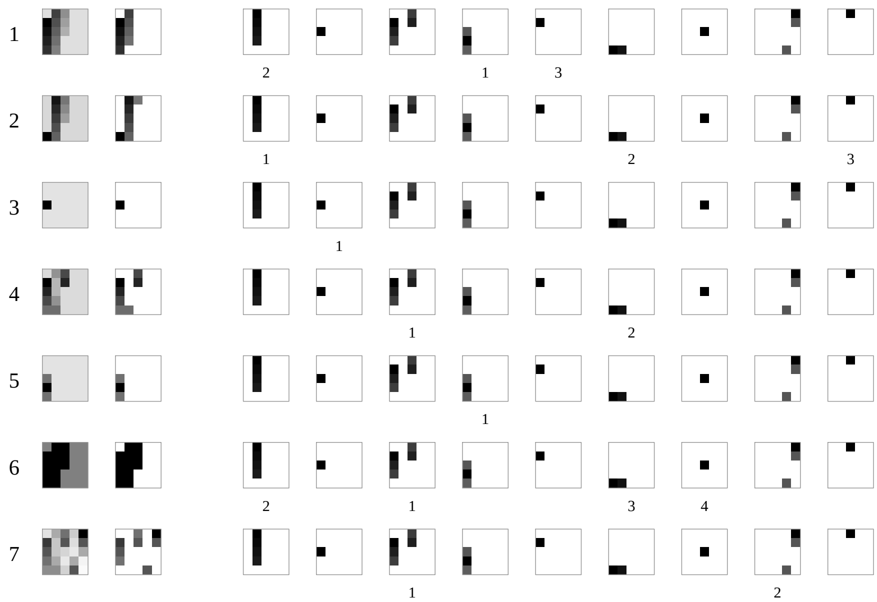


Abbildung 3.7: *stabile Klassifizierung reelwertiger Mustervektoren*

# Kapitel 4

## Zusammenfassung

Zu der parallel organisierten Suche nach einer geeigneten Klasse für ein Muster und der stabilen Klassifizierung dieses Musters im Standard- *ART*-Modell ist zusätzlich eine Analyse hinzugekommen, die es erlaubt die Muster verteilt zu klassifizieren. Diese Klassifizierung entspricht einer vollständigen Rekonstruktion aus orthogonalen Musteranteilen.

Die Analyseeigenschaft des erweiterten *ART*-Modells wurde durch zwei Zusätze erreicht:

- Situationsabhängige Steuerung der Aufmerksamkeit $\rho$ aus dem System heraus
- Merkmale, die ein Eingangsmuster (Obermenge) zusätzlich zu seiner gültigen Erwartung (Teilmenge) besitzt, werden nicht wie im Standard-*ART*-Modell ignoriert, sondern noch als virtuelles Muster dem System präsentiert.

Die parametrische Einstellbarkeit der Klassenanzahl des Standard-*ART*-Modells ist verlorengegangen. Meßreihen über den parametrischen Raum haben gezeigt, daß sich die mittlere Anzahl, der zur Musterrekonstruktion benötigten Musteranteile über die Aufmerksamkeit beschränkt einstellen läßt.

Die Übertragung der Erweiterung auf das *ART–2* Modell erweist sich als machbar, führt jedoch nicht zur Zerlegung superpositionierter, reellwertiger Mustervektoren.

In einer Implementation auf einem NeXT-Computer wurde im Rahmen dieser Arbeit das *ART–2* Modul realisiert, das sowohl das von Grossberg 1987 vorgestellte Modell, als auch dessen Erweiterung realisiert.

# Kapitel 5

## Bedienoberfläche

Im Rahmen dieser Arbeit entstand auf einem NeXT-Computer eine Implementation, die den komfortablen Zugriff auf alle Parameter des Standard- *ART*-Modells und seiner besprochenen Erweiterung bietet. Über eine einfache Dateischnittstelle (siehe Seite A.5) kann eine Musterquelle $F_K$ geladen werden und einem *ART*-Modul zugeführt werden. Das Klassifizierungsverhalten wird visualisiert, und die Gewichte $z_{ij}$ und $z_{ji}$ können in eine Datei ausgegeben werden. Für den Zugriff auf die Modulparameter und für die Ablaufsteuerung der Klassifizierung stehen in der Applikation zwei Bedienfenster und ein Ausgabefenster bereit.

### 5.1 Controlpanel

In diesem Fenster wird angezeigt welche Musterquelle geladen ist, aus wievielen Mustern sie besteht und wie groß die Dimension der Vektoren ist (siehe Abbildung 5.1). Aus diesem Fenster wird auch noch die Zuführung der Muster zum *ART*-Modul des aktiven `ARTDoc`-Objektes gesteuert.

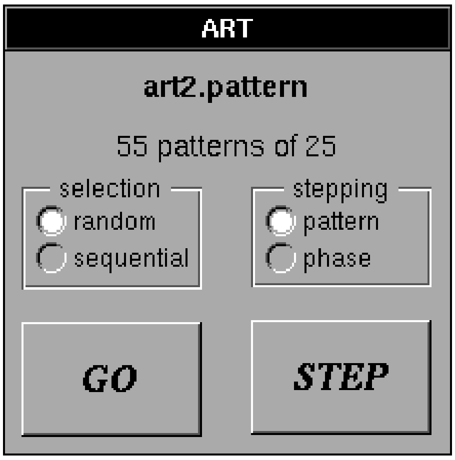


Abbildung 5.1: *Controlpanel der Applikation*

- Durch 'Drücken' des *GO* Knopfes werden dem *ART*-Modul kontinuierlich Mustervektoren zur Klassifizierung präsentiert. Nach dem 'Drücken' diese Knopfes ändert er seine Darstellung und Funktion, so daß mit ihm die kontinuierliche Präsentation auch wieder gestoppt werden kann. Die Musterabfolge kann entweder zufällig oder sequentiell gewählt werden.

- Mit dem *Step* Knopf wird der Ablauf der Klassifizierung nach Erreichen eines Schrittes wieder unterbrochen und für den nächsten Schritt muß der Knopf erneut 'gedrückt' werden. Je nach getroffener Auswahl bedeutet ein Schritt eine komplette Klassifizierung eines Musters oder eine einzelne Phase im Ablauf der Klassifizierung.

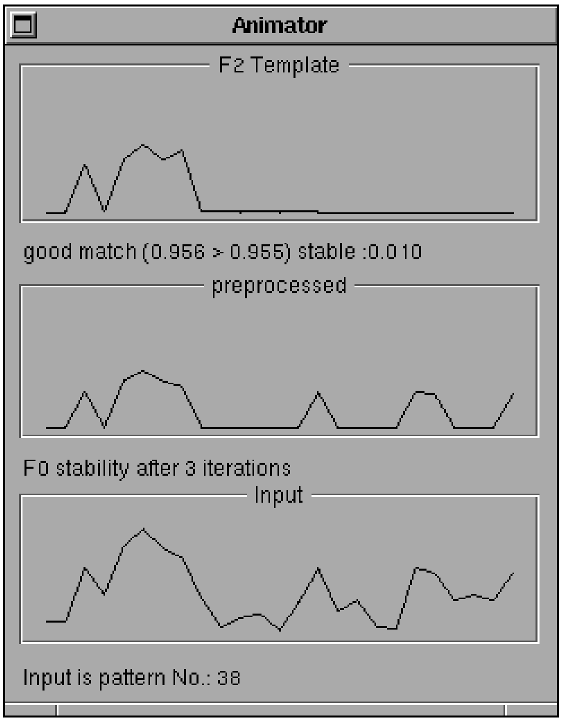


Abbildung 5.2: *Vektordarstellung im Animatorpanel*

## 5.2 Animatorpanel

In dem Fenster werden die Zustände des momentanen *ART*-Moduls je nach Wahl als Vektoren oder als zweidimensionale Graustufenmuster dargestellt (siehe Abbildung 5.2 und 5.3). Zusätzlich werden Kommentare ausgegeben, die das Verständnis für den Klassifizierungsablauf erleichtern sollen.

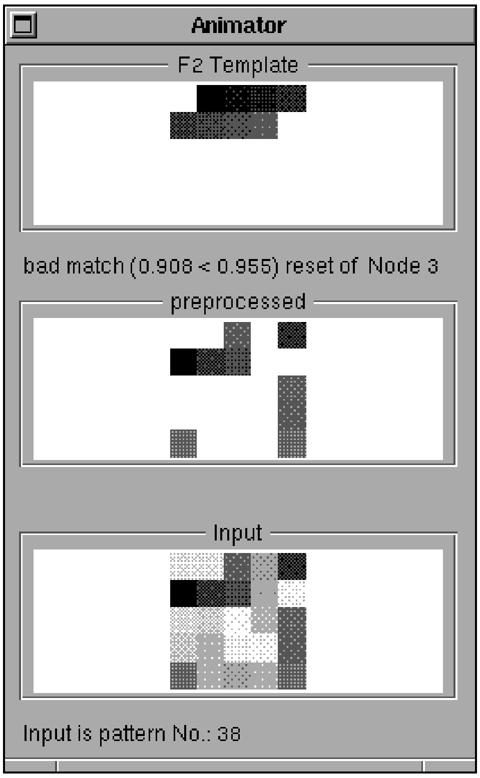


Abbildung 5.3: *Graustufendarstellung im Animatorpanel*

Das untere Fenster zeigt das Eingangsmuster, das mittlere seine kontrastverstärkte und normierte Darstellung (Vektor $\mathbf{p}$ aus *F0* siehe Abbildung 3.1 Seite 26 ) und das obere die normierte Erwartung $E^{(J)}$.

## 5.3 Document-Window

Dieses Fenster (Abbildung 5.4) zeigt Informationen an, die sich auf die Einstellungen des *ART*-Moduls und auf die Klassifizierungszustände beziehen. Im folgenden werden die Unterfenster des Document-Window beschrieben:

**ART Mode:** Es kann zwischen der Standard-*ART–2*-Modell und seiner Erweiterung gewählt werden.

**Learning:** Die Auswahl besteht zwischen dem *langsamen Lernen*, wie auf Seite 5 beschrieben, und einem stabilen Lernen, bei dem das Muster so oft dem *ART*-Modul präsentiert wird, bis die Änderung der Gewichte unter eine Schranke fallen, die mit dem Eingabefeld stable des Unterfensters Integration definiert werden kann.

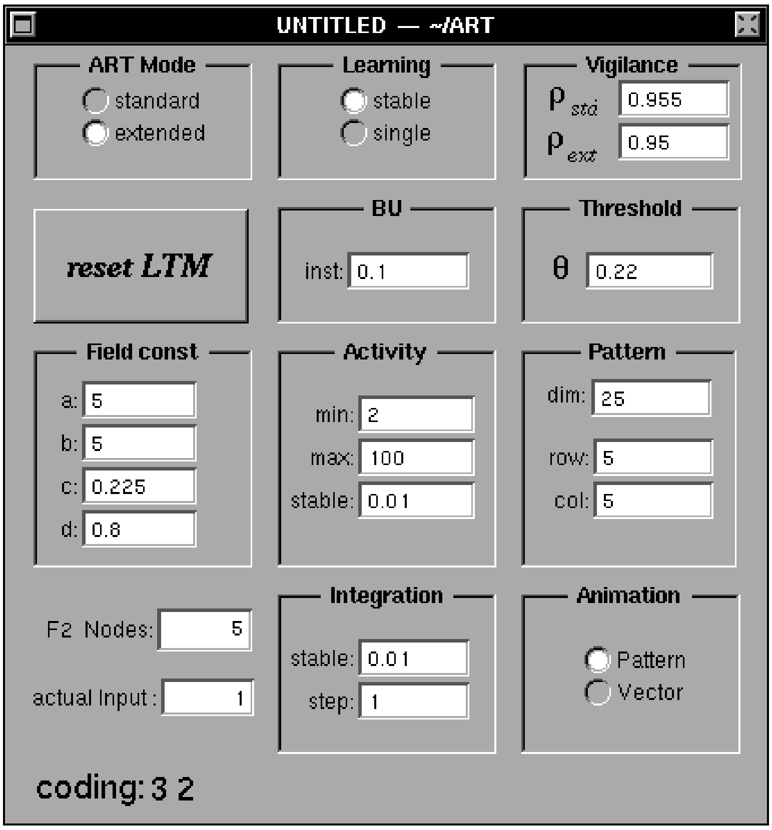


Abbildung 5.4: *Documentwindow der Applikation*

**Vigilance:** Hier werden die Aufmerksamkeiten $\rho_{std}$ und $\rho_{ext}$ eingestellt.

**reset** *LTM* **:** Mit diesem Knopf kann das *LTM* -Objekt vollständig initialisiert werden, alle erlernten Erwartungen werden gelöscht.

**BU:** Bei der Initialisierung nach Gl. 3.16 auf Seite 30 werden die Gewichte $z_{ij}$ zusätzlich mit dem Faktor aus diesem Eingabefenster multipliziert.

**Threshold:** In diesem Unterfenster kann der Schwellwert der nichtlinearen Funktion $f$ der *STM* -Felder verändert werden. Bei der Alloziierung eines neuen `ARTDoc`-Objektes wird der Wert auf $\theta = 1.1/\sqrt{M}$ voreingestellt.

**Field const:** Die Parameter $a$ und $b$ der *STM* -Objekte können in der Größe variiert werden; die Einstellungen der Lernrate $d$ (Gl. 3.1 und 3.2 Seite 25 und des Faktors $c$ ( Gl. 3.12 Seite 30) können ebenfalls geändert werden.

**Activity:** Die Angabe von min definiert die geringste, die von max die höchste Anzahl der Iterationen bei der Stabilisierung mit *STM* -Objektmethode `(int)iterate` (Seite 45). Mit stable wird die Schranke definiert, ab der die Änderung der Aktivitäten des *F1* -Feldes vernachlässigt wird und die Iteration als stabil bezeichnet wird (siehe auch Abschnitt 3.1).

**Pattern:** Mit row und col kann die zweidimensionale Graustufendarstellung eingestellt werden. Das Feld dim kann nicht editiert werden.

**Integration:** Ist im Unterfenster Learning die Auswahl stable angewählt, wird in diesem Unterfenster mit dem Eingabefenster stable definiert, ab wann die Veränderung der Gewichte vernachlässigt wird, und die Lerndifferentialgleichungen als konvergiert bezeichnet wird. Mit step wird die Schrittweite der numerischen Integration nach der Formel von Runge Kutta festgelegt (siehe auch A.2).

**Animation:** Hier kann die Darstellung im Animatorpanel umgeschaltet werden zwischen der Vektor- oder Graustufendarstellung.

**actual Input:** Der Index des momentan zur Klassifizierung anstehenden Musters der Musterquelle erscheint hier.

*F2* **Nodes:** Dieses Ausgabefeld gibt die Anzahl der momentan etablierten Klassen an.

**coding:** An dieser Stelle erscheint das Klassifizierungsergebnis. Für das Standard-*ART–2*-Modell ist es nur ein Index (siehe Gl. 1.9 Seite 7), für das erweiterte Modell die Reihe nacheinander aktivierter Neuronen (siehe Gl. 2.7 Seite 13). Ist das aktuelle Muster ohne Suche klassifiziert wird dieses mit einem no search angezeigt. Wurden zusätzlich alle Muster der Musterquelle ohne Suche klassifiziert wechselt der Hintergrund dieses Ausgabefeldes von grau nach weiß.

# Literaturverzeichnis


[1] Carpenter, G.A. : Neural Network Models for Pattern Recognition and Associative Memory,
*Neural Networks, Vol.2, Pergamon Press 1989*

[2] Carpenter, G.A., Grossberg, S. : A massively parallel architecture for a self-organizing neural pattern recognition machine,
*Computer Vision, Graphics and Image Processing, Academic Press, Inc., 1987*

[3] Carpenter, G.A., Grossberg, S. : *ART–2*: self-organization of stable category recognition codes for analog input patterns,
*Applied Optics, Vol.26, 1987*

[4] Carpenter, G.A., Grossberg, S. : *ART–3*: hierachical search using chemical transmitter in self-organizing Pattern recognition architectures,
*Neural Networks, Vol.3, Pergamon Press, 1990*

[5] Brause, R. : Neuronale Netze,
*B.G. Teubner Stuttgart, 1991*

[6] Mertins, A. : Statistische Nachrichtentheorie,
*Studienskriptum , 1.Auflage, TUHH 1992*

[7] Gaudiano, P. : *ART–2* Source Code,
*Boston University 1990*

[8] Müller-Tomfelde, C. : Adaptive Resonanztheorie,
*Vortrag im Rahmen des Seminars Neuronale Netze, TUHH 1993*

# Anhang A

## Implementation

Der Algorithmus für das *ART–2*-Modell und die vorhergehend besprochene Erweiterung liegt als Quellcode in *Objective-C* und als compilierte Applikation für einen NeXT-Computer vor. Der dem NeXT Software Kit zugrunde liegende *Objective-C* Compiler ist ein erweiterter C-Compiler unter Beibehaltung des ANSI Standards. Das *Application Kit*, der *Interface Builder* und *Project Builder* stellen die grundlegenden Werkzeuge der Programmierung dar.

Anleihen wurden bei einer reinen *C*-Implementation aus [7] gemacht und in *Objective-C* übersetzt.

**Application Kit :** Eine Objekthierachie, als Basis einer Applikation und als Quelle von Appliktionselementen, wie Fenster, Menueleisten und Ein-und Ausgabezellen.

**Interface Builder :** Eine Applikation zur Erstellung von Benutzeroberflächen zur Integration mit *Objective-C* erstellten Programmen. Hiermit werden die Schnittstellen zwischen dem Programm und seiner Darstellung auf dem Bildschirm generiert und graphisch editierbar verwaltet.

**Project Builder :** Ein zentrale Umgebung zum Compilieren von Quellcode in *Objective-C*. Hier werden die zur Schaffung von Applikation benötigten Dateien verwaltet.

Die Erweiterungen von *Objective-C* gegenüber ANSI-C bestehen aus den einer objekt-orientierten Hochsprache eigenen Elementen der Programmierung, wie Klassen- und Objektdefinitionen, deren Instanziierung und Vererbung und der Kommunikation zwischen den Instanzen. So ist zum Beispiel die Syntax zum Aufruf einer Methode eines Objektes

```
[receiver message]
```

Der *receiver* ist ein Zeiger auf eine Objektstrukur; die *message* muß eine Methode sein, die das Empfängerobjekt besitzt, und sie muß der Syntax dieser Methode geügen.

Für die Implementation wurden zwei elementare Klassen definiert, aus denen das *ART*-Modell besteht. Die Elemente des Modells werden aufgeteilt in *STM* -Objekt und *LTM* -Objekt. Die Neuronenfelder *F1* und *F2* mit ihren sich schnell ändernden Aktivitätszuständen werden der *STM* -Klasse zugeordnet, während die sich langsam ändernden Gewichte $z_{ij}$ und $z_{ji}$ zu einer *LTM* -Klasse gehören.

Jede der Klassen stellt neben den Platzhaltern für die Zustände ihrer Aktivität und ihrer Gewichte, auch die Methoden bereit, um zu einem *ART*-Modell verbunden zu werden.

## A.1 Das *STM* Objekt

Das *STM* -Objekt besitzt als Datenfeld Zeiger auf die Vektoren der Dimension $M$ des Neuronenfelds. Hinzu kommen die Feldkonstanten wie $a$, $b$ und $\theta$ und als dritte Gruppe, die für die Iteration nötigen Grenzbedingungen.

```
#import <objc/Object.h>
#include "ART_Func.h"
#define ART_ACT_OFF -1
@interface ART_STM :Object
{
int     M;              // field dimension
float   *P,*U,*W,       // left side
        *Q,*V,*X,       // right side
        *old_W,*old_P,  // last iteration state
        *I,*E;          // input at W and input at P

float   a,b,            // field constances
        d,              // learnrate
        theta;          // parameter of the non-linear function

int     i_min,i_max,    // iteration boundings
        act_node;       // the active node of a competionen fields
float   Act_stable;     // activity stable criterion
}
+ new;
- save;
- load;
- init:(int)Vec_dim;
- free;

// field initialisation
- set_a:(float)p1 b:(float)p2 theta:(float)p3 d:(float)p4;
- set_zero;
```

```
- set_I:(float *)I_PTR and_E:(float *)E_PTR;
- setIterationmin:(int)min max:(int)max stable:(float)stable;

// access methods to the node pointer
- (float *)get_P;
- (float *)get_U;
- (float *)get_W;
- (float *)get_Q;
- (float *)get_V;
- (float *)get_X;


// care full 'cause I and E aren't allocate in the STM object
- (float *)get_I;
- (float *)get_E;

// special for competition fields
// method return the index of maximum of U
- (int)MaxAct_with:(int)nodes;
- (int)activeNode;

// STM Object iteration and performance method
- (int)iterate;

@end
```

Die Methoden und Funktionen des *STM* Objekts teilen sich auf in drei Gruppen, die sich zum einen auf die Instanziierung des Objektes beziehen, dann auf den Zugriff auf die Zeiger der Vektoren und letztlich Performancemethoden darstellen, mit denen das *STM* -Objekt sowohl Vergleichsfelder als auch Konkurenzfelder modellieren kann.

Die Performancemethoden sollen an dieser Stelle herausgestellt werden. Das *F1* Feld des *ART–2*-Modells ist ein Vergleichsfeld, bei dem die Zustände der Aktivitäten in dem Feld iterativ stabilisiert werden. Die Eingangsvektoren des Feldes repräsentieren die Zeiger `I` und `E` . Die Funktion `(int)iterate` führt die Iteration, wie sie in Abschnitt 3.1 (Seite 26) beschrieben ist aus und liefert die Anzahl der benötigten Iterationsschritte an die aufrufende Instanz zurück.

```
// STM Object iteration and performance method
- (int)iterate
{
     int i,cntr=0;
     float ptmp,wtmp,vtmp;
     float v_d = 1.0;

     do {
       // Pass I of the iteration
       for (i=0; i<M; i++)  {
         old_W[i]=W[i];
```

```
        old_P[i]=P[i];
        W[i]=I[i]+a*U[i];
        P[i]=U[i]+d*E[i];
        }

      ptmp=L2_norm(P,M);
      wtmp=L2_norm(W,M);

      // Pass II of the iteration
      for (i=0; i<M; i++)  {
        Q[i]=P[i]/ptmp;
        X[i]=W[i]/wtmp;
        V[i]=fgross(X[i],theta)+b*fgross(Q[i],theta);
        }

      vtmp=L2_norm(V,M);

      // Pass III of the iteration
      for (i=0; i<M; i++)
        U[i]=V[i]/vtmp;

      // Check for stability by means of the absolute vectordiff.
      v_d=vec_diff(old_W,W,M)+vec_diff(old_P,P,M);

      // Increment the iterationcounter
      cntr++;
      }
    // Do iterate again
    while (
      // 1. W and P aren't stable and iteration didn't excced i_max
      ( v_d > Act_stable && cntr < i_max )
      // 2. iterations are less then i_min
      || cntr < i_min
    );
    return (int) cntr;
}
```

Das *F2* Feld wird ebenfalls aus der *STM* -Klasse definiert. Nur ist die Performancemethode hier nicht die eines Vergleichsfeldes, sondern die eines Konkurrenzfeldes. Es erhält nur dasjenige Neuron maximaler Aktivität einen signifikanten Ausgangswert.

```
// Special F2 competition method retrun the index of maximum of U
- (int)MaxAct_with:(int)nodes;
{
     int i;
     float vec_max=0.0;
     act_node=ART_ACT_OFF;
     for (i=0; i<nodes; i++)
       vec_max = (vec_max < U[i] ? U[i] : vec_max);
     if (vec_max == 0.0) return (act_node);
     for (i=0; i<nodes && U[i]<vec_max ; i++);
```

```
     act_node=i;
     return (act_node) ;
}
```

Der Funktion `(int)MaxAct_with:(int)nodes` wird als Parameter die Anzahl der etablierten Neuronen mit `(int)nodes` übergeben und die Funktion gibt als Antwort den Index des Elements des Vektors `U` zurück. Intern wird die Index noch in der Variablen `act_node` geführt. Ist im Vektor `U` keine Aktivität $u_i > 0$ wird die Definition `ART_ACT_OFF` zurück gegeben.

## A.2 Das *LTM* Objekt

Mit der *LTM* -Klasse werden die Gewichte $z_{ij}$ und $z_{ji}$ in den Zeigern `BU` und `TD` verwaltet und gemäß der Lerngleichungen 3.1 und 3.2 (Seite 25) verändert. Außerdem besitzt das *LTM* -Objekt das Ausschlußregister `Avail`, in dem vermerkt ist, ob die Erwartung eines Neurons $J$ aus *F2* noch für die Suche nach geeigneter Klassifizierung verfügbar ist. In der Variablen `estab` wird die Anzahl der etablierten Neuronen verwaltet.

```
import "ART_Func.h"

#define ART_LTM_ASCII 0
#define ART_LTM_BIN 1

#define LTM_NODE_RESET 0
#define LTM_NODE_AVAIL 1

@interface ART_LTM : Object
{
     int   M,N;           // field dimensions
     float **BU,**TD,     // learning arrays
           *oldTD,*oldBU, // buffer arrays
           d,h;           // learning rate, integation stepsize
     int   *Avail,        // notation of reset nodes
           estab;         // acount of established nodes
}
+ new;
- saveinto:(NXStream *)stream as:(int)t;
- loadfrom:(NXStream *)stream;
- init:(int)M_dim of:(int)N_dim;
- free;

// set the central parameter
- set_d:(float)learnrate h:(float)stepsize;
- zero_template:(int)template with:(float)BU_scale;
- set_zero_with:(float)BU_scale;
```

```
// transform a pattern from F1 to F2, seems like a correlation
- (float *)TD_of:(int)j;
- (float *)BU_of:(int)i;
- set:(float *)Vec of:(int)dim to_TD_of:(int)j;
- set:(float *)Vec of:(int)dim to_BU_of:(int)i;
- (int)korrelate_with:(float *)P to:(float *)X;


// managing the flag array and established nodes
- (int)newPattern;
- (int)resetNode:(int)j;
- (BOOL)NodeAvail:(int)i;
- (int)estabNodes;
- setestabNodes:(int)newestab;

// declaire a node as stable as it's norm excceds the thresh
- (BOOL)is:(int)Node stable:(float)thresh;

// performing the learning facility of the LTM array
- (float)learnPattern:(float *)Learn_Vec toNode:(int)F2_on;
@end
```

Die wichtigsten Methoden des *LTM* -Objekts sind
`(int)korrelate_with:(float *)P to:(float *)X` und
`(float)learnPattern:(float *)Learn_Vec toNode:(int)F2_on`.
Mit der `(int)korrelate_with` Funktion wird der Vektor `P` auf den Vektor `X` transformiert. In dem verschalteten *ART*-Modell ist `P` die Aktivität aus *F1* und `X` die des Konkurenzfeldes *F2* . Als Antwort wird die Anzahl der etablierten Erwartungen an den Aufrufer der Methode zurückgegeben.

```
// transform a pattern from F1 to F2, seems like a correlation
- (int)korrelate_with:(float *)P to:(float *)X
{
    int i,j;
    for (j=0; j < estab; j++) {
      X[j]=0.0;
      if ( Avail[j] )
        for (i=0; i<M; i++)
  X[j] += BU[i][j]*P[i];
      }
     return estab-1;
}
```

Ist ein Neuron in *F2* aktiv und die Erwartung des Neurons eine gültige Klassifizierung müssen die Gewichte gemäß der Lerngleichungen verändert werden. Die Funktion `RungeKutta` vollzieht die numerische Integration der Lerngleichung nach der Formel von Runge-Kutta mit der Schrittweite `h`. Die absolute Differenz zwischen den neu bestimmten Gewichten und ihren alten Werten wird mit `BUdiff+TDdiff` an den Aufrufer der Methode zurückgegeben, um eventuell mit dem Wert der Veränderung den Ablauf des Systems zu steuern. Ist `F2_on` ein noch unabhängiges Neuron muß die Variable `estab` inkrementiert werden.

```
// perform the learning facility of the LTM array
- (float)learnPattern:(float *)Learn_Vec toNode:(int)F2_on
{
     int i;
     float BUdiff=0.0,TDdiff=0.0;

     if (F2_on >= (estab-1) ) estab++;
     for (i=0; i<M; i++) {
       oldBU[i]=BU[i][F2_on];
       oldTD[i]=TD[F2_on][i];

       BU[i][F2_on]=RungeKutta(BU[i][F2_on],Learn_Vec[i],d,h);
       TD[F2_on][i]=RungeKutta(TD[F2_on][i],Learn_Vec[i],d,h);

       BUdiff += fabs(oldBU[i]-BU[i][F2_on]);
       }
     TDdiff=vec_diff(TD[F2_on],oldTD,M);
     return (BUdiff+TDdiff);
}
```

## A.3 Instanzen

Die Objektklasse `ART2_Basic` ist das zentrale Objekt des *ART–2*-Modells. Eine Instanz des `ART2_Basic`-Objekts stellt ein *ART–2*-Modul dar. Mit diesem Modul werden die nötigen Instanzen der *STM* - und *LTM* -Klassen gebildet und die grundlegenden funktionalen Verknüpfungen zwischen den Feldern als Methoden bereitgestellt (Abbildung A.1).

Die Methoden der `ART2_Basic`-Klasse fassen die Mechanismen des Ablaufs einer Klassifikation so zusammen, daß sie Schritt für Schritt abgearbeitet werden können und für die Simulation des Standard-*ART–2*-Modells und der Erweiterung verwendet werden können :

```
// set the input pattern and initialize the STM's
- setPattern:(float *)In;

// preprocessing the input returning the iterations of F0
- (int)preproc;

// choose the winner of F2 (F2 has no activity)
- (int)chooseF2;

// perform F1 with the preprocessed an it's template
-  compair;
```

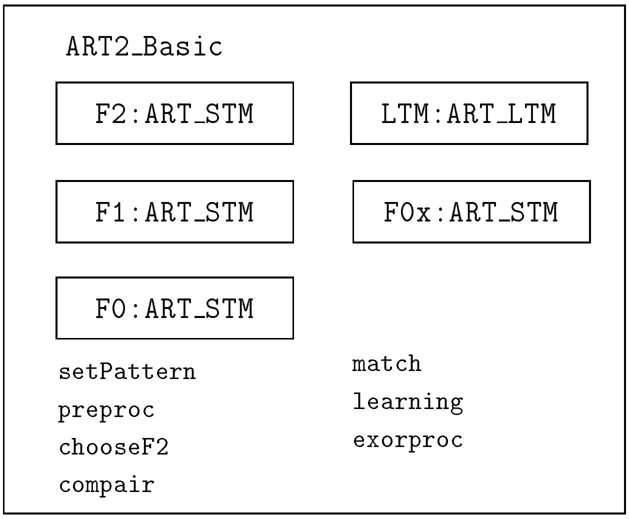


Abbildung A.1: *das* `ART2_Basic`-*Objekt*

```
// boolean for goood match
- (BOOL)match;

// perform learning and returns the stable criterion
- (float)learning;

// perform the exor processing returns the type of set
- (int)exorproc;
```

Hier sei nun beispielsweise der Ablauf der Benutzung der Methoden aufgelistet. `ART2Modul` ist ein Zeiger auf eine Instanz der Klasse `ART2_Basic`, und alle nötigen Initialisierungen seien erfolgt. Die Beschreibung stellt eine Klassifizierung nach dem Standard-*ART–2*-Modell dar, bei der die *LTM* -Gewichte nur die definierte Schrittweite `h` integriert werden (siehe auch Seite 5).

```
- (int)ART2std:(float *)Input;
{
     int F2_node;
     [ART2Modul setPattern:Input];
     [ART2Modul preproc];
     do {
       F2_node=[ART2Modul cooseF2];
       [ART2Modul compair];
       }
     while( ![ART2Modul match] );
     [ART2Modul learning];
```

```
    return F2_node;
}
```

Eine Klassifizierung mit den besprochen Erweiterungen bedarf der Zusatzmethode `(int)exorproc`, die darüber Auskunft gibt, ob es sich bei dem an *F1* anstehenden Vektor um eine Ober,Teil oder Mischmenge handelt. Auch hier ist die Instanz soweit initialisiert, daß nur noch der Ablauf der der Klassifizierung dargestellt ist.

```
- ART2ext:(float *)Input;
{
    int F2_node;
    int set_type;
    [ART2Modul setPattern:Input];
    [ART2Modul preproc];
    do {
      do {
        F2_node=[ART2Modul cooseF2];
        [ART2Modul compair];
        set_type=[ART2Modul exorproc];
        }
      while( ![ART2Modul match] );
      [ART2Modul learning];
      if (set_type==ART_SUP)
        [ART2Modul setExorPattern];
      else break;
      }
    while (set_type==ART_SUP);
    return
}
```

Die `match` Methode von `ART2_Basic` ist so aufgebaut, daß sie mit einer `art2_mode` Variablen aus `ART2_Basic` über den Modus informiert ist, also im erweiterten *ART–2*-Modell situationsabhängig die Aufmerksamkeit zwischen $\rho_{std}$ und $\rho_{ext}$ ändert (siehe auch Gl. 2.5 Seite 12 ).

Nach Beendigung der Klassifizierung sind in einem, von `ART2_Basic` bereitgestellten Vektor die von dem Eingangsmuster $k$ aktivierten Neuronen aus *F2* dokumentiert. Dieser Vektor entspricht der geordneten Reihe $g(k)$ :

$$k \longmapsto \{j_1, j_2, \ldots, j_{n_k}\} = g(k),$$

wie in Abschnitt 2.2 (Seite 13) und ist das Ergebnis der verteilten Klassifizierung durch orthogonale Musteranteile.

## A.4 Applikation

Für die entstandene Applikation wurden weiter Objekte gebildet, die für die Bedienung der Modulparameter und die Visualisierung der Zustände im *ART*-Modul zuständig sind.

`ARTDoc`: Im Zentrum steht das `ARTDoc`, das als Schnittstelle zwischen der *Application kit software* und dem `ART2_Basic`-Objekt dient. In der Applikation können mehrere Instanzen von `ARTDoc` kontrolliert werden, und sie sind über Fenster dem Benutzer zugänglich. In diesem Fenster können die Parameter der dazugehörigen `ART2_Basic` Instanz verändert werden.

`Controller`: Ein `Controller`-Objekt wird als das Objekt instantiiert,das stellvertretend für die `NXApp` die benutzererzeugten Ereignisse interpretiert und ausführt. Gleichzeitig kontrolliert das `Controller`-Objekt die Kommunikation in der Applikation.

`Animator`: Das `Animator`-Objekt ist ein passives Objekt, das von dem momentan aktiv gesetzten `ARTDoc` die Zustände der `ART2_Basic`-Instanz erhält und diese dem Benutzer visualisiert (siehe auch Abbildung 5.2 Seite 37 und Abbildung 5.3 Seite 38 ).

`Pattern`: Mit dem `Pattern`-Objekt wird eine Musterquelle bereitgestellt, aus der alle `ARTDoc`-Objekte ihre Eingangsmuster beziehen. Das `Pattern`-Objekt verwaltet die Eingabe der Muster über eine Datei.

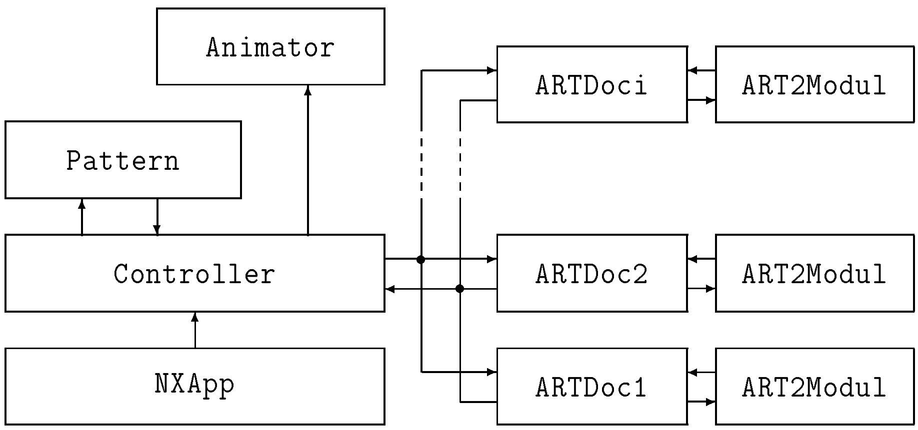


Abbildung A.2: *Aufbau der Applikation*

Für die Bedienung wurden mit dem *Interface Builder* Objekte der Ein- und Ausgabe zusammengestellt und mit den dazu korrespondierenden Methoden der Objekte in der Applikation verbunden (siehe Abschnitt 5). Diese Verknüpfung der Oberflächenelemente mit den Objekten der Applikation (Abbildung A.2) wurden am Bildschirm graphisch editiert und als Datei abgelegt werden sie zur Laufzeit zu der Applikation dazugeladen.

## A.5 Schnittstellen

Für die Datenein- und ausgabe stehen zwei Dateiformate bereit. Für eine größtmögliche Kompatibilität sind diese Dateien reine Textdateien, nachfolgender Formatierung.

**Eingabe** Um eine Musterquelle zu laden zu können, muß die erste Zeile der Datei mit der Namenserweiterung `pattern` wie folgt formatiert sein :

```
ascii [Anzahl] [Dimension]
```

Danach folgen zeilenweise die Mustervektoren. Die einzelen Elemente sind in `ASCII`-Dezimaldarstellung durch ein Leerzeichen voneinander getrennt.

**Ausgabe** Über eine Datei mit dem Name des zugehörigen Dokuments und der Namenserweiterung `LTMdat` erfolgt die Ausgabe der Gewichte $z_{ij}$ und $z_{ji}$ bzw. der Zeigerinhalte `BU` und `TD` (siehe A.2). Diese Ausgabe erscheint in zwei Abschnitten:

```
TD [Anzahl] [Dimension]
```

und

```
BU [Anzahl] [Dimension]
```

Nach jeder Einleitung erfolgt die zeilenweise Ausgabe der Vektoren. Die Darstellung der einzelnen Elemente eine Vektors erscheint ebenso wie bei der Eingabe in `ASCII`-Dezimaldarstellung mit einem trennenden Leerzeichen.

Zusätzlich kann ein verändertes Dokument gespeichert und geladen werden. Die Namenserweiterung der Datei ist `art`. In dieser Datei sind sowohl die Gewichte als auch die Parameter des entsprechenden Dokuments in binärer Form gespeichert und nicht für den direkten Zugriff durch den Benutzer gedacht.

## A.6 Zusatz

Für die, in dieser Arbeit abgebildeten Darstellungen der Klassifizierungsabläufe reellwertiger Mustervektoren (3.6 auf Seite 33 und 3.7 auf Seite 34) wurden der *Post Script* Rumpf einer MatLab Graphik erweitert und mit den Ausgabedaten des Klassifizierungsablaufs gefüllt. Die Einbindung einer Ausgabe in diesem Format ist leider nicht realisiert, da ein einwandfreies Funktionieren einer solchen Ausgabe eine intensive *Post Script* Programmierung bedingte.