Fehlererkennung und Fehlerkorrektur in Codes

Blockcodes und Hamming–Abstand

Untersuchungen zu Codierungen von Informationen, die über einen Nachrichtenkanal übertragen werden sollen, konzentrieren sich in der Regel auf einen der folgenden Aspekte:

1. Kompaktheit: Die codierte Information sollte möglichst kurz sein.
2. Geheimhaltung: Die im Kanal übertragene Information sollte ohne Kenntnis eines Schlüssels nicht (oder nur sehr schwer) zu decodieren sein.
3. Fehlererkennung und Fehlerkorrektur: Treten bei der Übertragung vereinzelte Fehler auf sollte das erkannt werden und gegebenenfalls korrigiert werden können.

Es ist klar, dass man nicht alle Aspekte in einer Codierung berücksichtigen kann, denn die Aspekte 1 und 3 stehen sich diametral gegenüber: Um eine Information kompakt zu repräsentieren, muss man auf Redundanz verzichten, aber andererseits ist Redundanz notwendig, um Übertragungsfehler erkennen.

Wichtige Themen im Zusammenhang mit dem ersten Aspekt sind statistische Datenanalyse, Stringmatching–Algorithmen und Huffman–Codes.
Der zweite Aspekt wird nicht der Codierungstheorie zugeordnet, sondern begründet ein eigenständiges Forschungsgebiet, die Kryptographie.
Im Folgenden werden wir uns nur mit dem dritten Aspekt beschäftigen.

Definition: Sei A eine endliche Menge von zu codierenden Symbolen (Informationen) und Q ein q–elementiges Alphabet, dass sogenannte Kanalalphabet . Eine injektive Funktion $\varphi : A \longrightarrow Q^*$ wird *Codierung* von A genannt. Das Bild der Funktion φ ist der zugehörige *Code* $C = \mathrm{Im}(\varphi)$, die Elemente von C werden *Codewörter* genannt. Der Code C ist ein *Blockcode* der Blocklänge n, wenn alle Codewörter aus C die Länge n haben.

Jeder Blockcode C der Blocklänge n kann damit auch als Teilmenge von Q^n und seine Elemente als n–Tupel beschrieben werden. Zur besseren Lesbarkeit vereinbaren wir die folgenden Bezeichnungen. Mit $\vec{u}, \vec{v}, \vec{w}$ werden beliebige Tupel (Vektoren) aus Q^n benannt. Wenn betont werden soll, dass es sich bei dem Tupel um ein Codewort handelt, verwenden wir die Bezeichnungen $\vec{c}, \vec{c'}$ oder Ähnliches.
Definition: Der Hamming–Abstand zwischen zwei n–Tupeln $\vec{v} = (v_1, \ldots, v_n)$ und $\vec{w} = (w_1, \ldots, w_n)$ wird mit $d_H(\vec{v}, \vec{w})$ oder kurz mit $d(\vec{v}, \vec{w})$ bezeichnet. Er ist bestimmt durch die Anzahl der Stellen an denen sich die zwei Tupel unterscheiden, d.h.

$$d_H(\vec{v}, \vec{w}) = d(\vec{v}, \vec{w}) = |\{i \,|\, 1 \leq i \leq n \wedge v_i \neq w_i\}|$$

Fehlererkennende und fehlerkorrigierende Codes

Zur Fehlerekennung und Fehlerkorrektur muss einschränkend gesagt werden, dass hier nur Fehler der Art betrachtet werden, dass einzelne Einträge eines Tupels, also Symbole aus Q, bei der Übertragung verändert werden. Der Fall von fehlenden oder fälschlicherweise eingeschobenen Einträgen wird nicht berücksichtigt. Damit ist der Hamming–Abstand ein geeignetes Beschreibungsmittel für die Fehleranzahl:

Wenn bei der Übertragung eines Codeworts $\vec{c}$ genau k Fehler auftreten, dann ist k der Hamming–Abstand zwischen $\vec{c}$ und dem empfangenen Tupel $\vec{v}$. Die folgende Definition dient zur Beschreibung aller Tupel mit höchstens k Fehlern.

Definition: Die Kugel mit Radius $k \in \mathbb{N}$ in Q^n um einen Punkt $\vec{u}$ besteht aus allen Punkten $\vec{v}$ deren Hamming–Abstand zu $\vec{u}$ kleiner oder gleich k ist:

$$B_k(\vec{u}) = \{\, \vec{v} \in Q^n \mid d(\vec{v}, \vec{u}) \leq k \,\}$$

Lemma: Der Hamming–Abstand ist eine Metrik auf Q^n, d.h. für beliebige $\vec{u}, \vec{v}, \vec{w} \in Q^n$ sind die folgenden drei Eigenschaften erfüllt:

1. $d(\vec{u}, \vec{v}) \geq 0$ und $d(\vec{u}, \vec{v}) = 0 \iff \vec{u} = \vec{v}$
2. $d(\vec{u}, \vec{v}) = d(\vec{v}, \vec{u})$ (Symmetrie)
3. $d(\vec{u}, \vec{v}) + d(\vec{v}, \vec{w}) \geq d(\vec{u}, \vec{w})$ (Dreiecksungleichung)

Definition: Der Minimalabstand eines Codes $C \subseteq Q^n$ wird mit $d(C)$ bezeichnet und ist der kleinste Abstand zwischen zwei Codewörtern aus C, d.h.

$$d(C) = \min \{\, d(\vec{c}, \vec{c'}) \mid \vec{c}, \vec{c'} \in C \text{ und } \vec{c} \neq \vec{c'} \}$$

Wie wir sehen werden, ist der Minimalabstand ein entscheidender Schlüssel zur Charakterisierung von Fehlererkennungs– und Fehlerkorrektur–Eigenschaften eines Codes.

Definition: Ein Code $C \subseteq Q^n$ ist k–fehlererkennend, wenn für jedes Codewort $\vec{c} \in C$ jedes Tupel $\vec{v} \in B_k(\vec{c}) \setminus \{\vec{c}\}$ (das sich also von $\vec{c}$ an mindestens einer und höchstens k Stellen unterscheidet) nicht in C liegt und damit als fehlerhaft erkannt wird.
Der Code C ist k–fehlerkorrigierend, wenn für jedes Codewort $\vec{c} \in C$ und für jedes Tupel $\vec{v} \in B_k(\vec{c})$ (das sich also von $\vec{c}$ an höchstens k Stellen unterscheidet) $\vec{c}$ das eindeutig nächste Codewort zu $\vec{v}$ ist und damit die $\leq k$ Fehler in $\vec{v}$ durch Suche nach dem nächsten Codewort korrigiert werden können.

Da in dieser Definition der Zusammenhang zwischen Fehlern und Hamming–Abstand schon sehr deutlich hervorgehoben wurde, ist die folgende Charakterisierung von fehlererkennenden und fehlerkorrigierenden Codes offensichtlich.

Satz: Für einen Code $C \subseteq Q^n$ gelten die folgenden Äquivalenzen:

- C ist k–fehlererkennend $\iff \forall\, \vec{c} \in C \quad B_k(\vec{c}) \cap C = \{\vec{c}\}$
  $\iff d(C) \geq k+1$

- C ist k–fehlerkorrigierend $\iff \forall\, \vec{c}, \vec{c'} \in C \quad \Big(\vec{c} \neq \vec{c'} \Rightarrow B_k(\vec{c}) \cap B_k(\vec{c'}) = \emptyset\Big)$
  $\iff d(C) \geq 2k+1$

Beispiele: Die folgenden binären Codierungen (d.h. $Q = \{0,1\}$) beziehen sich auf die Situation, dass die zu codierende Menge A bereits die Form Q^m hat, also aus binären m–Tupeln besteht, die bei der Codierung durch Hinzufügung redundanter Informationen in n–Tupel umgewandelt werden. Die zu codierenden Tupel haben demnach immer die Form $\vec{v} = (v_1, \ldots, v_m)$ bzw. $\vec{w} = (w_1, \ldots, w_m)$.

1. Doppelcodierung: $\varphi_2 : Q^m \longrightarrow Q^{2m}$ $(n = 2m)$, wobei

   $$\varphi_2(v_1, \ldots, v_m) = (v_1, \ldots, v_m, v_1, \ldots, v_m)$$

   Der zugehörige Code $C_2 = \mathrm{Im}(\varphi_2)$ hat den Minimalabstand 2, denn $d(\varphi_2(\vec{v}), \varphi_2(\vec{w})) = 2 \cdot d(\vec{v}, \vec{w})$. Damit ist C_2 1–fehlererkennend.

2. Codierung mit Paritätsbit: $\varphi_{par} : Q^m \longrightarrow Q^{m+1}$ $(n = m+1)$, wobei

   $$\varphi_{par}(v_1, \ldots, v_m) = (v_1, \ldots, v_m, p) \quad \text{mit} \quad p = (v_1 + \ldots + v_m) \bmod 2$$

   Auch hier hat der zugehörige Code $C_{par} = \mathrm{Im}(\varphi_{par})$ hat den Minimalabstand 2, denn im Fall $d(\vec{v}, \vec{w}) \geq 2$ überträgt sich die Ungleichung automatisch auf die Codewörter und im Fall $d(\vec{v}, \vec{w}) = 1$ müssen die Paritätsbits für $\vec{v}$ und $\vec{w}$ verschieden sein, woraus sich $d(\varphi_p(\vec{v}), \varphi_p(\vec{w})) = 2$ ergibt. Damit ist der Code C_{par} 1–fehlererkennend.

3. Um den Minimalabstand zu vergrößern kann man von der Doppelcodierung zur dreifachen oder k–fachen Codierung übergehen und erreicht damit $d(C_k) = k$.

4. Eine etwas bessere Alternative zur Dreifachcodierung ist die Doppelcodierung mit Paritätsbit $\varphi_{2,par} : Q^m \longrightarrow Q^{2m+1}$, wobei

   $$\varphi_{2,par}(v_1, \ldots, v_m) = (v_1, \ldots, v_m, v_1, \ldots, v_m, p) \quad \text{mit} \quad p = (v_1 + \ldots + v_m) \bmod 2$$

   Der Minimalabstand $d(C_{2,par}) = 3$ ergibt sich durch eine einfache Fallunterscheidung.

5. Leider erreicht man durch mehrfache Paritätsbits über der gleichen Grundmenge keine Verbesserung des Minimalabstands. Sinnvoller ist es, mehrere Paritätsbits über verschiedenen (geeignet gewählten!) Teilmengen der Eingabebits zu bilden. Ein Beispiel dafür sind die sogenannten Kreuzsicherungscodes, bei denen m eine

Quadratzahl oder zumindest Produkt aus zwei annähernd gleichen ganzen Zahlen sein sollte.
Sei $m = k^2$ dann verwendet man für den Kreuzsicherungscode $\varphi_{kr} : Q^m \longrightarrow Q^{m+2k}$ insgesamt $2k = 2\sqrt{m}$ Paritätsbits. Dazu trägt man die k^2 Bits des Eingabetupels $\vec{v}$ Zeile für Zeile in eine $k \times k$ Matrix ein und bildet alle Zeilenparitätsbits $p_1, \ldots p_k$ und alle Spaltenparitätsbits $p'_1, \ldots p'_k$ und definiert:

$$\varphi_{kr}(v_1, \ldots, v_m) = (v_1, \ldots, v_m, p_1, \ldots, p_k, p'_1, \ldots p'_k)$$

Der Minimalabstand $d(C_{kr}) = 3$ ergibt sich aus der folgenden Fallunterscheidung:

$d(\vec{v}, \vec{w}) \geq 3$		$d\left(\varphi_{kr}(\vec{v}), \varphi_{kr}(\vec{w})\right) \geq 3$
$d(\vec{v}, \vec{w}) = 2$	Unterschiedliche Stellen in einer Zeile	$d\left(\varphi_{kr}(\vec{v}), \varphi_{kr}(\vec{w})\right) = 2 + 2 = 4$ ($2 + 2$ Spaltenparitätsbits)
	Unterschiedliche Stellen in einer Spalte	$d\left(\varphi_{kr}(\vec{v}), \varphi_{kr}(\vec{w})\right) = 2 + 2 = 4$ ($2 + 2$ Zeilenparitätsbits)
	sonst	$d\left(\varphi_{kr}(\vec{v}), \varphi_{kr}(\vec{w})\right) = 2 + 2 + 2 = 6$ ($2 + 2$ Spaltenpar.-bits $+ 2$ Zeilenenpar.-bits)
$d(\vec{v}, \vec{w}) = 1$		$d\left(\varphi_{kr}(\vec{v}), \varphi_{kr}(\vec{w})\right) = 1 + 1 + 1 = 3$ ($1 + 1$ Spaltenpar.-bit $+ 1$ Zeilenenpar.-bit)

Hamming–Code

Der Kreuzsicherungscode benötigt zur Codierung von $A = \{0, 1\}^4$ vier zusätzliche Paritätsbits, um den Minimalabstand 3 zu realisieren. Mit etwas Knobelei kommt man darauf, dass auch schon drei zusätzliche Paritätsbits ausreichen, wie der folgende Code zeigt, der nach Hamming benannt ist:

$$\begin{aligned} \text{Sei} \quad \vec{v} &= (v_1, v_2, v_3, v_4). \\ p_1 &:= (v_2 + v_3 + v_4) \bmod 2 \\ p_2 &:= (v_1 + v_3 + v_4) \bmod 2 \\ p_3 &:= (v_1 + v_2 + v_4) \bmod 2 \\ \varphi_{ham}(\vec{v}) &:= (v_1, v_2, v_3, v_4, p_1, p_2, p_3) \end{aligned}$$

Die Analyse, dass der Minimalabstand dieses Hamming–Codes 3 ist, erfolgt wieder durch eine Fallunterscheidung:
Ist $d(\vec{v}, \vec{w}) \geq 3$, dann überträgt sich diese Ungleichung auch auf die Codewörter.
Im Fall $d(\vec{v}, \vec{w}) = 2$ zeigt sich, dass entweder ein Paritätsbit verschieden ist (wenn sich

$\vec{v}$ und $\vec{w}$ an der vierten Stelle unterscheiden) oder sogar zwei (wenn $\vec{v}$ und $\vec{w}$ an der vierten Stelle gleich sind)
Im Fall $d(\vec{v}, \vec{w}) = 1$ sind entweder zwei Paritätsbits verschieden (wenn $\vec{v}$ und $\vec{w}$ an der vierten Stelle gleich sind) oder alle drei (wenn sich $\vec{v}$ und $\vec{w}$ an der vierten Stelle unterscheiden).

Informationsrate

Definition: Die Informationsrate eines Codes $C \subseteq \{0,1\}^n$ ist der Quotient $\frac{\log_2 |C|}{n}$ oder allgemein für $C \subseteq Q^n$ der Quotient $\frac{\log_q |C|}{n}$, wobei $q = |Q|$. Damit beschreibt die Informationsrate das Längenverhältnis der Informationswörter zu den Codewörtern.

Für die besprochenen Beispiele ergeben sich die folgenden Informationsraten:

- Die Informationsrate der Doppelcodierung ist $\frac{1}{2}$ und die Informationsrate der Dreifachcodierung ist $\frac{1}{3}$.
- Die Informationsrate der Doppelcodierung mit Paritätsbit ist $\frac{m}{2m+1} < \frac{1}{2}$.
- Die Informationsrate des Kreuzsicherungscodes für $A = \{0,1\}^4$ ist $\frac{1}{2}$ und der besprochene Hamming–Code hat eine bessere Informationsrate von $\frac{4}{7} > \frac{1}{2}$.
- Die Informationsrate des Kreuzsicherungscodes für $A = \{0,1\}^{k^2}$ ist $\frac{k^2}{k^2+2k} = 1 - \frac{2}{k}$, d.h. mit ausreichend großen Werten von k man kann sich beliebig dicht an die 1 annähern.

Der letzte Punkt muss aber auch kritisch betrachtet werden: Es ist zwar möglich, mit großem k einen 1-fehlerkorrigierenden Code mit sehr guter Informationrate zu konstruieren, aber dann darf auf k^2 Bits auch nur ein Fehler auftreten. Um also zu einer weniger oberflächlichen Bewertung zu kommen, müsste man die Informationsrate mit der Fehlerwahrscheinlichkeit in Zusammenhang bringen.