Informationstheorie Zusammenfassung

Fabian Hahn, Dino Wernli

15. März 2009

Ergänzungen zur Wahrscheinlichkeitszusammenfassung

Verbundwahrscheinlichkeit

Die *Verbundwahrscheinlichkeit* ist eine andere Bezeichnung für die gemeinsame Gewichtsfunktion mehrerer gemeinsam verteilter Zufallsvariablen. Also zum Beispiel:

$$P[X_1 = x_1, X_2 = x_2, X_3 = x_3, \ldots, X_n = x_n] \quad (1)$$

$$\text{Kurzschreibweise: } P[X_1, X_2, X_3, \ldots, X_n] \quad (2)$$

Formale allgemeine Pfadregel

Verbundwahrscheinlichkeiten können mit Hilfe bedingter gemeinsamer Wahrscheinlichkeiten entwickelt werden:

$$P[X_1, \ldots, X_n] = P[X_1, \ldots, X_{n-1}] \cdot P[X_n | X_1, \ldots X_{n-1}] \quad (3)$$

$$= \ldots = \prod_{i=1}^{n} P[X_i | X_1, \ldots, X_{i-1}] \quad (4)$$

Dies entspricht gerade dem Ausmultiplizieren eines Pfades in einem Wahrscheinlichkeitsbaum. Insofern stellt diese Entwicklung auch die Formalisierung der *allgemeinen Pfadregel* dar.

Marginalisierung

Die *Marginalisierung* ist eine andere Bezeichnung für das Wegsummieren oder -integrieren einer oder mehrerer Zufallsvariablen einer Verteilung, wenn man sich für die Verteilung unabhängig von diesen interessiert:

$$P[X_1 = x_1] = \sum_{x_2 \in \mathcal{W}(X_2)} P[X_1 = x_1, X_2 = x_2] \quad (5)$$

Bedingte Marginalisierung

Anstatt der *Verbundwahrscheinlichkeit* kann man durch Anwendung von (4) zum *Marginalisieren* auch bedingte Wahrscheinlichkeiten verwenden:

$$P[X_1 = x_1] = \sum_{x_2} P[X_1 = x_1 | X_2 = x_2] \cdot P[X_2 = x_2] \quad (6)$$

Bayes-Netze

Bayes-Netze sind eine Möglichkeit, Wahrscheinlichkeitssysteme mit gekoppelten bedingten Zufallsvariablen zu modellieren. Sie sind nützlich, da man bedingte Wahrscheinlichkeiten einer Zufallsvariable auf die im Graphen direkt mit der Zufallsvariablen verbundenen Bedingungen reduzieren kann.
Abbildung 1 zeigt ein einfaches Bayes-Netz. In diesem gilt zum Beispiel:

$$P[D|A, B, C] = P[D|B, C] \quad (7)$$

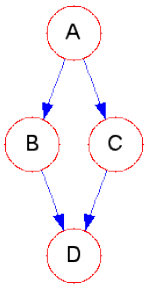

Abbildung 1: Ein einfaches Bayes-Netz

Bedingter Erwartungswert

Der *bedingte Erwartungswert* beschreibt den Durchschnitt der Auswertung einer Zufallsvariablen über viele Versuche unter der Bedingung, dass die Auswertungen anderer Zufallsvariablen bereits bekannt sind:

$$E[X|Y = y] = \sum_{x} x \cdot P[X = x | Y = y] \quad (8)$$

Marginalisierter Erwartungswert

Mit Hilfe obiger Definition lassen sich nun auch Erwartungswerte *marginalisieren*:

$$E[X] = \sum_{y} E[X|Y = y] \cdot P[Y = y] = E\left[E[X|Y]\right] \quad (9)$$

Stochastische Prozesse

Ein Wahrscheinlichkeitsexperiment, das aus einer zeitlichen Abfolge mehrerer Zufallsvariablen besteht, heisst *stochastischer Prozess*:

$$\{X_{t_1} = x_1, X_{t_2} = x_2, \ldots, X_{t_n} = x_n\} \quad (10)$$

Stationarität

Ein *stochastischer Prozess*, bei welchem die statistischen Eigenschaften (Erwartungswert, Varianz, ...) dessen Zufallsvariablen unabhängig vom betrachteten Zeitpunkt t_i sind, heisst *stationär*. Es gilt also:

$$E[X_{t_i}] = E[X_{t_j}] \quad (11)$$

$$P[X_{t_i}, X_{t_j}] = f(|i - j|) \quad (12)$$

Ergodizität

Ein *stochastischer Prozess* heisst *ergodisch*, falls dessen Zufallsvariablen zu jedem Zeitpunkt den gleichen Erwartungswert haben wie ihre durchschnittliche Auswertung über alle

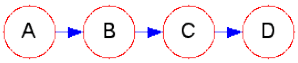

Abbildung 2: Eine einfache Markov-Kette

Zeitpunkte eines Versuchs:

$$\forall i : E[X_{t_i}] = \lim_{n\to\infty} \frac{X_{t_1} + X_{t_2} + \ldots + X_{t_n}}{n} \tag{13}$$

Intuitiv gesprochen ist es egal, ob man über Auswertungen zu einem festen Zeitpunkt mehrerer Experimente oder über die Auswertungen des *stochastischen Prozesses* eines Experiments mittelt. *Ergodische stochastische Prozesse* sind immer auch *stationär*.

Beispiel: Das n-malige Würfeln eines Würfels im Abstand von je einer Sekunde ist ein *ergodischer Prozess*: Es spielt keine Rolle, ob man die n Mal hintereinander würfelt und dann den Durchschnitt bildet, oder ob man 1000 Experimente gleichzeitig durchführt (mit 1000 Würfeln gleichzeitig würfeln) und von deren erstem Wurf den Durchschnitt nimmt.

Markov-Ketten

Ein *stochastischer Prozess* heisst *Markov-Kette* oder *Markov-Prozess der Ordnung 1*, wenn gilt:

$$P[X_n|X_1, X_2, \ldots, X_{n-1}] = P[X_n|X_{n-1}] \tag{14}$$

Die bedingte Wahrscheinlichkeit eines Kettenglieds ist also nur von seinem direkten Vorgänger abhängig. Abbildung 2 zeigt eine einfache solche Kette.
Für *Markov-Ketten* gelten die *Entropieeigenschaften* (45) und (46) sowie das *Informationsverarbeitungslemma* (47)+(48).

Übergangswahrscheinlichkeitsmatrix

Eine *Markov-Kette* wird durch eine *Übergangswahrscheinlichkeitsmatrix* beschrieben:

$$\boldsymbol{P}_{k,l} = P[X_i = x_k|X_{i-1} = x_l] \tag{15}$$

Markov-Zustandsautomaten

Eine *stationäre Markov-Kette*, deren Zufallsvariablen X_i äquivalent sind mit diskretem Wertebereich Q, kann als *Markov-Zustandsautomat* aufgefasst werden. Q ist die Zustandsmenge und die Funktion $p_X(p|q)$ beschreibt die Übergangswahrscheinlichkeit von einem Zustand in den nächsten. Abbildung 3 zeigt einen *Markov-Automaten* mit drei Zuständen. Dabei gilt für jeden Zustandsknoten q_i folgende Bedingung:

$$\sum_{k=1}^{n} p_X(q_k|q_i) = 1 \tag{16}$$

Für die Wahrscheinlichkeit eines Zustands q_i gilt:

$$p_X(q_i) = \sum_{k=1}^{n} p_X(q_i|q_k) \cdot p_X(q_k) \tag{17}$$

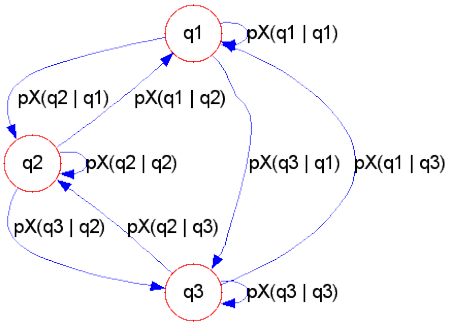

Abbildung 3: Ein Markov-Automat mit drei Zuständen

Um also diese Zustandswahrscheinlichkeiten zu bekommen, muss man das Eigenwertproblem der zugehörigen **Zustandsübergangsmatrix** lösen:

$$\boldsymbol{P}_{k,l} = p_X(q_k|q_l) \tag{18}$$

In dem Feld der k-ten Zeile und der l-ten Spalte steht also die Übergangswahrscheinlichkeit, mit welcher man von Zustand l zum Zustand k wechselt.
Die *Zustandsübergangsmatrix* ist auch in anderer Weise praktikabel: Multipliziert man sie t mal mit sich selbst, so steht im Feld der k-ten Zeile und der l-ten Spalte, mit welcher Wahrscheinlichkeit man in t Schritten vom Zustand l ausgehend im Zustand k landet. Lässt man $t \to \infty$ gehen, so gehen die Spaltenvektoren der resultierenden Matrix gegen den Eigenvektor mit den Zustandswahrscheinlichkeiten.

Entropie

Um eine Zufallsvariable mit n verschiedenen, gleichwahrscheinlichen Zuständen binär zu kodieren, benötigt man $\lceil \log_2 n \rceil = \lceil -\log_2 p_n \rceil$ Bits. Möchte man beliebige Wahrscheinlichkeitsverteilungen kodieren, so benötigt man eine allgmeinere Definition für den Informationsgehalt bzw. die Unsicherheit.
Die *Entropie* einer diskreten Wahrscheinlichkeitsverteilung $[p_1, p_2, \ldots, p_n]$ ist gegeben durch:

$$H([p_1, p_2, \ldots, p_n]) = -\sum_{i=1}^{n} p_i \cdot \log_2 p_i \tag{19}$$

Die *Entropie* einer diskreten Zufallsvariable X ist analog definiert, sie lässt sich ausserdem auch durch einen Erwartungswert ausdrücken:

$$H(X) = -\sum_{x_i} P[X = x_i] \cdot \log_2 (P[X = x_i]) \tag{20}$$

$$= E[-\log_2 (P[X])] \tag{21}$$

Binäre Entropiefunktion

Die *Entropie* einer binären Zufallsvariablen $X \sim Be(p)$ ist gegeben durch die *binäre Entropiefunktion*:

$$h(p) = -p \cdot \log_2 p - (1-p) \cdot \log_2(1-p) \tag{22}$$

$$h(0) = h(1) = 0 \tag{23}$$

$h(p)$ ist strikt konkav und besitzt ein Maximum bei $h(\frac{1}{2}) = 1$.

Entropieschranken

Sei $\chi = \{x_1, x_2, \ldots, x_n\}$ die Menge aller Zustände einer diskreten Zufallsvariable X oder auch ihr **Alphabet**. Dann gilt:

$$0 \leq H(X) \leq \log_2 |\chi| \tag{24}$$
$$H(X) = 0 \Leftrightarrow \exists! i : P[X = x_i] = 1 \tag{25}$$
$$H(X) = \log_2 |\chi| \Leftrightarrow \forall i : P[X = x_i] = \frac{1}{|\chi|} \tag{26}$$

Verbundentropie

Seien X, Y zwei diskrete Zufallsvariablen. Ihre gemeinsame Entropie oder *Verbundentropie* ist gegeben durch:

$$H(XY) = -\sum_{x_i, y_i} p_{XY}(x_i, y_i) \cdot \log_2 \left(p_{XY}(x_i, y_i)\right) \tag{27}$$
$$= E\left[-\log_2 \left(P[X, Y]\right)\right] \tag{28}$$

Für die *Verbundentropie* gilt:

$$H(X) \leq H(XY) \leq H(X) + H(Y) \tag{29}$$

Gleichheit links in (29) gilt genau dann, wenn Y durch X bereits vollständig bestimmt ist, also gilt:

$$\forall x \exists y : P[Y = y | X = x] = 1 \tag{30}$$

Gleichheit rechts in (29) gilt genau dann, wenn X und Y unabhängig sind.

Bedingte Entropie

Die *bedingte Entropie* einer Zufallsvariable X, gegeben eine andere Zufallsvariable Y, ist definiert als:

$$H(X|Y) = H(XY) - H(Y) \tag{31}$$

Die *bedingte Entropie* ist also die restliche Unsicherheit über X, wenn man Y kennt. Es gilt:

$$0 \leq H(X|Y) \leq H(X) \tag{32}$$

Analog zur *Verbundentropie* gilt linke Gleichheit, wenn X durch Y vollständig bestimmt ist, rechte Gleichheit, wenn X und Y unabhängig sind. Letztere bedeutet auch, dass zusätzliche Information die Unsicherheit niemals erhöhen kann!

Aliter: Alternativ lässt sich die *bedingte Entropie* auch wie folgt herleiten. Die **teilbedingte Entropie** ist gegeben durch:

$$H(X|Y = y) = -\sum_{x_i} p_{X|Y}(x_i, y) \cdot \log_2 \left(p_{X|Y}(x_i, y)\right) \tag{33}$$

Die *bedingte Entropie* ergibt sich nun aus dem Erwartungswert der *teilbedingten Entropien*. Damit kann die *bedingte Entropie* nun auch als Erwartungswert über alle Zustandspaare (x, y) dargestellt werden:

$$H(X|Y) = \sum_{y_i} P[Y = y_i] \cdot H(X|Y = y_i) \tag{34}$$
$$= E\left[-\log_2 \left(p_{X|Y}(X, Y)\right)\right] \tag{35}$$

Gegenseitige Information

Die *gegenseitige Information*, die eine Zufallsvariable X über eine andere Zufallsvariable Y gibt (und umgekehrt), ist gegeben durch:

$$I(X;Y) = I(Y;X) = H(X) + H(Y) - H(XY) \tag{36}$$
$$= H(X) - H(X|Y) \tag{37}$$
$$= H(Y) - H(Y|X) \tag{38}$$

In diesem Sinne ist $I(X;Y)$ auch die Reduktion der Unsicherheit über X, wenn man Y erfährt. Es gilt:

$$I(X;Y) \geq 0 \tag{39}$$

Bedingte Information

Die bedingte, gegenseitige Information, welche die Zufallsvariable X über eine andere Zufallsvariable Y gibt (und umgekehrt), gegeben eine weitere Zufallsvariable Z, ist gegeben durch:

$$I(X;Y|Z) = H(XZ) + H(YZ) - H(XYZ) - H(Z) \tag{40}$$
$$= H(X|Z) - H(X|YZ) \tag{41}$$

Damit ist $I(X;Y|Z)$ auch die Reduktion der Unsicherheit über X, wenn man Y erfährt, aber Z bereits kennt. Für die *bedingte Information* gilt:

$$I(X;Y|Z) \geq 0 \tag{42}$$

Abbildung 4 zeigt diese Zusammenhänge graphisch dargestellt. Dabei gilt für das Zentralglied:

$$R(X;Y;Z) = H(X) + H(Y) + H(Z) - H(XY) \tag{43}$$
$$- H(XZ) - H(YZ) + H(XYZ)$$

Achtung: $R(X;Y;Z)$ kann auch negativ werden, wenn Z eine stärkere Reduktion von $H(X|Y)$ als von $H(X)$ bewirkt, also gilt:

$$H(X|Y) - H(X|YZ) > H(X) - H(X|Z) \tag{44}$$

Entropie in Markov-Ketten

Erweitern wir die *Markov-Kette* aus Abbildung 2 um Zufallsvariablen, welche die Zustandsübergänge modellieren. Die neue Kette ist in Abbildung 5 dargestellt: So wird zum Beispiel X durch B in Y überführt. Ihre Markov-Eigenschaft lässt sich wie folgt in *Entropien* ausdrücken:

$$H(Z|XY) = H(Z|Y) \tag{45}$$
$$I(X;Z|Y) = 0 \tag{46}$$

Informationsverarbeitungslemma

Sei eine *Markov-Kette* mit Zustandsübergängen wie in Abbildung 5 gegeben. Dann gelten folgende zwei Ungleichungen:

$$I(X;Z) \leq I(Y;Z) \tag{47}$$
$$I(X;Z) \leq I(X;Y) \tag{48}$$

Intuitiv gesprochen bedeutet dies, dass nichts die Information, welche Y über X (und umgekehrt) bzw. Z über Y enthält, erhöhen kann.

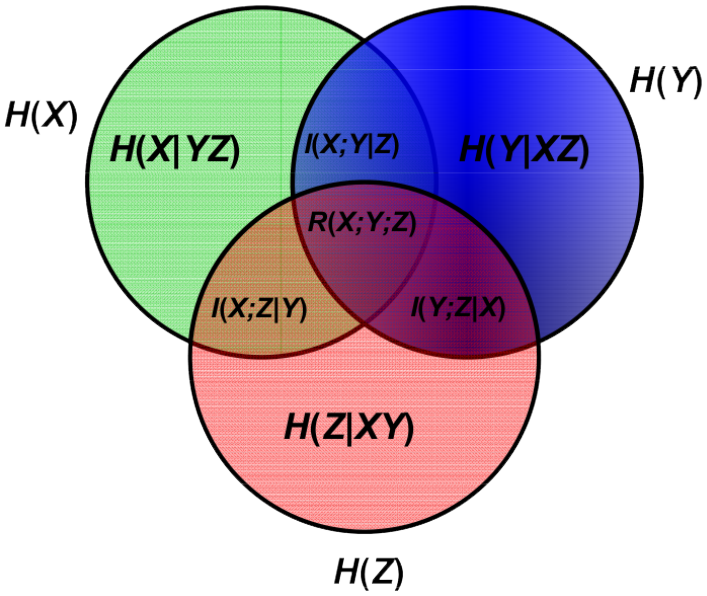

Abbildung 4: Entropiediagramm

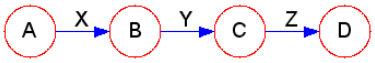

Abbildung 5: Eine Markov-Kette mit Zustandsübergängen

Kettenregel für Entropien

Obige Definitionen lassen sich auch auf beliebige Kopplungen vieler Zufallsvariablen X_i anwenden:

$$H(X_1 X_2 \dots X_n) \leq \sum_{i=1}^{n} H(X_i) \quad (49)$$

$$H(X_1 \dots X_{r-1} | X_r \dots X_n) = H(X_1 \dots X_n) - H(X_r \dots X_n) \quad (50)$$

Durch wiederholte Anwendung zweiterer Definition ergibt sich die *Kettenregel* für Entropien:

$$H(X_1 \dots X_n) = \sum_{i=1}^{n} H(X_i | X_1 \dots X_{i-1}) \quad (51)$$

Dies funktioniert selbst bei zusätzlichen Bedingungen:

$$H(X_1 \dots X_n | Y) = \sum_{i=1}^{n} H(X_i | X_1 \dots X_{i-1} Y) \quad (52)$$

Die Reihenfolge der Abspaltung der Zufallsvariablen X_i ist dabei egal (sie dürfen also umnummeriert werden).

Informationsquellen

Informationsquellen senden eine Folge von **Symbolen** aus einem zugehörigen **Alphabet**. Das Auftreten eines *Symbols* aus der *Quelle* heisst **Ereignis**. Das zuletzt aufgetretene *Ereignis* heisst **Zustand**.

Quellenentropie

Die *Entropie* eines gesendeten *Symbols* einer *Quelle* heisst *Quellenentropie*. Sind alle *Ereignisse* gleich wahrscheinlich, so ist sie maximal.

Markov-Quellen

Eine *Markov-Quelle* der Stufe n ist, analog zur *Markov-Kette*, eine *Informationsquelle*, deren *Ereigniswahrscheinlichkeiten* jeweils direkt abhängig der letzten n aufgetretenen *Ereignisse* sind. Es gilt also:

- $n = -1$: Alle *Ereignisse* gleich wahrscheinlich
- $n = 0$: Jedes *Ereignis* hat eine feste Wahrscheinlichkeit
- $n = 1$: Die *Wahrscheinlichkeit* eines *Ereignisses* ist direkt abhängig vom letzten aufgetretenen *Ereignis*
- $n = 2$: Die *Wahrscheinlichkeit* eines *Ereignisses* ist direkt abhängig von den letzten zwei aufgetretenen *Ereignissen*
- ...

Für die folgenden Definitionen sei jeweils $n = 1$.

Berechnung der Zustandswahrscheinlichkeiten

Sei $\boldsymbol{A}$ die zur *Quelle* gehörige *Zustandsübergangsmatrix* und $\boldsymbol{p}$ ein Vektor mit den gesuchten k *Zustandswahrscheinlichkeiten*. Dann gilt es, die folgende Funktion zu minimieren:

$$E(\boldsymbol{p}) = |\boldsymbol{A} \cdot \boldsymbol{p} - \boldsymbol{p}| \quad (53)$$

$$\text{Nebenbedingung: } F(\boldsymbol{p}) = -1 + \sum_{i=1}^{k} p_i = 0 \quad (54)$$

Laut Lagrange lässt sich das Minimum nun durch Lösen des folgenden Gleichungssystems finden:

$$\nabla E(\boldsymbol{p}) = \lambda \cdot \nabla F(\boldsymbol{p}) \quad (55)$$

$$F(\boldsymbol{p}) = 0 \quad (56)$$

Dieses Vorgehen ist äquivalent dazu, einen normierten Eigenvektor $\boldsymbol{p}$ zum Eigenwert 1 der Matrix $\boldsymbol{A}$ zu bestimmen.

Markov-Entropie

Über eine *Markov-Quelle* mit N möglichen *Zuständen* liegt Unsicherheit in zweifacher Hinsicht vor:

1. Welcher Zustand tritt als nächstes ein?
2. Welcher Zustand liegt gerade vor?

Um den ersten Punkt zu klären, berechnen wir die *Entropie* eines *Zustandsübergangs* von einem *Zustand* x_j in einen beliebigen anderen *Zustand* x_i:

$$H_j = -\sum_{i=1}^{N} p(x_i|x_j) \cdot \log_2 p(x_i|x_j) \quad (57)$$

Um nun auch den zweiten Punkt zu berücksichtigen, bildet man von diesen *Entropien* den Erwartungswert über die *Zustandswahrscheinlichkeiten* und erhält so die *Markov-Entropie*:

$$H_M = \sum_{j=1}^{N} p(x_j) \cdot H_j \quad (58)$$

Quellencodierung

Code

Ein *Code* C über einem **Codealphabet** Δ mit $D = |\Delta|$ für eine Menge χ ist eine Abbildung $C : \chi \to \Delta^*$.

- Sei $x \in \chi$. Dann heisst $C(x)$ das **Codewort** von x.
- Sei $x \in \chi$. Dann heisst $l_C(x) = |C(x)|$ die **Länge** von $C(x)$
- Sei $C : \chi \to \Delta^*$ ein *Code* mit:

$$\forall x \in \chi \; \forall y \in \chi : x \neq y \leftrightarrow C(x) \neq C(y) \tag{59}$$

  Dann heisst C **nicht-degeneriert**.

- Sei $C : \chi \to \Delta^*$ ein *Code*, bei welchem folgende Abbildung für beliebige $x_i \in \chi$ eindeutig ist:

$$[x_1||x_2||\dots||x_n] \to [C(x_1)||C(x_2)||\dots||C(x_n)] \tag{60}$$

  Dann heisst C **eindeutig decodierbar**.

- Sei $C : \chi \to \Delta^*$ ein *Code* mit:

$$\forall c \in \Delta^* \; \forall d \in \Delta^* \; \neg\exists e \in \Delta^* \setminus \{\lambda\} : c = d||e \tag{61}$$

  Dann heisst C **präfixfrei**. Jeder *präfixfreie Code* ist *eindeutig decodierbar*.

- Ein *Code* C zur *Codierung* einer Zufallsvariablen X heisst **optimal**, falls die durchschnittliche **Codelänge** $E[l_C(X)]$ minimal ist.
- Ein *Code*, dessen *Codewörter* alle gleich lang sind, heisst **gleichmässig**, sonst **ungleichmässig**.
- Ein *Code* heisst **universell**, wenn er für eine Klasse von *Informationsquellen* jede *Quelle* aus dieser Klasse asymptotisch auf die *Entropierate* codiert.

Codebäume

Jeder *Code* $C : \chi \to \Delta^*$ kann als Teilmenge der Knoten eines *Codebaumes* T dargestellt werden. Dabei ist jeder Knoten entweder ein Blatt oder hat maximal D Kinderknoten. Ein *Codebaum* heisst **ausgefüllt**, wenn jeder innere Knoten genau D Kinderknoten hat. Ein *präfixfreier Code* heisst **ausgefüllt**, wenn sein *Codebaum* ausgefüllt ist und jedes Blatt einem *Codewort* zugeordnet ist.
Sei B die **Blattmenge** eines *Codebaumes* und $P(b)$ mit $b \in B$ die Wahrscheinlichkeit eines Blattes. Dann gilt:

$$\sum_{b \in B} P(b) = 1 \tag{62}$$

Blattentropie

Die *Blattentropie* eines *Codebaumes* T ist definiert als:

$$H_T = -\sum_{b \in B} P(b) \cdot \log_2 P(b) \tag{63}$$

Blatttiefe

Für jedes $b \in B$ eines *Codebaumes* T ist $t(b)$ die *Tiefe* von b im Baum, wobei der Wurzelknoten die *Tiefe* 0 hat.

Mittlere Blatttiefe

Die *mittlere Blatttiefe* t_T eines *Codebaums* T ist gegeben durch:

$$t_T = \sum_{b \in B} P(b) \cdot t(b) \tag{64}$$

Sei Z die Menge der inneren Knoten von T und $P(z)$ mit $z \in Z$ die Wahrscheinlichkeit eines inneren Knotens. Dann gilt:

$$t_T = \sum_{z \in Z} P(z) \tag{65}$$

Für einen *ausgefüllten, präfixfreien Code* C für eine Zufallsvariable X ist t_T gleich der mittleren Wortlänge von C.

Kraft'sche Ungleichung

Ein D-ärer, *präfixfreier Code* C mit L *Codewörtern* der Längen $l_1, l_2, \dots, l_L$ existiert genau dann, wenn gilt:

$$\sum_{i=1}^{L} D^{-l_i} \leq 1 \tag{66}$$

McMillan-Theorem

Die *Kraft'sche Ungleichung* (66) gilt auch für jeden D-ären *eindeutig decodierbaren Code* (nicht unbedingt *präfixfrei*).

Folgen:

- *Eindeutig decodierbare Codes* sind niemals stärker als *präfixfreie Codes*.
- Sei ein beliebiger *Code* gegeben, der die *Kraft'sche Ungleichung* erfüllt. Dann existiert ein *präfixfreier Code* mit gleichen *Codewortlängen*.
- **Aber:** Nicht jeder *Code*, der die *Kraft'sche Ungleichung* erfüllt, muss *eindeutig decodierbar* sein.

1. Shannon'sches Codierungstheorem

Die mittlere *Codewortlänge* eines *optimalen präfixfreien Codes* über dem *Codealphabet* Δ mit $|\Delta| = D$ für eine Zufallsvariable X erfüllt folgende Ungleichung:

$$\frac{H(X)}{\log_2 D} \leq E[l_C(X)] < \frac{H(X)}{\log_2 D} + 1 \tag{67}$$

Im binären Fall ($D = 2$) vereinfacht sich dies zu:

$$H(X) \leq E[l_C(X)] < H(X) + 1 \tag{68}$$

Redundanz

Die *Redundanz* R_C eines *Codes* C für eine *Quelle* X ist definiert als:

$$R_C = E[l_C(X)] - H(X) \geq 0 \tag{69}$$

Jeder *optimale Code* hat minimale *Coderedundanz*, diese muss aber nicht unbedingt gleich Null sein!

Erweiterte Quellen

Bei Betrachtung von (67) und (68) stellt sich gezwungenermassen die Frage, wie nahe man an die untere Schranke des *1. Shannon'schen Codierungstheorems* herankommen kann.
Sei X eine zu *codierende Informationsquelle* mit *Quellalphabet* χ. Fasst man nun je m von X gesendete *Symbole* aus χ als Blöcke zusammen und interpretiert diese als von X^m gesendete *Symbole* aus dem **Blockalphabet** χ^m, so lässt sich diese **Ersatzquelle** mit einem *präfixfreien Code* über einem *Codealphabet* Δ mit $|\Delta| = D$ *codieren*. Das *1. Shannon'sche Codierungstheorem* (67) für diesen lässt sich nun wie folgt umformen:

$$\frac{H\left(X^m\right)}{\log_2 D} \leq E\left[l_C\left(X^m\right)\right] < \frac{H\left(X^m\right)}{\log_2 D} + 1 \tag{70}$$

$$\Leftrightarrow \frac{m \cdot H(X)}{\log_2 D} \leq E\left[l_C\left(X^m\right)\right] < \frac{m \cdot H(X)}{\log_2 D} + 1 \tag{71}$$

$$\Leftrightarrow \frac{H(X)}{\log_2 D} \leq \frac{E\left[l_C\left(X^m\right)\right]}{m} < \frac{H(X)}{\log_2 D} + \frac{1}{m} \tag{72}$$

$$\Leftrightarrow \frac{H(X)}{\log_2 D} \leq E\left[l_C\left(X_E(m)\right)\right] < \frac{H(X)}{\log_2 D} + \frac{1}{m} \tag{73}$$

Mit einer solchen *erweiterten Quelle* X_E ist es also möglich, durch die Wahl grosser Blockgrössen m mit der durchschnittlichen *Codelänge* beliebig nahe an die *Entropieschranke* heranzukommen. Im binären Fall ($D = 2$) vereinfacht sich (73) wie (68) zu:

$$H(X) \leq E\left[l_C\left(X_E(m)\right)\right] < H(X) + \frac{1}{m} \tag{74}$$

Fano-Ungleichung

Möchte man eine beliebige Zufallsvariable X mit Wertebereich χ anhand einer beliebigen anderen Zufallsvariablen Y schätzen und bezeichnet P_e die Wahrscheinlichkeit für eine Fehlschätzung, so gilt:

$$h\left(P_e\right) + P_e \cdot \log_2\left(|\chi| - 1\right) \geq H\left(X|Y\right) \tag{75}$$

Ist X eine binäre Zufallsvariable ($|\chi| = 2$), so vereinfacht sich (75) zu:

$$h\left(P_e\right) \geq H\left(X|Y\right) \tag{76}$$

Ist X ein N-Bit-String, d.h. $X = [Z_1, Z_2, \ldots, Z_n]$, so interessiert einen oft nicht die Wahrscheinlichkeit P_e einer Fehlschätzung von X (wie bisher), sondern die durchschnittliche Bitfehlerwahrscheinlichkeit $\overline{P}_e$ (also die Wahrscheinlichkeit einer Fehlschätzung eines Z_i). Für diese gilt:

$$h\left(\overline{P}_e\right) \geq \frac{H\left(X|Y\right)}{N} \tag{77}$$

Verlustbehaftete Datenkompression

Bewegt man sich mit einer *Codierung* unter der durch das *1. Shannon'sche Codierungstheorem* gegebenen unteren Schranke für die *mittlere Codewortlänge*, so wird die *Decodierung* des *Codes* sicher verlustbehaftet sein.
Sei X^N ein N-Bit-String aus der binären *Informationsquelle* X und $C(X^N)$ dessen binäre *Codierung*. Aus $E\left[l_C\left(X^N\right)\right] \geq H\left(C\left(X^N\right)\right)$ folgt nun:

$$H\left(X^N|\, C\left(X^N\right)\right) = H\left(X^N, C\left(X^N\right)\right) - H\left(C\left(X^N\right)\right) \tag{78}$$

$$= H\left(X^N\right) - H\left(C\left(X^N\right)\right) \tag{79}$$

$$\geq H\left(X^N\right) - E\left[l_C\left(X^N\right)\right] \tag{80}$$

(78)⇒(79), da sämtliche Information über $C\left(X^N\right)$ bereits in X^N enthalten ist.
Mit der speziellen *Fano-Ungleichung* (77) ergibt sich nun für die mittlere Bitfehlerwahrscheinlichkeit $\overline{P}_e$ beim *Decodieren*:

$$\overline{P}_e \geq h^{-1}\left(\frac{H\left(X^N \mid C\left(X^N\right)\right)}{N}\right) \tag{81}$$

$$\geq h^{-1}\left(\frac{H\left(X^N\right) - E\left[l_C\left(X^N\right)\right]}{N}\right) \tag{82}$$

$$= h^{-1}\left(H(X) - \frac{E\left[l_C\left(X^N\right)\right]}{N}\right) \tag{83}$$

(81)⇒(82), da auch h^{-1} im betrachteten h-Intervall $\left[0, \frac{1}{2}\right]$ monoton steigend ist.

Codeoptimalität

Es gilt:

- Der *Codebaum* jedes *optimalen präfixfreien Codes* ist *ausgefüllt*.
- Für jede *Informationsquelle* X existiert ein *optimaler präfixfreier Code*, dessen Codierungen für beiden unwahrscheinlichsten *Symbole* aus X sich nur im letzten Bit unterscheiden.
- Sei C ein binärer, *präfixfreier Code* für die Wahrscheinlichkeitsverteilung $[p_1, p_2, \ldots, p_{L-1}, p_L], p_1 \geq p_2 \geq \ldots \geq p_L$ mit *Codebaum* T. Sei C' der resultierende *Code* für $[p_1, p_2, \ldots, p_{L-1} + p_L]$, wenn man die beiden unwahrscheinlichsten *Blätter* aus T durch ihren gemeinsamen Vorfahren ersetzt. Dann ist C genau dann *optimal*, wenn C' *optimal* ist.

Huffman-Coding

Der folgende Algorithmus liefert einen *optimalen, präfixfreien Code* für die Wahrscheinlichkeitsverteilung $[p_1, p_2, \ldots, p_L]$:
Fasse in $L - 1$ Schritten jeweils die beiden unwahrscheinlichsten Knoten zu einem zusammen, indem für den neuen die beiden Wahrscheinlichkeiten addiert werden.
Dem entstandenen Codebaum müssen nun nur noch bei jeder Gabelung die Symbole 0 und 1 zugewiesen werden. Die *Optimalität* des Codes folgt direkt aus dem dritten Punkt des vorhergehenden Abschnitts.

Shannon-Fano-Coding

Der folgende Algorithmus liefert einen *suboptimalen, präfixfreien Code* für die Wahrscheinlichkeitsverteilung $[p_1, p_2, \ldots, p_L]$, welcher aber ebenfalls die Schranken des *1. Shannon'schen Codierungstheorems* einhält:

1. Ordne die Wahrscheinlichkeiten nach fallenden Werten und betrachte die sortierte Liste
2. Teile die aktuell betrachtete Liste in zwei Unterlisten auf mit möglichst gleich grossen Wahrscheinlichkeitssummen ein
3. Weise der ersten Unterliste das Symbol 0 zu, der zweiten das Symbol 1

4. Falls die erste Unterliste noch mehr als ein Element hat, betrachte diese und führe die Schritte 2-5 rekursiv aus.

5. Falls die zweite Unterliste noch mehr als ein Element hat, betrachte diese und führe die Schritte 2-5 rekursiv aus.

Arithmetic-Coding

Der folgende Algorithmus *codiert* das von einer *Informationsquelle* X mit Verteilung $[p_1, p_2, \ldots, p_L]$ generierte Wort $s = X^n$. Für grosse Wortlängen n strebt die mittlere *Codelänge* dieser *Codierung* gegen die *Quellenentropie* $H(X)$.

1. Betrachte das Intervall $[0, 1)$

2. Teile das aktuell betrachtete Intervall gemäss Wahrscheinlichkeitsverteilung auf

3. Nimm das nächste Symbol x_i aus s und betrachte das Unterintervall, das $P(x_i)$ gemäss in Schritt 2 durchgeführter Teilung entspricht

4. Führe Schritte 2+3 aus, bis das ganze Wort s gelesen wurde

5. Weise dem Wort s die ersten $\lceil -\log_2 P(s) \rceil + 1$ Bits der in der Mitte des betrachteten Intervalls liegenden Binärzahl zu

Die aktuelle untere Intervallgrenze und die aktuelle Intervallgrösse können mit folgenden Rekursionen mitgeführt werden:

```
newleft = prevleft + left[x[i]] * prevsize;
newsize = prevsize * size[x[i]];
```

Dabei bezeichnet `left[x[i]]` die untere Intervallgrenze des Symbols x_i in der Wahrscheinlichkeitsverteilung und `size[x[i]]` dessen Wahrscheinlichkeit.

Ganzzahlencodierung

Sei $j \in \mathbb{N}$ eine natürliche Zahl, $B(j)$ ihre Binärdarstellung, $B'(j)$ ihre Binärdarstellung ohne führende Eins und $L(j)$ ihre binäre Länge. Folgender rekursiver *Code* kodiert j asymptotisch auf *Entropierate*:

$$C^n(j) = \begin{cases} 0^{j-1}1 & n = 0 \\ C^{n-1}(L(j)) \parallel B'(j) & n > 0 \end{cases} \tag{84}$$

Intervalllängencodierung

Sei X eine binäre *Quelle* mit $P[X = 1] = p$ und dementsprechend $P[X = 0] = 1 - p$. Eine lange Zeichenfolge mit $p << \frac{1}{2}$ kann man nun *codieren*, indem man anstatt die Folge selbst jeweils die Abstände zwischen aufeinanderfolgenden Einsen *codiert* (zum Beispiel mit der im vorigen Abschnitt eingeführten *Ganzzahlencodierung*). Für $p \to 0$ erhält man so einen asymptotisch *optimalen Code*.

Lempel-Ziv-Coding

Der *Grundcodierungsalgorithmus* von *Lempel-Ziv codiert* einen bekannten String ohne Wissen über die *Quellenstatistik* in zwei Durchläufen *optimal*:

1. Der String wird von links nach rechts durchlaufen und in disjunkte Teilstrings unterteilt.

2. Die Teilstrings werden von links nach rechts durchlaufen und jeder Teilstring durch ein Zahlentupel *codiert*, dessen erste Zahl die Nummer des bereits kodierten Teilstrings, welcher dem längsten Präfix des aktuellen Teilstrings entspricht, und dessen zweite Zahl das letzte Bit, das sich vom Präfix als einziges unterscheidet, angibt. Für die erste Zahl wird eine Null verwendet, falls der Teilstring keinen bzw. einen leeren Präfix hat.

Lempel-Ziv-Welch-Coding

Das *Lempel-Ziv-Welch-Coding* stellt eine Erweiterung des *Lempel-Ziv-Coding* dar, bei welchem man nur noch einen Durchlauf braucht und somit der zu *codierende* String am Anfang noch nicht komplett bekannt sein muss. Die *Codierung* erfolgt mit folgendem Algorithmus:

1. Initialisiere das Wörterbuch mit allen möglichen Einzelzeichen (weise jedem von ihnen einen Index zu). Initialisiere neuen Präfix w' als leeren String.

2. Aktualisiere den aktuellen Präfix: $w = w'$

3. Lese nächstes Zeichen des Inputstrings, falls noch nicht am Ende. Ansonsten gebe *Codierung* von w aus und stoppe.

4. Konkateniere aktuelles Zeichen z mit aktuellem Präfix w und speichere dies als neuen Präfix: $w' = w||z$

5. Überprüfe, ob w' bereits im Wörterbuch vorhanden ist. Wenn ja, fahre fort mit Schritt 2.

6. Gebe die *Codierung* des alten Präfix w aus (den Wörterbuchindex von w).

7. Füge w' zum Wörterbuch hinzu

8. Ersetze neuen Präfix durch aktuelles Zeichen: $w' = z$

9. Fahre fort mit Schritt 2

Kanalcodierung

Binärer Kanal

Ein *binärer Kanal* ist ein Übertragungsmodell, bei welchem das *Symbol* 1 mit Wahrscheinlichkeit δ fälschlicherweise auf 0 und das *Symbol* 0 mit Wahrscheinlichkeit ε fälschlicherweise auf 1 abgebildet wird. Mit x_s wird jeweils das Senden des *Symbols* s bezeichnet und mit $y_{s'}$ das Empfangen des *Symbols* s'. Allgemein modelliert die Zufallsvariable X das Senden und die Zufallsvariable Y das Empfangen eines durch den *Kanal* geschickten *Symbols*.
Ein *binärer Kanal* heisst **symmetrisch**, wenn $\delta = \varepsilon$ gilt.
Ein *binärer Kanal* heisst **gedächtnisfrei**, falls ein empfangenes *Symbol* y nur von dem zugehörigen gesendeten *Symbol* x abhängt.
Beim **binären Auslöschungskanal** findet keine Bitinversion statt, sondern ein gesendetes Bit kommt mit der Wahrscheinlichkeit $\delta = \varepsilon$ einfach gar nicht an.

Transinformation

Die *Transinformation* H_T ist die pro *Kanalzeichen* übertragene Information. Für den *binären Kanal* ergibt sich:

$$H_T = I(X;Y) = H(Y) - H(Y|X) \tag{85}$$

Dementsprechend ist das Ziel jeweils, die *Transinformation* zu maximieren.

Blockcodes

Ein *Blockcode* C der **Blocklänge** N für einen *Kanal* mit Inputalphabet γ ist eine Teilmenge $C = \{c_1, c_2, \ldots, c_M\} \subset \gamma^N$ der N-Tupel über γ. Die **Rate** von C ist definiert als:

$$R = \frac{\log_2 M}{N} = \frac{K}{N} \tag{86}$$

Schätzer

Wird das *Codewort* $c_j = [c_{j1}c_{j2}\ldots c_{jN}]$ über einen *gedächtnisfreien Kanal* übertragen, so kommt beim Empfänger das Wort $y^N \in \mathcal{W}\left(Y^N\right)$ an mit der Verteilung:

$$P_{Y^N|X^N}\left(y^N, c_j\right) = \prod_{i=1}^{N} P_{Y|X}\left(y_i, c_{ji}\right) \tag{87}$$

Die *Decodierung* der fehlerbehafteten Zeichenfolge kann nun als *Schätzproblem* betrachtet werden: Dabei soll die Zufallsvariable X^N durch Beobachtung von Y^N mit einer Funktion als $f\left(Y^N\right)$ geschätzt werden, wobei die *Kanalcharakteristik* $P_{Y|X}$ bekannt ist. Eine solche Schätzung heisst **optimal**, wenn sie die Wahrscheinlichkeit $P_{X^N}\left(f\left(Y^N\right)\right)$ einer korrekten Schätzung maximiert:

$$\begin{aligned} P_{X^N}\left(f\left(Y^N\right)\right) &= \sum_{y^N \in Y^N} P_{X^N Y^N}\left(f\left(y^N\right), y^N\right) \\ &= \sum_{y^N \in Y^N} P_{Y^N|X^N}\left(y^N, f\left(y^N\right)\right) \cdot P_{X^N}\left(f\left(y^N\right)\right) \end{aligned} \tag{88}$$

Nun können folgende zwei Fälle auftreten:

1. P_X ist bekannt: In diesem Fall muss f so gewählt werden, dass $P_{Y^N|X^N}\left(y^N, f\left(y^N\right)\right) \cdot P_X\left(f\left(y^N\right)\right)$ über alle $y^N \in Y^N$ maximiert wird. Dies bezeichnet man als **Minimum-Error-Estimation**.
2. P_X ist unbekannt (Normalfall): Man nimmt an, dass alle Werte von X gleichverteilt sind. Da $P_{X^N}(c)$ dann für alle c gleich ist, wird es in (88) zu einer Konstante und f muss nur noch so gewählt werden, dass $P_{Y^N|X^N}\left(y^N, f\left(y^N\right)\right)$ über alle $y^N \in Y^N$ maximiert wird. Dies wird bezeichnet als **Maximum-Likelihood-Estimation**.

Kapazität

Die *Kapazität* eines *Kanals* ist definiert als maximale *Transinformation*, die über ihn übertragen werden kann:

$$C = \max_{P_x} H_T = \max_{P_x} \{H(Y) - H(Y|X)\} \tag{89}$$

Die *Kapazität* stellt eine obere Grenze dar für die *Rate*, mit welcher Informationen zuverlässig über den *Kanal* übertragen werden können.

Sei γ das *Kanalalphabet* und damit auch die Wertemenge von Y, sei $t = H(Y|X = x)$ gleich für alle $x \in X$ und sei $\sum_{x \in X} P[Y = y \mid X = x]$ gleich für alle $y \in Y$, wobei die zweite Eigenschaft bedeutet, dass eine Gleichverteilung am *Kanaleingang* auch eine solche am *Kanalausgang* bewirkt. Dann vereinfacht sich die Berechnung der *Kapazität* zu:

$$C = \log_2 |\gamma| - t \tag{90}$$

Sei nun U^K eine mit einem *Blockcode* der Länge N über einen *Kanal* mit *Kapazität* C versandte Information und $\tilde{U}^K$ der *decodierte*, geschätzte Information nach der *Kanalübertragung*. Dann gilt:

$$H\left(U^K \mid \tilde{U}^K\right) \geq H\left(U^K\right) - N \cdot C \tag{91}$$

Eine *Kanalbenutzung* kann also die Unsicherheit über das gesendete *Codewort* höchstens um $N \cdot C$ verringern. Wenn man also U^K durch $\tilde{U}^K$ vollständig bestimmen will, so muss gelten:

$$N \geq \frac{H\left(U^K\right)}{C} \tag{92}$$

2. Shannon'sches Codierungstheorem

Wird ein *gedächtnisfreier, binärer Kanal* der *Kapazität* C zur Übertragung echt zufälliger Informationsbits mit *Rate* $R > C$ benutzt, so gilt für die mittlere Bitfehlerwahrscheinlichkeit beim Empfänger:

$$h\left(\overline{P}_e\right) \geq 1 - \frac{C}{R} \tag{93}$$

Überträgt man diese Informationsbits mit Rate $R < C$, so existiert für jedes ε ein N-*Blockcode* mit $M = 2^{R \cdot N}$ *Codewörtern*, sodass die maximale *Decodierfehlerwahrscheinlichkeit* für ein *Codewort* kleiner als ε ist:

$$\max_j P\left[\text{ Fehler } \mid X^N = c_j\right] < \varepsilon \tag{94}$$

Hammingdistanz

Die *Hammingdistanz* zweier Wörter ist die Anzahl der Positionen, an welchen sie sich unterscheiden.
Die **Minimaldistanz** eines *Codes* ist die minimale *Hammingdistanz* zwischen zwei *Codewörtern* des *Codes*. Ein Code mit *Minimaldistanz* d erlaubt es, $d-1$ Fehler zu detektieren oder $\left\lfloor \frac{d-1}{2} \right\rfloor$ zu korrigieren.

Lineare Blockcodes

Ein *linearer Blockcode* ist ein *Blockcode* mit $M = q^k$ *Codewörtern* der Länge N, der die algebraische Struktur eines Vektorraums über dem dem endlichen Körper $GF(q)$ hat.
Das **Hamminggewicht** eines *Codewortes* ist die Anzahl sich von Null unterscheidenen Positionen.
Die *Minimaldistanz* eines *linearen Blockcodes* ist gleich dem kleinsten *Hamminggewicht* seiner *Codewörter*.
Die Abbildung eines Informationsvektors a auf ein *Codewort* c kann mit Hilfe einer $K \times N$-**Generatormatrix** als Matrixmultiplikation dargestellt werden:

$$c = a \cdot G \tag{95}$$

Dabei bilden die Zeilen von G eine Basis des *Codewortraums*. Hat die *Generatormatrix* die Form $G = [I_k \ A]$, so heisst sie **systematisch**.
Die $(N-K) \times N$-Matrix H heisst **Parity-Check-Matrix** für den *Code*, falls gilt:

$$\forall c: \ c \cdot H^T = 0 \tag{96}$$

Die Zeilen von H spannen also den orthogonalen Komplementärraum zum *Codewortraum* auf. Die *Minimaldistanz* eines *linearen Codes* ist gleich der kleinsten Anzahl linear abhängiger Spalten in H.

Systematische Parity-Check-Matrix

Sei $G = [I_K \ A]$ eine *systematische Generatormatrix*. Dann ist $H = \left[-A^T \ I_{N-K}\right]$ die zugehörige *Parity-Check-Matrix*.

Syndromcodierung

Das Verfälschen eines *Codewortes* c beim Versenden durch einen verrauschten Kanal kann als Addition eines Fehlervektors e betrachtet werden. Durch Multiplikation des empfangenen Wortes $\tilde{c} = c + e$ mit der *Parity-Check-Matrix* H^T bekommt man das **Syndrom** s:

$$(c+e) \cdot H^T = c \cdot H^T + e \cdot H^T = e \cdot H^T = s \tag{97}$$

Anstatt nun wie bisher anhand von $c + e$ ein c zu schätzen, ist es effektiver, e aufgrund von s zu schätzen und dann c durch die Subtraktion $(c+e) - e$ zu bestimmen. Damit muss man anstelle des *Coderaumes* der Dimension N nur noch den *Syndromraum* der Dimension $N - K$ durchsuchen. Dies wird erreicht, indem man eine Tabelle anfertigt, in der man sich für jedes der q^{N-K} *Syndrome* das Fehlermuster e mit dem kleinsten *Hamminggewicht* notiert (da man davon ausgehen muss, dass das empfangene Wort nur auf minimalste Art verfälscht wurde).
Um nun Fehler zu korrigieren multipliziert man ein empfangenes Wort $\tilde{c}$ mit H^T, bekommt so das *Syndrom* s, bestimmt aus der Tabelle das Fehlermuster e und subtrahiert dies von $\tilde{c}$, um auf c zu schliessen.
Zu beachten ist, dass mit dieser Methode alle Fehlermuster e mit *Hamminggewicht* $\leq r = \left\lfloor \frac{d_{min}-1}{2} \right\rfloor$ nur dann eindeutig korrigierbar sind, wenn genügend verschiedene *Syndrome* bzw. genügend $N - K$ Kontrollbits vorhanden sind. Bei einem *Binärcode* ($q = 2$) gibt es für einen Fehlervektor mit *Hamminggewicht* i genau $\binom{N}{i}$ Möglichkeiten. Das macht für alle Fehlervektoren mit *Hamminggewicht* $\leq r$ die Anzahl $\sum_{i=1}^{r} \binom{N}{i}$. Um nun eine eindeutige Zuordnung zu erhalten, muss gelten:

$$2^{N-K} \geq \sum_{i=1}^{r} \binom{N}{i} \tag{98}$$

Hamming-Codes

Für jedes $r > 1$ existiert ein *Hamming-Code* aus $N = 2^r - 1$ Bits mit $K = N - r$ Informationsbits und *Minimaldistanz* $d_{min} = 3$.
Die Spalten der *Parity-Check-Matrix* H eines solchen *Codes* bestehen aus allen $2^r - 1$ vom Nullvektor unterschiedlichen Vektoren der Dimension r. Offensichtlich ist es sehr leicht, diese *systematisch* anzuordnen und so auf die *Generatormatrix* G zu kommen.

Duale Codes

Ein *Code* C' heisst *dual* zu einem anderen Code C, falls seine *Generatormattrix* $G_{C'} = H_C$, also gerade gleich der *Parity-Check-Matrix* von C ist. Der *duale Code* zu C' ist wieder C.

Dualer Hamming-Code

In einem zu einem *Hamming-Code dualen Code* ist der Informationsgehalt mit r klein, die *Minimaldistanz* jedoch sehr gross mit 2^{r-1}. Die Länge bleibt mit $2^r - 1$ gleich.

Singleton Bound

Die *Minimaldistanz* eines *linearen Codes* der Länge N aus K Informationsbits über $GF(q)$ beträgt höchstens $N - K + 1$.

Polynomevaluationscodes

Die *Polynomevaluationscodes* sind *lineare Codes*, bei welchen die *Codewörter* der Länge N als N Auswertungen eines Polynoms des Grades $K - 1$ über $GF(q)$ an N verschiedenen Stellen interpretiert werden.
Polynomevaluationscodes erreichen die durch den *Singleton Bound* gegebene Schranke für die *Minimaldistanz*.