1 Osnove

1.1 Ponovitev logaritmov

- $log_a x = \frac{log_b x}{log_b a}$
- $log_b(\frac{x}{y}) = log_b x - log_b y$
- $x = b^y \implies log_b x = y$
- $log_2 x = log x$
- $0 log 0 = 0$

1.2 Entropija

je povprecje vseh lastnih informacij:

$$H(X) = \sum_{i=1}^{n} p_i I_i = -\sum_{i=1}^{n} p_i log p_i$$

Lastnosti: je zvezna, simetricna funckija (vrsni red p_i ni pomemben, sestevanje je komutativno). Je vedno vecja od 0 ($p_i \geq 0 \rightarrow -p_i \log p_i \geq 0 \rightarrow H(X) \geq 0$) in navzgor omejena z $\log n$.
Ce sta dogodka **neodvisna** velja aditivnost: $H(X,Y) = H(X) + H(Y)$.
Vec zaporednih dogodkov neodvisnega vira: $X^l = X \times \cdots \times X \rightarrow H(X^l) = lH(X)$.

2 Kodi

2.1 Uvod

Kod sestavljajo *kodne zamenjave*, ki so sestavljene iz znakov **kodne abecede**. Stevilo znakov v kodni abecedi oznacujemo z **r**.
Ce so $\{p_1, \ldots, p_n\}$ verjetnosti znakov $\{s_1, \ldots, s_n\}$ osnovnega sporocila in $\{l_1, \ldots, l_n\}$ dolzine prejetih kodnih zmanjav, je povprecna dolzina kodne zamenjave

$$L = \sum_{i=1}^{n} p_i l_i$$

2.2 Tipi kodov

- **optimalen** - ce ima najmanjso mozno dolzino kodnih zamenjav
- **idealen** - ce je povprecna dolzina kodnih zamenjav enaka entropiji
- **enakomeren** - ce je dolzina vseh kodnih zamenjav enaka
- **enoznacen** - ce lahko poljuben niz znakov dekodiramo na en sam nacin
- **trenuten** - ce lahko osnovni znak dekodiramo takoj, ko sprejmemo celotno kodno zamenjavo

2.3 Kraftova neenakost

Za dolzine kodnih zamenjav $\{l_1, \ldots, l_n\}$ in r znaki kodne abecede obstaja trenutni kod, iff

$$\sum_{i=1}^{n} r^{-li} \leq 1$$

2.4 Povp. dolzina, ucinkovitost

Najkrajse kodne zamenjave imamo, ce velja:

$$H_r(X) = L \rightarrow l_i = \lceil -\log_r p_i \rceil$$

Ucinkovitost koda:

$$\eta = \frac{H(X)}{L \log_r}, \eta \in [0, 1]$$

Kod je **gospodaren**, ce je L znotraj:

$$H_r(X) \leq L < H_r(X) + 1$$

kjer je $H_r(X)$:

$$H_r(X) = -\sum_{i=1}^{n} \frac{\log_{p_i}}{\log_r} = \frac{H(X)}{\log_r}$$

2.5 Shannonov prvi teorem

Za nize neodvisnih znakov dozline n obstajajo kodi, za katere velja:

$$\lim_{n\rightarrow\infty} \frac{L_n}{n} = H(X)$$

pri cemer je $H(X)$ entropija vira X.
Postopek kodiranja po Shannonu:

1. znake razvrstimo po padajocih verjetnostih
2. dolocimo stevilo znakov v vsaki kodni zamenjavi (l_k)
3. za vse simbole izracunamo komulativne verjetnosti ($P_k = \sum_{i=1}^{k-1} p_i$)
4. P_k pretvorimo v bazo r. Kodno zamenjavo predstavlja prvih l_k znakov necelega dela stevila

2.6 Fanojev kod

Postopek kodiranja:

1. znake razvrstimo po padajocih verjetnostih
2. znake razdelimo v r cim bolj enako verjetnih skupin
3. Vsaki skupini priredimo enega od r znakov kodne abecede
4. Deljenje ponovimo na vsaki od skupin. Postopek ponavljamo, dokler je mogoce

2.7 Huffmanov kod

Huffmanov postopek kodiranja poteka od spodaj navzgor (Pri Fanoju je ravno obratno). Pri huffmanovem kodu imamo dve fazi:

1. Zdruzevanje
   (a) Posici r najmanj verjetnih znakov in jih zdruzi v sestavljeni znak, katerega verjetnost je vsota verjetnosti vseh znakov
   (b) Preostale znake skupaj z novo sestavljenim znakom spet razvrsti
   (c) Postopek ponavljaj dokler ne ostane samo r znakov
2. Razdruzevanje
   (a) Vsakemu od preostalih znakov priredi po en znak kodirne abecede
   (b) Vsak sestavljeni znak razstavi in mu priredi po en znak kodirne abecede
   (c) Ko zmanjka sestavljenih znakov, je postopek zakljucen

Pred kodiranjem, je vedno pametno preveriti, ce imamo zadostno stevilo znakov. Veljati mora:

$$n = r + k(r-1), k \geq 0$$

Ce imamo premalo znakov, jih po potrebi dodamo s verjetnostjo $p = 0$.
Huffmanov kod lahko razsirimo tako, da vec osnovnih znakov zdruzujemo v sestavljene znake $\rightarrow$ bolj ucinkoviti kodi. Vendar naletimo na nevarnost kombinacijske eksplozije.

2.8 Aritmeticni kod

Je **hiter** in **blizu optimalnemu** kodu, ter manj ucinkovit kot Huffmanov, vendar se izogne kombinacijski eksploziji. Vsak niz je predstavljen kot realno stevilo $0 \leq R < 1$, kar nam pove, da daljsi kot bo niz, bolj natancno mora biti podano naravno stevilo R.
Postopek kodiranja(znakov ni potrebno razvrstiti):

1. Zacnemo z intervalom $[0, 1)$
2. Izbrani interval razdelimo na n podintervalov, ki se ne prekrivajo. Sirine podintervalov ustrezajo verjetnostim znakov. Vsak podinterval predstavlja en znak
3. Izberemo podinterval, ki ustreza iskanemu znaku
4. Ce niz se ni koncan, izbrani podinterval ponovno razdelimo (bne 2.tocka)
5. Niz lahko predstavimo s poljubnim realnim stevilom v zadnjem podintervalu

Ko dobimo realni interval, ga samo se pretvorimo v binarnega s pomocjo klasicnega pretvarjanja iz dec v bin stevilski sistem.

2.9 Kod Lempel-Ziv (LZ77)

Stiskanje temelji na osnovi slovarja, tako, da ne potrebujemo racunati verjetnosti za posamezne znake. **Kodirnik** med branjem niza gradi slovar, in **dekodirnik** med branjem kodnih zaamenjav rekonstruira slovar in znake.
Kodiranje: uporablja drseca okna, znaki se premikajo iz desne na levo. Referenca je podana kot trojcek:

- odmik - razdalja do zacetka enakega podniza v medpomnilniku
- dolzina enakega podniza
- naslednji znak

npr. (0, 0, A) - ni ujemanja, (4, 3, B) - 4 znake nazaj se ponovi 3 znakovni podniz, ki se nato zakljuci s B.
dekodiranje: sledimo kodnim zamenjavam

2.10 Deflate

Gre za predelan LZ77. Uporablja pare (odmik, dolzina). Ce ujemanja v kodni tabeli ni, zapise kar znak. Uporablja dve kodni tabeli:

- **tabela za znake in dolzine** - 285 simbolov (0-255 za osnovne znake, 256 konec bloka, 257-285 kodira dolzine) Kodne zamenjave brez dodatnih bitov, se zakodira s Huffmanom.
- **tabela odmikov**

Niz znakov se razdeli na bloke(64k) vsak blok se kodira na enega od treh nacinov:

1. **brez stiskanja** osnovni znaki se prepisejo
2. **stiskanje s staticnim Huffmanom** (verjetnosti podane vnaprej), Huffmanovo drevo ni zakodirano v bloku
3. **stiskanje s Huffmanom** izracunamo verjetnosti za vsak blok

Glava posameznega bloka: 1bit - zadnji/ni zadnji blok + 2bita tip stiskanja + pri (3) se Huffmanovo drevo Ker Huffmanovo drevo ni enolicno, uvedemo kanonicni Huffmanov kod. Postopek:

1. znake razvrstimo najprej po dolzinah kodnih zamenjav in nato po abecedi
2. prvi simbol ima same nicle
3. vsakemu naslednjemu znaku dodelimo naslednjo binarno kodo (prejsnji + 1)
4. ce je kodna zamenjava daljsa od binarne kode stevila, na koncu pripnemo niclo
5. ponavjlaj (3) do konca

Na taksen nacin dosezemo, da je potrebno kodirati samo dolzine kodnih zamenjav.

2.11 Kod Lempel-Ziv (LZW)

Osnovni slovar je podan in ga sporti doponjujemo. Alogritem za **kodiranje**:

```
N = ""
ponavljaj:
    preberi naslednji znak z
    ce je [N,z] v slovarju:
        N = [N, z]
    drugace:
        izpisi indeks k niza N
        dodaj [N, z] v slovar
        N = z
    izpisi indeks k niza N
```

Algoritem za **dekodiranje**:

```
preberi indeks k
poisci niz N, ki ustreza indeksu k
izpisi N
L = N
ponavljaj:
    preberi indeks k
    ce je k v slovarju:
        poisci niz N
    drugace:
        N = [L, L(1)]
    izpisi N
    v slovar dodaj [L, N(1)]
    L = N
```

LZW doseze optimalno stiskanje, probliza se entropiji.

2.12 Verizno kodiranje ali RLE (run lenght encoding)

Namesto originalnih podatkov, sharnjujemo dolzino verige (fffeef → 3f2e1f). Problemu, ko se podatki ne ponavljajo, se izognemo tako, da izvedemo kombinacijo direktnega kodiranja in kodiranja RLE.

2.13 Stiskanje z izgubami

S taksnim nacinom stiskanja lahko dosezemo veliko boljsa kompresijska razmerja, vendar izgubimo podatke. Zato ga uporabljamo samo s formati, kjer se ne ukvarjamo z integriteto podatkov(MP3, MPEG, JPEG, ...). Postopki kodiranj znanih formatov:

- **JPEG**
  1. priprava slike → ker je svetlost bol pomembna, je barvna resolucija obicajno zmanjsana (YC_RC_B)
  2. aproksimacija vsake od treh komponent s 2D DCT
  3. kvantizacija → podatki ki bolj izstopajo so shranjeni manj natancno kot tisti ki so staticni
  4. kodiranje blokov s pomocjo entropije
  5. RLE cik-cak po sliki
  6. RLE kodiramo z Huffmanom ali Aritmenticnim kodom
- **MP3**
  1. Modified DCT
  2. odstranitev za cloveka neslisnih frekvenc
  3. stereo, ce sta si L in R pretvorimo v mono
  4. Huffman na koncu
- **MPEG**
  1. uvodno kodiranje → celotna slika JPEG
  2. nato pa kodiramo samo spremembe, ki so se zgodile v sliki JPEG s pomocjo vektorja premika. V primeru, da je prevec razlik, se ponovno kodira JPEG slika.

2.14 Kompresijsko razmerje

Izracunamo ga po formuli → stisnjeni binarni zapis C(M) / binarni zapis dokumenta (M):

$$R = C(M)/M$$

3 Kanali

3.1 Uvod

Kanali so strukture, ki opisujejo medsebojno povezanost. Kanal prenasa informacijo o spemenljivki X do spremenljivke Y. Matematicno ga opisemo s **pogojnimi verjetnostmi**, ki povezujejo izhodne verjetnosti z vhodom.

3.2 Diskretni kanal brez spomina

Povezuje diskretni nakljuicni spremenljivki, s koncno mnozico stanj $X = \{x_1, \ldots, x_r\}$ in $Y = \{y_1, \ldots, y_s\}$. Obema nakljicnima spremenljivkama pripadajo tudi posamezne verjetnosti $P_X = \{p(x_1), \ldots, p(x_r)\}$ in $P_Y = \{p(y_1), \ldots, p(y_s)\}$. Kjer velja, da je vsota posamezne mnozice verjetnosti enaka 1. Kanal je definiran kot mnozica **pogojnih verjetnosti**

$$p(y_j|x_i).$$

Pogojna verjetnost nam pove verjetnost za dogodek y_j na izhodu iz kanala, ce je na vhodu v kanal dogodek x_i. Brez spomina je zato, ker so pogojne vrjetnosti konstantne in torej neodvisne od prehodnih simbolov, velja

$$\sum_j p(y_j|x_i) = 1.$$

Kanal popolnoma podamo z $r \times s$ pogojnimi verjetnostmi.

3.2.1 Binarni simetricni kanal (BSK)

Gre za poseben primer diskretnega kanala brez spomina. Napaka kanala je p, saj se z verjetnostjo p znak prenese v napacnega.

$$P_k = \begin{pmatrix} 1-p & p \\ p & 1-p \end{pmatrix}$$

3.3 Pogojna entropija

Pogojna entropija spremenljivke Y pri znanem X se zapise kot $H(Y|X)$. Vzemimo, da se je zgodil dogodek $x_i \in X$. Entropija dogodka Y je potem

$$H(Y|x_i) = -\sum_{j=1}^{s} p(y_j|x_i)\log(p(y_j|x_i)).$$

Velja: $0 \leq H(Y|x_i)$.
Ce pa o dogodku X vemo le da se je zgodil, se lahko spomnemo na vis in uporabimo **vezano verjetnost** dogodkov X in Y, ki pravi:

$$p(x_i, y_i) = p(x_j|x_i)p(x_i)$$

Za entropijo:

$$H(Y|X) = \sum_i p(x_i)H(Y|x_i)$$
$$= -\sum_{i=1}^{r}\sum_{j=1}^{s} p(x_i, y_i)\log p(y_i|x_i)$$

Splosno velja: $0 \leq H(Y|X) \leq H(Y)$, ce poznamo spremenljivko X, se nedolocenost Y ne more povecati (lahko se pomanjsa).

3.4 Vezana entropija spremeljivk

Vezana entropija nakljucnih spremenljivk X in Y je entropija para (X, Y). Pomembne zveze:

- $p(x_i, y_j) = p(y_j|x_i)p(x_i)$,
- $\sum_j p(x_i, y_j) = p(x_i)$,
- $\sum_i p(x_i, y_j) = p(y_j)$,
- $\sum_{i,j} p(x_i, y_j) = 1$

Velja: $H(X, Y) = H(Y|X) + H(X)$, kar nam pove, da ce najprej izvemo, kaj se je zgodilo v dogodku X in potem dobimo se dodatne informacije od dogodku Y, vemo vse.

3.4.1 Obrat kanala

Ker velja tudi $H(X, Y) = H(X|Y) + H(Y)$, kanal lahko **obrnemo** (sepravi vhod Y in izhod X). Pri tem ne obracamo fizicnega procesa, ampak samo verjetnostno strukturo, ki definira kanal. **Pogoj:** poznati moramo vhodne verjetnosti. Iz njih lahko dolocimo izhodne verjetnosti, ki jih lahko uporabimo kot vhodne verjetnosti v obrnjeni kanal. Lastnosti:

- izracun izhodnih verjetnosti $p(y_j) = \sum_i p(y_j, x_i)p(x_i)$
- obratne pogojne vrjetnosti $p(x_i, y_j) = p(y_j|x_i)p(x_i) = p(x_i|y_j)p(y_j)$

Za sprejemnika sporocila so obratne pogojne verjetnosti zelo pomembne, saj z njimi lahko iz prejetih znakov doloci verjetnost za vhodne znake.

3.5 Medesebojna informacija

Pove nam, koliko o eni spremenljivki izvemo iz druge spremenljivke, definicija:

$$I(X;Y) = H(X) - H(X|Y)$$

Lastnosti:

- $I(X;Y) = H(X,Y) - H(X|Y) - H(Y|X)$
- $I(X;Y) = H(X) - H(X|Y)$
- $I(X;Y) = H(Y) - H(Y|X)$
- $I(X;Y) = H(X) + H(Y) - H(X,Y)$
- $I(X;Y) =$ simetricna glede na X in Y
- $I(X;Y) = -\sum_i \sum_j p(x_i, y_j)\log \frac{p(x_i)p(y_j}{p(x_i, y_j)}$
- $I(X;Y) \geq 0$
- $I(X;X) = H(X)$