Poročilo za prvo laboratorijsko vajo predmeta Informacija in Kodi

Gašper Šavle[1]

[1] *Univerza v Ljubljani, Fakulteta za Elektrotehniko*
E-pošta: gaspersavle@yahoo.com

Abstract

This project explores the entropy of compressed and uncompressed files across various file types to analyze the relationship between compression and information entropy. Entropy, a measure of randomness in data, provides insights into the efficiency of compression algorithms and the inherent information content within a file.

Using Python, the entropy calculation is automated with libraries such as NumPy and collections for efficient data analysis. Shannon's entropy formula quantifies the uncertainty of data values, and files from diverse formats, including text, images, and audio, are examined in both original and compressed states.

The findings highlight trends in entropy changes across formats, offering insights into how file structure and compression impact data randomness. This study contributes to understanding data compression efficiency and its implications for storage optimization and information theory.

1 Uvod

Naloga analizira informacijske lastnosti računalniških datotek z izračunom entropije stacionarnega diskretnega informacijskega vira. Razvit program izračuna entropijo na podlagi statistične analize pogostosti nizov 8-bitnih znakov (byte) iz abecede z 256 možnimi simboli. Cilj je preučiti entropijo različnih vrst datotek (zvočne, slikovne, besedilne) ter njihovih kompresiranih različic z izgubnimi in brezizgubnimi metodami.

Analiza pokriva različne dolžine nizov (od $n = 1$ do $n = 5$), pri čemer se izračunane vrednosti H_1 do H_5 primerjajo glede na vrsto datoteke in kompresijo. Rezultati ponujajo vpogled v strukturo podatkov in vpliv kompresije na informacijsko vsebino. Namen naloge je poglobiti razumevanje entropije in njenega pomena pri obdelavi in shranjevanju podatkov.

2 Metodologija

Za nalogo je bila uporabljena programska implementacija v Jupyter Notebooku, ki omogoča izračun entropije datotek različnih vrst (besedilnih, zvočnih in slikovnih). Uporabljene so bile knjižnice, kot so `numpy` za interpretacijo nekaterih formatov datotek, `collections.Counter` za štetje pogostosti elementov, `math` za logaritemske operacije ter knjižnica `wave` za obdelavo zvočnih datotek. Razvita je bila funkcija `split_in_tuples`, ki razdeli vsebino datoteke na nize različnih dolžin, le-te pa so določene v klicu funkcije. Implementacija te funkcije omogoča generaliziran pristop k izračunu entropije visjih redov. Poleg funkcije `split_in_tuples` je bila implementirana tudi funkcija `calculate_entropy`, ki na podlagi frekvence ponavljanja posameznih nizov znakov izračuna Shannonovo entropijo. Funkcija uporablja formulo:

$$H = -i \cdot \sum P(i) \cdot log_2 P(i) \tag{1}$$

kjer $P(i)$ predstavlja verjetnost posameznega niza znakov.

Program odpira različne vrste datotek (tekstovne, zvočne in slikovne), jih prebere v binarni obliki in na njih izvede statistično analizo. Izračuni entropije so bili izvedeni na nizih podatkov, dolžine od 1, do 5.

Analizirane so bile razlike med vrednostmi entropij za nekompresirane in kompresirane datoteke, kar je omogočilo vpogled v vpliv strukturiranosti in kompresije podatkov na njihovo informacijsko vsebino.

2.1 Slikovna datoteka

Izvorna slika je bila podana v formatu BMP, ki je nekompresiran format z visoko kakovostjo in velikimi datotekami, primeren za natančno grafično obdelavo. S programom Gimp je datoteka bila stisnjena v datoteko, formata PNG, ki uporablja brezizgubno kompresijo, podpira prosojnost in je idealen za spletne grafike ter logotipe. PNG kompresija temelji na filtriranju in algoritmu DEFLATE [1]. Filtriranje odstrani redundanco tako, da shrani razliko med vrednostjo pikslov in predvideno vrednostjo na podlagi sosednjih pikslov. Nato algoritem DEFLATE stisne podatke z uporabo kombinacije LZ77 [2] in Huffmanovega kodiranja[3], kar omogoča zmanjšanje velikosti datoteke brez izgube kakovosti.. JPG pa z izgubno kompresijo omogoča majhne datoteke, kar je optimalno za fotografije, vendar na račun rahle izgube kakovosti.[1]

2.2 Tekstovna datoteka

Tekstovna datoteka je bila podana v formatu `.txt`, ta vsebuje nekompresirane podatke v obliki surovega teksta, kar pomeni, da je velikost datoteke sorazmerna z dolžino

besedila. Kompresija v format `.ZIP` uporablja algoritem DEFLATE, ki združuje LZ77 za nadomeščanje ponavljajočih se vzorcev s kazalci in Huffmanovo kodiranje za učinkovito shranjevanje znakov. Kompresija zmanjša velikost datoteke brez izgube podatkov, saj shranjuje samo bistvene informacije. Odklenjen ZIP omogoča prost dostop do stisnjene vsebine brez varnostnih omejitev. Zaklenjen ZIP uporablja enak kompresijski algoritem kot odklenjen ZIP, a vključuje šifriranje podatkov z algoritmom, kot je AES[4]. Za dostop do vsebine je potrebno geslo, kar zagotavlja dodatno zaščito pred nepooblaščenim dostopom. Šifriranje varuje stisnjene podatke, ne da bi bistveno vplivalo na velikost datoteke. Kompresija je bila dosežena z uporabo sistemskega orodja `zip`, vgrajenega v operacijski sistem *Linux*.

2.3 Zvočna datoteka

Zvočna datoteka je bila poadana v formatu WAV, to je nekompresiran format, ki zagotavlja maksimalno kakovost zvoka, a ustvarja zelo velike datoteke, primerne za profesionalno obdelavo. FLAC uporablja brezizgubno kompresijo, ki zmanjša velikost datoteke za 40–60 % ob popolni ohranitvi kakovosti, kar je idealno za arhiviranje. FLAC stisne zvok brez izgube podatkov z iskanjem ponavljajočih se vzorcev in napovedovanjem prihodnjih vrednosti signala na podlagi linearne napovedi. Preostale razlike (napake napovedi) se učinkovito kodirajo z algoritmom Golomb-Rice[5], kar zmanjša velikost datoteke. Proces omogoča popolno rekonstrukcijo izvirnega zvoka ob predvajanju. MP3 uporablja izgubno kompresijo, ki močno zmanjša velikost (do 90 %), vendar na račun rahle izgube podrobnosti, zato je najprimernejši za prenos in splošno uporabo. MP3 uporablja algoritme za zmanjšanje velikosti z odstranitvijo podatkov, ki jih človeško uho težko zazna, pri čemer se zvok pretvori v frekvenčni prostor (MDCT). Nato se manj pomembne frekvence zmanjšajo ali odstranijo (perceptual coding)[6], preostali podatki pa se stisnejo z Huffmanovim kodiranjem. Rezultat je močno zmanjšana velikost datoteke z minimalno opazno izgubo kakovosti.

3 Rezultati

Pri obdelavi različnih tipov datotek v različnih formatih so bili izhodi naslednji:

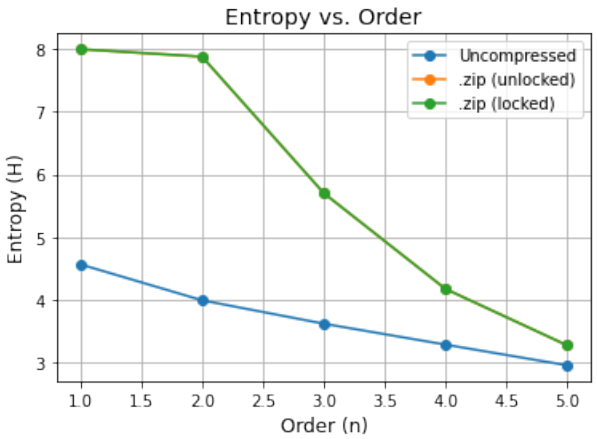
Iz tabele (1) je razvidno, da pri formatu .txt entropija pada z naraščanjem reda, saj višji redi odražajo večjo predvidljivost podatkov zaradi njihove naravne strukture. Na začetku, pri prvem redu, je entropija najvišja (4.57), kar kaže na večjo stopnjo naključnosti, medtem ko pri petem redu pade na 2.96, kar odraža večjo urejenost.

Pri odklenjenem .zip formatu stisnjeni podatki kažejo bistveno višjo začetno entropijo (skoraj 8), saj stiskanje odstrani redundanco in ustvari bolj naključne podatke. Kljub temu entropija pri višjih redih še vedno pada, vendar počasneje kot pri formatu .txt, kar pomeni, da stisnjeni podatki ohranjajo nekaj notranje strukture. Zaklenjen .zip format se pri entropiji skoraj ne razlikuje od odklenjenega, kar nakazuje, da zaklepanje ne spremeni osnovnih statističnih lastnosti podatkov.

Tabela 1: Entropije tekstovnih datotek, reda od 1 do 5, pri različnih formatih

Text entropies		
Format	Order	Entropy
.txt	1	4.5685
	2	3.9952
	3	3.6223
	4	3.2885
	5	2.9587
.zip(unlocked)	1	7.9986
	2	7.8780
	3	5.7050
	4	4.1772
	5	3.2774
.zip(locked)	1	7.9995
	2	7.8815
	3	5.7050
	4	4.1772
	5	3.2774

Skupno gledano rezultati kažejo, da stiskanje znatno poveča začetno entropijo, vendar pri višjih redih razkrije preostale vzorce v podatkih, medtem ko zaklepanje teh vzorcev ne spreminja.



Slika 1: Graf spremembe entropije tekstovnih datotek, v relaciji z redom entropije

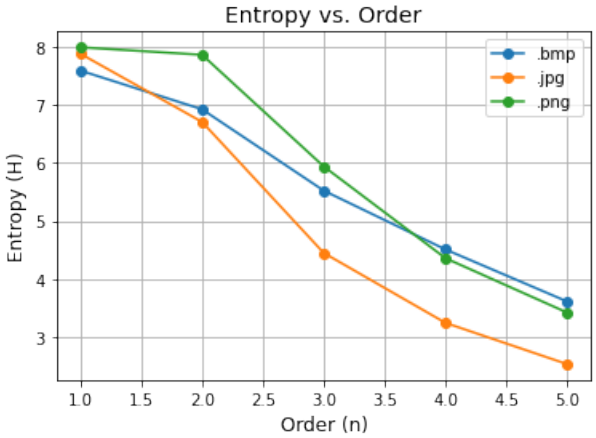
Tabela (2) nakazuje, da ima format .bmp visoko začetno entropijo (7.59 pri 1. redu), ki z višjimi redi izrazito pada (3.62 pri 5. redu), kar kaže na močno strukturo surovih podatkov. Format .jpg zaradi izgubne kompresije dosega višjo začetno entropijo (7.88), ki hitro pada na 2.54 pri 5. redu, saj kompresija odstrani večino redundantnih informacij, a ohrani osnovne vzorce. Format .png ima najvišjo začetno entropijo (7.99), značilno za brezizgubno kompresijo, vendar entropija s povečevanjem reda pada počasneje (3.43 pri 5. redu), kar kaže na ravnotežje med strukturo in optimizacijo podatkov.

Skupno se brezkompresijski formati (.bmp) odlikujejo z ohranjanjem surovih vzorcev, medtem ko kompresija (.jpg, .png) poveča začetno naključnost, a vseeno raz-

Tabela 2: Entropije slikovnih datotek, reda od 1 do 5, pri različnih formatih

Image entropies		
Format	Order	Entropy
.bmp	1	7.5885
	2	6.9223
	3	5.5233
	4	4.5119
	5	3.6165
.jpg	1	7.8780
	2	6.7053
	3	4.4515
	4	3.2490
	5	2.5386
.png	1	7.9894
	2	7.8601
	3	5.9392
	4	4.3620
	5	3.4257

kriva preostale strukture pri višjih redih.



Slika 2: Graf spremembe entropije slikovnih datotek, v relaciji z redom entropije

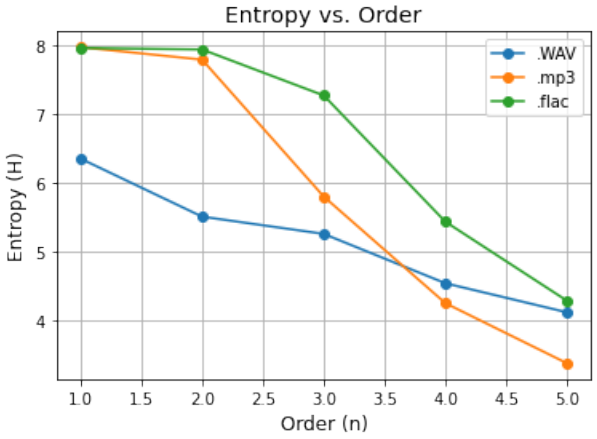
Iz tabele (3) je razvidno, da ima format .wav začetno entropijo 6.35, ki z višjimi redi postopoma pada na 4.12, kar odraža prepoznavne vzorce v surovih podatkih.

Format .mp3, z izgubno kompresijo, dosega višjo začetno entropijo (7.97), vendar ta hitro pada na 3.38 pri 5. redu, saj kompresija močno odstrani redundantne informacije. Format .flac, kot brezizgubna kompresija, ima podobno začetno entropijo kot .mp3 (7.96), vendar z višjimi redi pada počasneje (4.28 pri 5. redu), kar kaže na boljšo ohranitev podrobnosti podatkov.

Brezkompresijski formati (.wav) imajo nižjo začetno naključnost in bolj jasne vzorce, izgubni formati (.mp3) povečajo začetno naključnost, medtem ko .flac ohranja ravnotežje med naključnostjo in strukturo.

Tabela 3: Entropije zvočnih datotek, reda od 1 do 5, pri različnih formatih

Audio entropies		
Format	Order	Entropy
.wav	1	6.3524
	2	5.5080
	3	5.2569
	4	4.5394
	5	4.1163
.mp3	1	7.9716
	2	7.7903
	3	5.8000
	4	4.2497
	5	3.3757
.flac	1	7.9595
	2	7.9382
	3	7.2692
	4	5.4339
	5	4.2839



Slika 3: Graf spremembe entropije zvocnih datotek, v relaciji z redom entropije

Literatura

[1] Wikipedia contributors, "Deflate — Wikipedia, the free encyclopedia," 2024. [Online; accessed 16-November-2024].

[2] Wikipedia contributors, "Lz77 and lz78 — Wikipedia, the free encyclopedia," 2024. [Online; accessed 16-November-2024].

[3] Wikipedia contributors, "Huffman coding — Wikipedia, the free encyclopedia," 2024. [Online; accessed 16-November-2024].

[4] Wikipedia contributors, "Advanced encryption standard — Wikipedia, the free encyclopedia," 2024. [Online; accessed 16-November-2024].

[5] Wikipedia contributors, "Golomb coding — Wikipedia, the free encyclopedia," 2024. [Online; accessed 17-November-2024].

[6] Wikipedia contributors, “Perceptual audio coder — Wikipedia, the free encyclopedia,” 2024. [Online; accessed 17-November-2024].