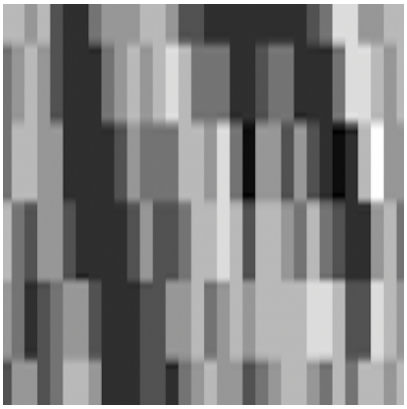


By the same token

Description

Automated natural language processing (NLP) of the kind deployed in conversational AI typically breaks blocks of text not just into words, but into *tokens*. Tokens are strings of characters extracted from a corpus of texts â?? the set of texts an NLP system is trained on.

Tokens are more efficient than storing just whole words. So, the word â??**oligarch**â?• is made of two tokens â??**olig**â?• and â??**arch**â??. The string â??**olig**â?• also appears in â??**olig**opolyâ?• and several other words. The string â??**arch**â?• appears in other words, such as â??**arch**way,â?• â??an **arch**y,â?• â??**arch**itecture.â?• Depending on how frequently these words appear in the training data, it may be more efficient to store and process these word fragment tokens rather than whole words, especially as a word may have several different endings â?? â??**s**,â?• â??**ies**â??, â??**ed**,â?• etc, which are also tokens.

Urban tokens

As well as conversational AI, Iâ??m interested in the urban world of people, bricks, concrete and other material things. What is our everyday experience of tokenization? Breaking the world into parts is an everyday, culturally bound practice. It is strongly influenced by language and contributes to the formation of language. Itâ??s well known that what we identify as an object, thing, entity, phenomenon, or event in any circumstance derives from practical engagement with our lifeworld, even if we donâ??t have words for everything we are using or seeing.

Where we street-life neophytes see splashes of coloured paint on a wall as just vandalism, the well-informed see an artful composition of â??tags,â?• â??throwsâ?• and â??pieces. What we see depends on our approach to the world of graffiti and our inculcation into its language practices. So too, the thing you trip over on the pavement emerges for the moment as a crack, a misplaced slab or a protruding pipe. Buildings, roads, parks are similarly defined and bounded depending on whether your perception is framed by ownership, servicing, tourism, photography, catastrophe, complaint or flights of imagination. I draw here on the phenomenology of Martin Heidegger, for whom things emerge as objects in our environment in the practical context of concerned dealings in the lifeworld. See post

Nothing as it seems.

Image recognition

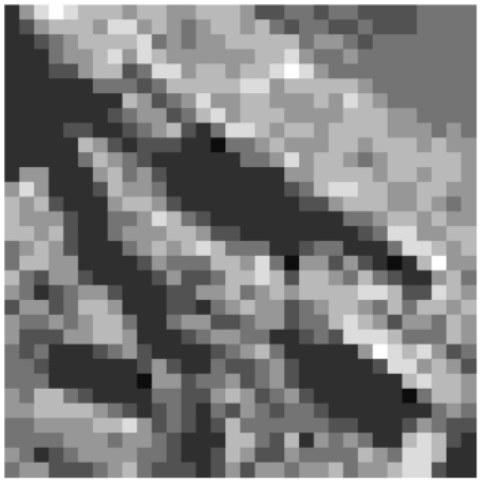
One of the key challenges of automating the recognition of objects in photographs is to identify objects (human beings, plants, animals, vehicles, roadsigns) in a scene that are of interest to someone and to some end: e.g., for navigation in self-drive cars.

Tokenisation is an operation deployed in early automated image recognition techniques. An image tokeniser would search for chunks of pixels that constitute patterns. These pixel patterns may recur in the same image, or across different images.

Cities are more than pictures, but maps and aerial photographs are well-established as means of relating cities to images. After all, things get built in plan and urban spatial data is commonly plotted on a ground plane.

Tokenising map data

To illustrate tokenization of mapped data (i.e. aerial images), I took a parcel of Edinburgh-Leithâ??s docklands, reduced it to a 32Ã?32 pixel grid with a 10 value grey scale (0-9) with 0 as black, 9 as white.

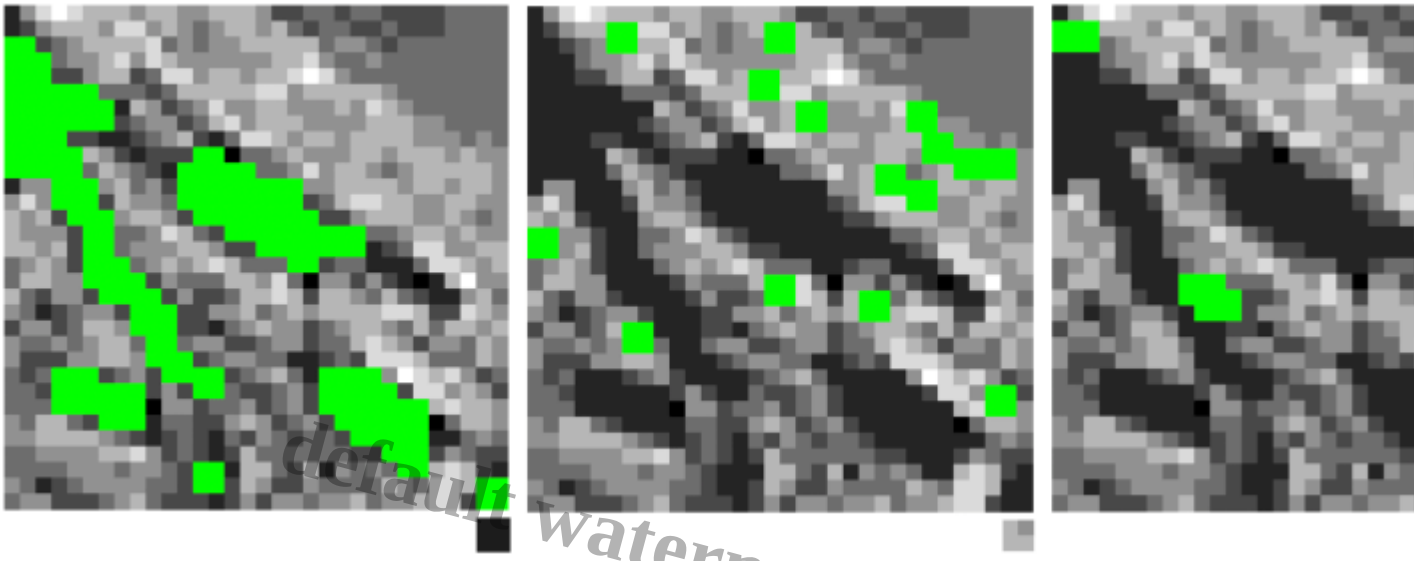


Though itâ??s hard for a human interpreter to read such a patchwork of greys, there are patterns in this data, and they repeat across the image or across several images of different places.

With help from ChatGPT4 I created a program in Python that searches the matrix of 0-9 pixel values for commonly occurring patterns. There are 1,024 pixels in the image. Instances of just 0s, 1s, 2, etc do not constitute patterns, but two 0s next to each other might if they recur. There are no 1Ã?2 (00) patches in this image though. The first â??interestingâ?• pattern is made of 2Ã?2 patterns consisting of just 1s (1111) highlighted as green patches in the first image below.

Patterns of varying pixel values are potentially of greater interest as they could indicate recurring features in the landscape, such as a boat berthed at a wharf, vegetation, a shoreline. I wonâ??t attempt to interpret what these are in the second 2Ã?2 (5455) image, nor in the third image, which identifies

larger patches of 2Ã?3 (124112). We would need higher image resolution, some means of matching relative lightness values across different spectra, and a machine learning system trained to identify such features. Here I am interested just in the computationally intensive process of identifying recurring patterns as tokens.



Patches larger than 3Ã?3 donâ??t seem to reveal any repetitions across this image. Note also that itâ??s unlikely that patterns that are not contiguous will yield any tokens of interest, though image tokens donâ??t need to all be rectangular.

As an indication of the scale of the computation for just 2Ã?2 patterns, here are all the 2Ã?2 pixel combinations that exist in the image and appear more than once. The number in brackets is their frequency. This is a subset of the 10^4 possible combinations of 2Ã?2 pixels.

1111 (104), 3333 (36), 1121 (19), 5555 (17), 1211 (16), 5455 (15), 5545 (14), 5445 (11), 4544 (10), 5554 (9), 5454 (9), 2332 (8), 4555 (8), 4444 (8), 5544 (8), 4455 (8), 3343 (8), 5534 (7), 4545 (7), 4445 (7), 1311 (7), 4554 (7), 3444 (6), 4433 (6), 3422 (6), 4454 (6), 3433 (6), 3423 (6), 4434 (6), 4254 (6), 4534 (5), 3233 (5), 2323 (5), 5355 (5), 5556 (5), 4344 (5), 4333 (5), 4354 (5), 1112 (5), 2142 (5), 4423 (5), 2121 (5), 1131 (5), 2412 (4), 4523 (4), 5434 (4), 3454 (4), 4334 (4), 4464 (4), 4355 (4), 2342 (4), 5444 (4), 1122 (4), 2312 (4), 1142 (4), 2111 (4), 5443 (4), 2311 (4), 4234 (4), 1412 (3), 6555 (3), 5546 (3), 5466 (3), 3323 (3), 3322 (3), 2222 (3), 6456 (3), 3434 (3), 3334 (3), 2333 (3), 1312 (3), 2211 (3), 2411 (3), 5566 (3), 4244 (3), 2221 (3), 1011 (3), 5354 (3), 4243 (3), 4265 (3), 1132 (3), 3412 (3), 6535 (3), 4324 (3), 2443 (3), 1242 (3), 3232 (3), 6745 (2), 7655 (2), 4443 (2), 4245 (2), 2253 (2), 3223 (2), 6655 (2), 3255 (2), 2233 (2), 5564 (2), 6365 (2), 3243 (2), 4513 (2), 5533 (2), 6455 (2), 5664 (2), 3354 (2), 3311 (2), 3254 (2), 3523 (2), 5465 (2), 3344 (2), 5565 (2), 2154 (2), 1310 (2), 4345 (2), 5435 (2), 2143 (2), 0311 (2), 5446 (2), 4353 (2), 3234 (2), 3411 (2), 6444 (2), 2131 (2), 4535 (2), 6445 (2), 3142 (2), 4533 (2), 2153 (2), 6545 (2), 3312 (2), 1133 (2), 1123 (2), 3435 (2), 2141 (2), 2344 (2), 3342 (2), 6615 (2), 5424 (2), 4142 (2), 2144 (2), 3545 (2), 5634 (2), 4565 (2), 3144 (2), 1212 (2), 5442 (2), 2343 (2), 3534 (2), 1213 (2), 2234 (2), 4242 (2), 3542 (2), 5543 (2), 2244 (2), 6566 (2), 2453 (2), 4232 (2), 3345 (2), 2611 (2), 2322 (2), 1143 (2), 2132 (2), 3353 (2), 3332 (2), 3222 (2)

To be exhaustive, a program would need to generate combinations for 3Ã?1, 1Ã?3, 2Ã?3, 3Ã?2, etc up to about 5Ã?5. It would also need to look for patterns in shapes other than rectangles, e.g. L-shaped patches of pixels. Identifying tokens in strings of text is simpler than in our 2D urban world.

My objective here is to show that as a key element of natural language processing tokenisation also belongs to human perception, automated image analysis and hence urban contexts.

Text tokenization

Consider the string of words: â??archival oligarch anarchic research.â?• Text tokenisation involves identifying patterns in 1 dimension rather than in 2 dimensions, and including the whole alphanumeric character set rather than pixel lightness values. Using the same program I was able to identify tokens of 4 characters in length â?? effectively grid units in alphanumeric space of 1Ã?4. As indicated in the following trace of the token options, the string â??**arch**â?• is the best candidate for a â??superâ?• token in this modest word set.

> arch (4), rchi (2), chiv (1), hiva (1), ival (1), olig (1), liga (1), igar (1), garc (1), anar (1), narc (1), chic (1), rese (1), esea (1), sear (1), earc (1)

There are several criteria for calculating tokens for a more realistic, deep learning training corpus. The lexicon of tokens should be comprehensive enough to reconstruct any substring in the corpus. Ideally, the complete set of tokens should not exceed the set of all dictionary words or symbols derived from the corpus. There can be some redundancy among the token set and this can help in capturing different nuances or grammatical aspects. Tokens can vary significantly in length, ranging from a single character to a group of words.

The tokenization procedure should break down a text string with a granularity that enables efficiency and simplicity balanced with the need to maintain nuances in the original text. Multiword expressions or compound words should preferably be kept as single tokens to retain their specific uses in sentences. The tokenization process should be designed to be computationally efficient. The approach to tokenization should be adaptable to different tasks: conversational AI, text generation in specific domains, translation, etc.

While experimenting with automated text tokenization, I discovered that there are three tokenization modules in common usage and able to be accessed via Python programming: Byte-Pair Encoding (BPE), SentencePiece and WordPiece. They typically sort words in length order starting with the longest, calculate word frequency. Words with high frequency become tokens. For lower frequency words search for the occurrence of each substring, 1, 2, 3 and more characters at a time. For very large corpora of training data, the process may sample the text to derive the initial lexicon of tokens. The process is iterative and may merge tokens tokens.

To repeat, the tokenisation process, so important to automated natural language processing, mirrors the automated analysis of images, which in turn represents an attempt to mimic human perception and cognition, in turn shadowing processes critical in the perception and practical engagement with environment.

Category

1. Artificial Intelligence

Tags

1. Image recognition
2. map data

Date Created
September 30, 2023
Author
rcoyne99