

10/11/24

Video Link :-

<https://youtu.be/zduSFxRajkE?list=PLAqhlrjkbWl23v9cThsA9GvCAUhRvKZ>

Tokenizer - Andrej Karpathy

tokenizing :-

- converting every possible character into a token (Integer)
- Using an embedding table

(Ex) - if there are 65 possible tokens, then in the embedding table there are 65 tables. These rows are trainable parameters [Using back propagation] - these vectors are fed into Transformers.

Tokenization → is the process for translating strings or texts into sequence of tokens and Vice Versa

"Tik tokenizer - vocab app" → Line tokenizer count

→ encodings are better than unicoders.
↳ doesn't change much

↳ they change as time

UTF 8, UTF 16, UTF 32

- most common
 - Variable length
 - backward compatible for ASCII encoding
- fixed length
has multiple zeros and not desirable

→ Byte Pair Encoding Algorithm → allows to compress the byte sequences of the UTF-8 encodings

→ The byte encoding can be directly used in the transformers but, the attention calculations will be very huge → it will be tokenization free. [Good]

Algorithm :-

- ① find pair of tokens that occur frequently
- ② Replace that pair with a single new token
- ③ Repeat the step ① and ② again till the last pairs are found.

Picture (upload)

```
aaabdaaabc
```

The byte pair "aa" occurs most often, so it will be replaced by a byte that is not used in the data, such as "Z". Now there is the following data and replacement table:

```
ZabdZabac
Z=aa
```

Then the process is repeated with byte pair "ab", replacing it with "Y":

```
ZYdZYac
Y=ab
Z=aa
```

The only literal byte pair left occurs only once, and the encoding might stop here. Alternatively, the process could continue with [recursive](#) byte pair encoding, replacing "ZY" with "X":

```
XdXac
X=ZY
Y=ab
Z=aa
```

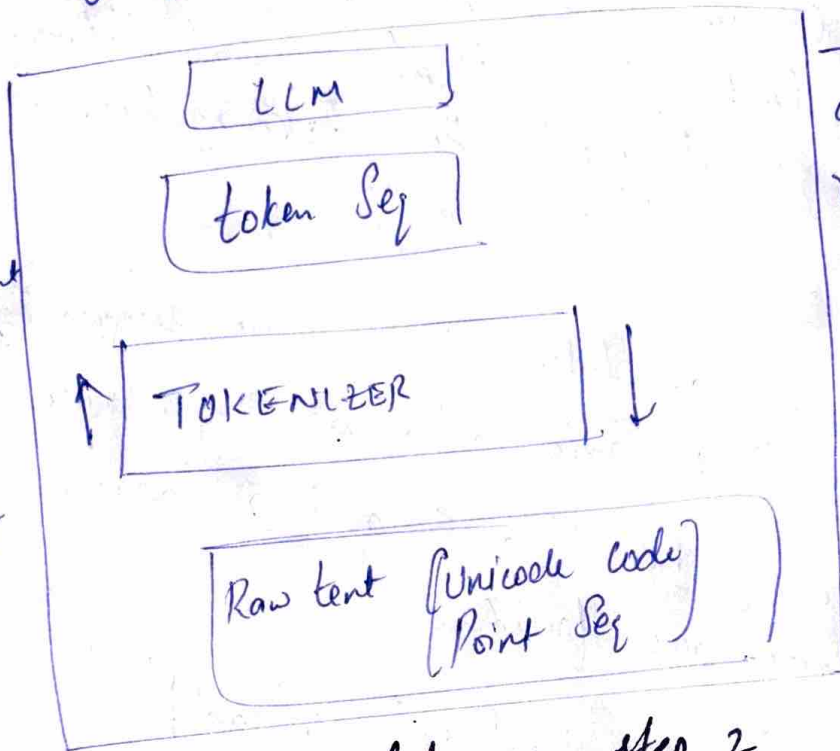
This data cannot be compressed further by byte pair encoding because there are no pairs of bytes that occur more than once.

To decompress the data, simply perform the replacements in the reverse order.

→ tokenizer is a completely separate object from the large language model itself

①

→ tokenizer is completely separate independent module from LLM
→ has its own training dataset of text on which BPE algorithm is applied



②

→ translates back and forth b/w raw text and sequence of tokens

→ language model is trained later as step 2

decoding :-

Vocab : mapping/dictionary in python
→ mapping id to bytes object of that token.

→ (0, 255)

→ tokens at this stage are raw bytes, need to decode those using UTF 8

Text → Alphabet

Encoding :-

1. get raw bytes of the tokens after merged [BPEmerged]
2. get the pair with min merges

(CODE the tutorial) **ToDo**

BPE → Practical middle ground between character and word level LM, that interpolates between **word level inputs for frequent symbol sequences** and **character level inputs for infrequent sequences**.

→ BPE operates on **Unicode points** not byte sequences

→ Tokenizer can do both encode and decode

GPT2 - enforce few rules so that some parts of the vocabulary will not get merged.
Written that rules in the Encoder class under
"Self.pat = Regen code"

[Tokenization
BPE algorithm
modification]

Regen → extension of re

→ It will take the pattern and try to match with the strings (left to right), will get all occurrences and put in a list.

→ Tokenization in GPT works differently for uppercase and lower case

tik token - Open AI's official tokenization library

- Changed the size in GPT 4 compared to GPT 2
(equivalent, executes faster)

Vocab size \rightarrow 50k to 100k
(GPT2) (GPT4)

<|end of text|> Special Tokens.

These special tokens are fully a single token

<|im_start|> , <|im_end|>

"CLook-base" \rightarrow OpenAI encoding

\rightarrow Transformer model surgery is required when the special tokens are added to the model. Increase the vocab size

ToDo Exercise :- minbpe exercise in Karpathy's Github
BPE Algorithm | Karpathy/minbpe (Training a tokenizer)

<https://github.com/hrithiksagar/minbpe-exercise-tokenization>

Link :-

Sentence Piece - Can train and inference BPE tokenization
(Used in llama, mistral)
• Runs BPE on Unicode Points Directly
• encodes with utf8 and then encodes the raw bytes instead

(Q) Why can't Vocab size = infinite?

- (A) {
- token embedding table is going to grow
 - linear layer is going to grow
 - lot of computation will be going on

- (2) {
- Because we have more parameters, we are going to train these parameters
 - model will not have time to think per some number of characters in text.

Solid GoldMagikarp : observed that there are few chunks of tokens are very weird.
the output when asked about those tokens leads to weird outputs such as hallucinations. (trigger words)

↓
reddit user
[who used to post a lot]

undefined never before seen
data
appeared in forward pass, but never trained in backward pass