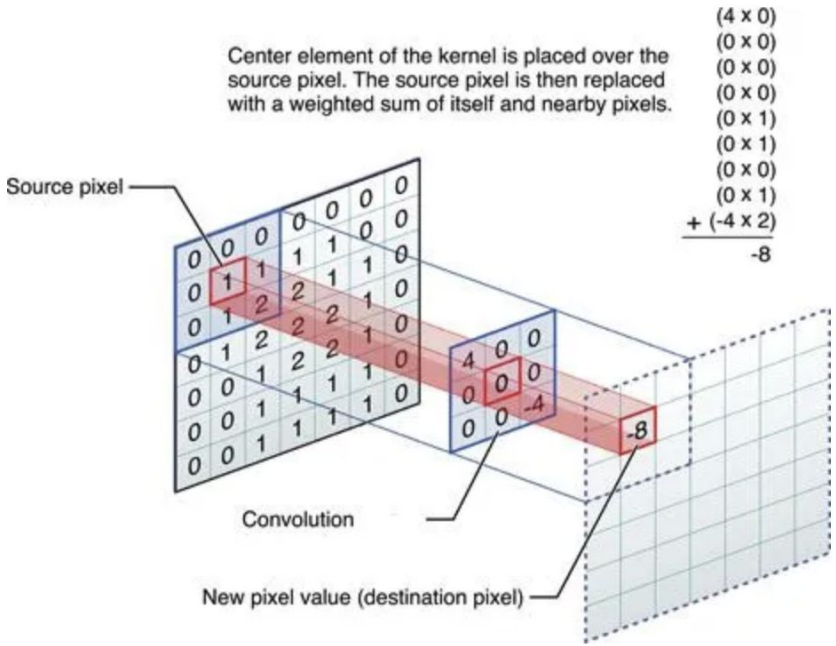
Convolution Style

Group 1

Convolution



Convolution Operation on a 7×7 matrix with a 3×3 kernel

https://medium.com/@bdhuma/6-basic-things-to-know-about-convolution-daef5e1bc411

Convolution "Style"

- Convolution is creatively applied to count the frequency of words in a file
- They use a Convolutional Neural Network with fixed (untrained) parameters

How it works

- Each word is assigned an integer
- The length of the binary representation of the largest integer is found: BIN_SIZE
- Matrix x is created with the binary reps of each word
  - x[i] gets the ith word's binary representation as a vector of 1s and 0s
  - Serves as the input
- Unique words are identified
- A 'filter' 1xBIN_SIZE is made for each unique word
  - When a filter is convolved with its corresponding word in x, the output is 1. Otherwise, the convolution output is <1.
  - These filters are added to a convolutional 'layer'
- Every filter is convolved with x, generating a sum of matching words each time
- Most frequent words are outputted along with their frequency

How it works - example

- Suppose 'hello' = 001, 'convolution' = 100, and input.txt = "hello convolution hello"
- x = [[001],
  [100]
  [001]]
- Filter for hello = [-1/2, -1/2, 1/1], filter for convolution = [1/1, -1/2, -1/2]
- Convolution for 'hello' filter:
  - Pos0: -1/2 * 0 + -1/2 * 0 + 1/1 * 1 = 1 (match, add 1 to output)
  - Pos1: -1/2 * 1 + -1/2 * 0 + 1/1 * 0 = -1/2 (does not match, set to 0 for output)
  - Pos2: -1/2 * 0 + -1/2 * 0 + 1/1 * 1 = 1 (match, add 1 to output)
  - Output: [1, 0, 1]
  - Sum to give frequency of 2
- Convolution for 'convolution' filter:
  - Pos0: -1/2; Pos1: 1; Pos2: -1/2; Output = [0, 1, 0]; Sum = 1

Output of Initial Python code

- Input was a text file containing Pride and Prejudice by Jane Austen

Model: "sequential"

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 56615, 1, 6398)	89,572
re_lu (ReLU)	(None, 56615, 1, 6398)	0
lambda (Lambda)	(None, 1, 6398)	0
reshape (Reshape)	(None, 6398)	0

Total params: 89,572 (349.89 KB)
Trainable params: 89,572 (349.89 KB)
Non-trainable params: 0 (0.00 B)
1/1 ━━━━━━━━━━━━━━━━━━━━ **3s** 3s/step

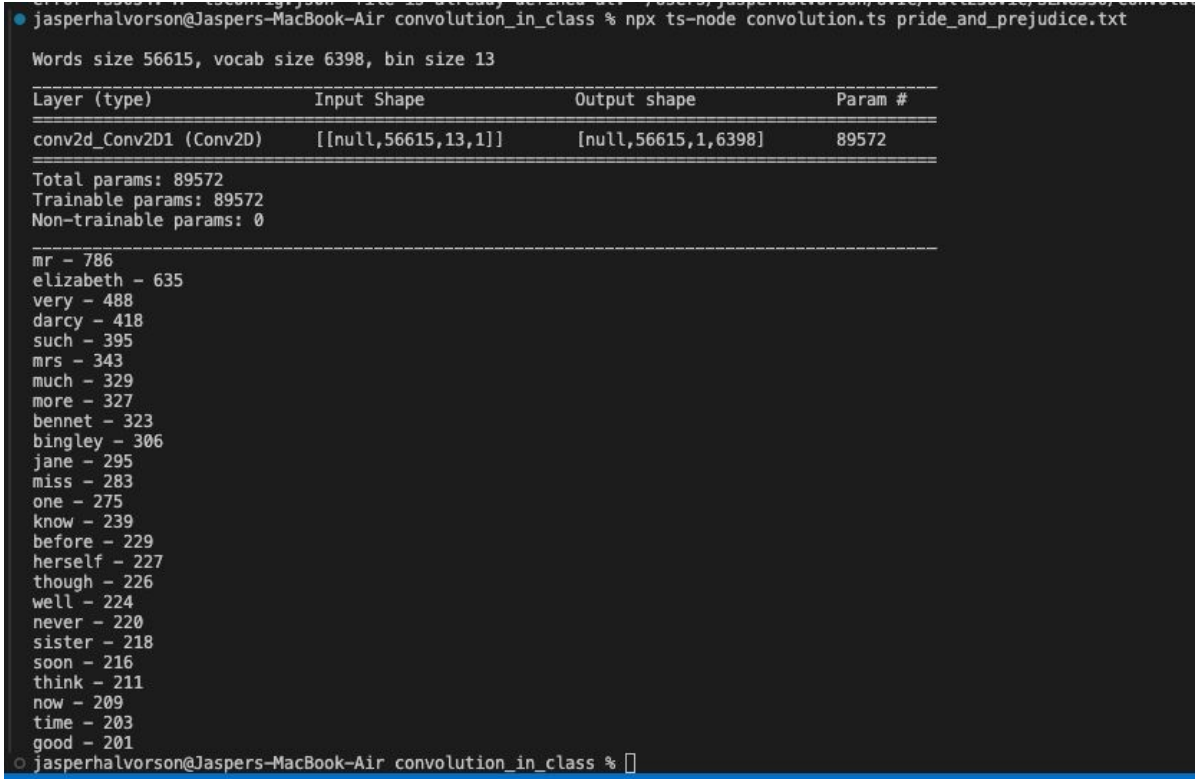
```
mr - 786.0
elizabeth - 635.0
very - 488.0
darcy - 418.0
such - 395.0
mrs - 343.0
much - 329.0
more - 327.0
bennet - 323.0
bingley - 306.0
jane - 295.0
miss - 283.0
one - 275.0
know - 239.0
before - 229.0
herself - 227.0
though - 226.0
well - 224.0
never - 220.0
sister - 218.0
soon - 216.0
think - 211.0
now - 209.0
time - 203.0
good - 201.0
```

Convert Initial Code to TypeScript

- Prompt used: "Please convert the following code to TypeScript:\n <python code>"
- ML/AI is not generally done in TypeScript
- Around 70 lines in Python and around 230 in TypeScript

TypeScript Output

- Same input as used for Python file
- Produces the same output as the Python script



Convolution Style Conclusion

- Convolution has a wide range of applications
- The word frequency problem has several easier solutions