Memory Management for Large Datasets in Machine Learning

Sara Davis
Graduate Student Lesson
CS 646

Students will be able to:

- List common methods for managing large datasets in memory
- Compare and contrast methods for managing large datasets in memory

You should remember from previous lectures:

- Memory is finite
- Generalized techniques for fitting data in memory:
  - Swapping
  - Paging

Real world problems:

- Machine learning needs lots of data
- The more data you have, the more memory you need
  - Especially RAM

So how do we store all of that information in memory if we have a lot of data?

- Scenario:
  - You have a dataset (2 GB memory)
  - A deep learning model (2-3 GB memory)
  - You have 4 GB of RAM on your laptop
  - What’s the problem with the usage?

Implement Chunking!

- Very common
- Cut the dataset into smaller chunks, and only load what will fit in memory at one time
- Perform machine learning on the chunk, get score/metric/value
- Perform machine learning on next chunk, get score/metric/value… and so on…
- Perform pooling, averaging, or voting to determine what the score was over the whole dataset

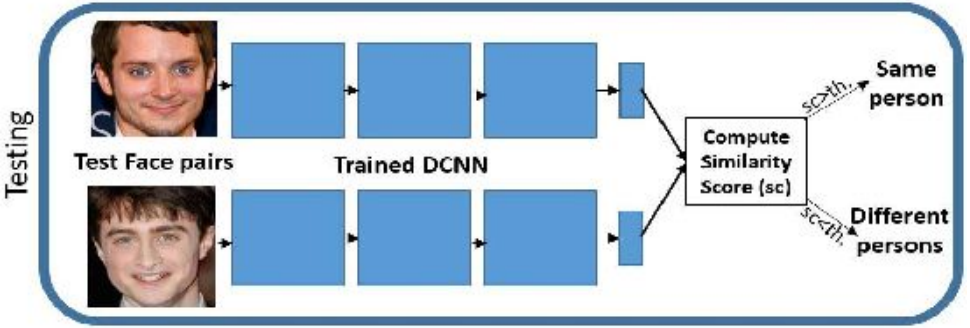
Implement Indexing!

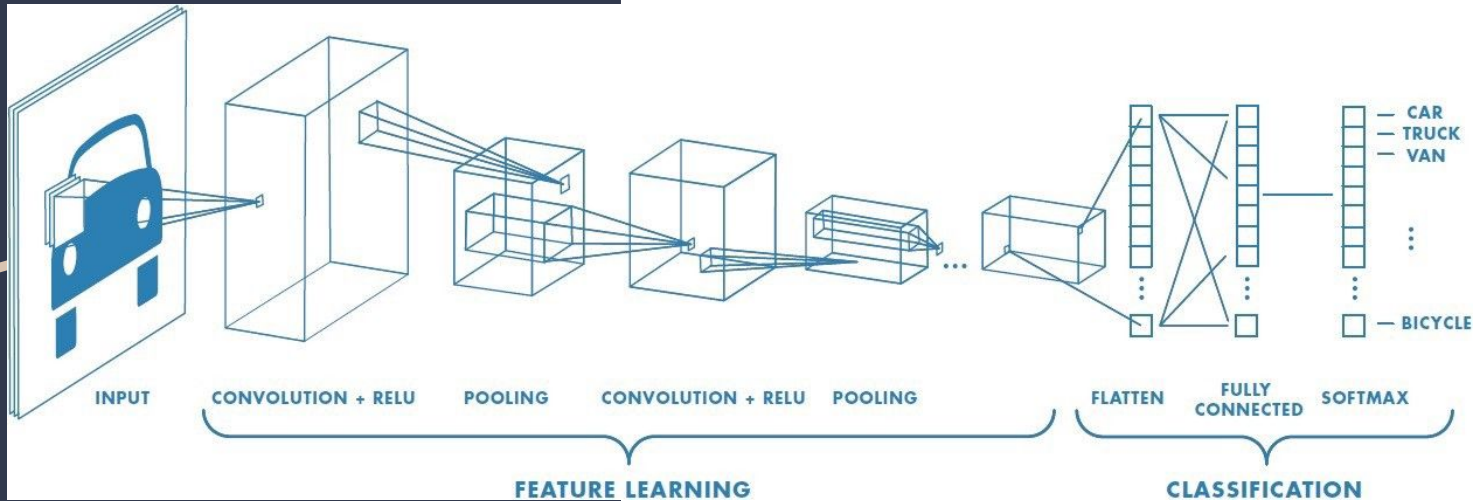
- Similar to hashing/paging
- You associate each element in your dataset with some int, and only pull the associated element when you need it to train

Chunking vs. Indexing- useful when

- Chunking: need a quick implementation, need to use less memory than compression but more than indexing
- Indexing: memory overhead is a concern
- Sometimes, it may be appropriate to apply both

Quick Application: Face Verification using Convolutional Neural Networks





In Summary

- Chunking = load the data in chunks
- Indexing = load a value associated with the data, then access the data using a table
- It’s totally acceptable to combine methods, and other methods do exist

Questions?