Deep Learning

Or why you should just ask a computer to figure it out.

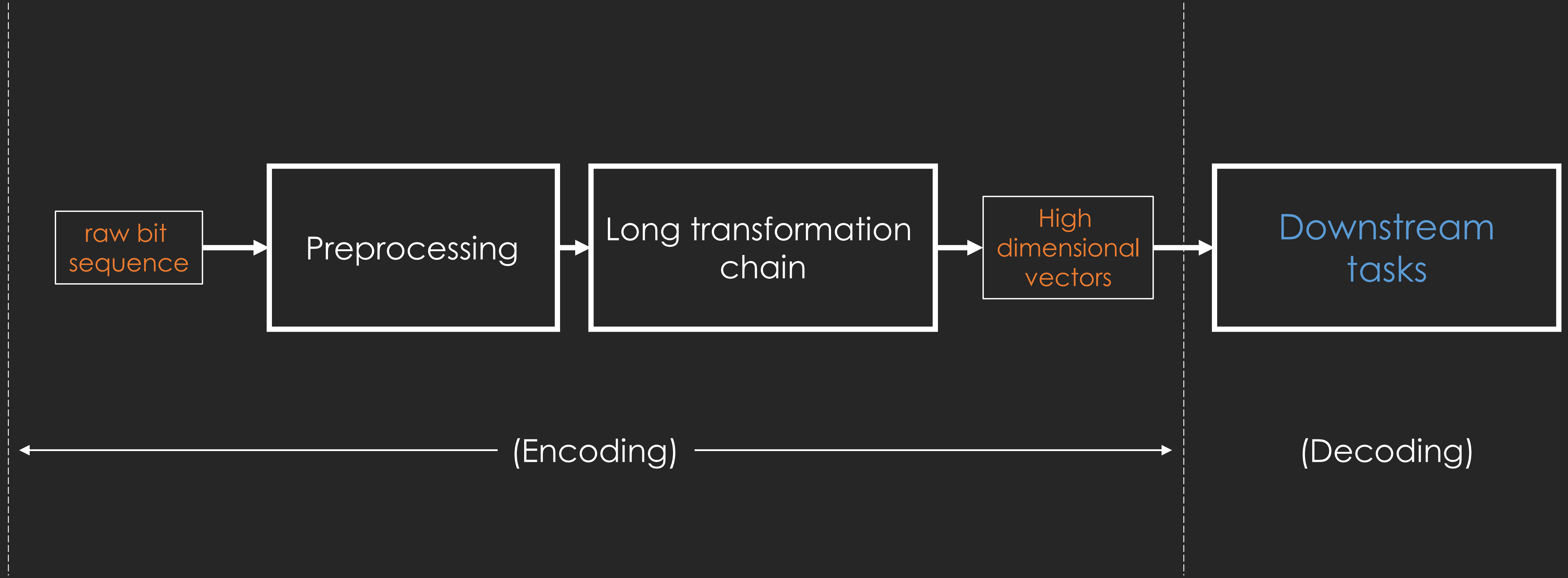
Deep Learning

The worst explanation ever

A long chain of linear and non-linear tensor (read: matrix) transformations to transform input to output.

Note: Read chapter 1 of d2l.ai for more context.

A Graphical Explanation



raw bit sequence

Preprocessing

Long transformation chain

High dimensional vectors

Downstream tasks

(Encoding)

(Decoding)

When should Deep Learning be used?

When the behavior of the system cannot be reduced to a finite(and manageable) set of rules.

What are the ingredients?

Data

Model

Objective function

Algorithm

Applications

A very incomplete list.

Natural language Processing (NLP)

(RNNs and Transformers)

1. Text classification
2. Text generation
3. Summarization/ Extraction

Taking RNNs further:

1. Time series analysis
2. Audio Analysis

Computer Vision (CV)

(CNNs, GANs, UNets, etc)

1. Image/video classification
2. Segmentation
3. Object detection
4. Style Transfer
5. Generation
6. Super resolution

Taking CV further:

1. Audio Analysis (Frequency spectrum analysis)

Dataset Sources

1. AWS datasets (link)
2. Google dataset search (link)
3. Kaggle (link)
4. DrivenData (link)

Note: You **MUST** find a dataset set and problem you’re interested in, or you’ll get bored.

PyTorch vs. TensorFlow

Somewhat subjective at certain points

PyTorch

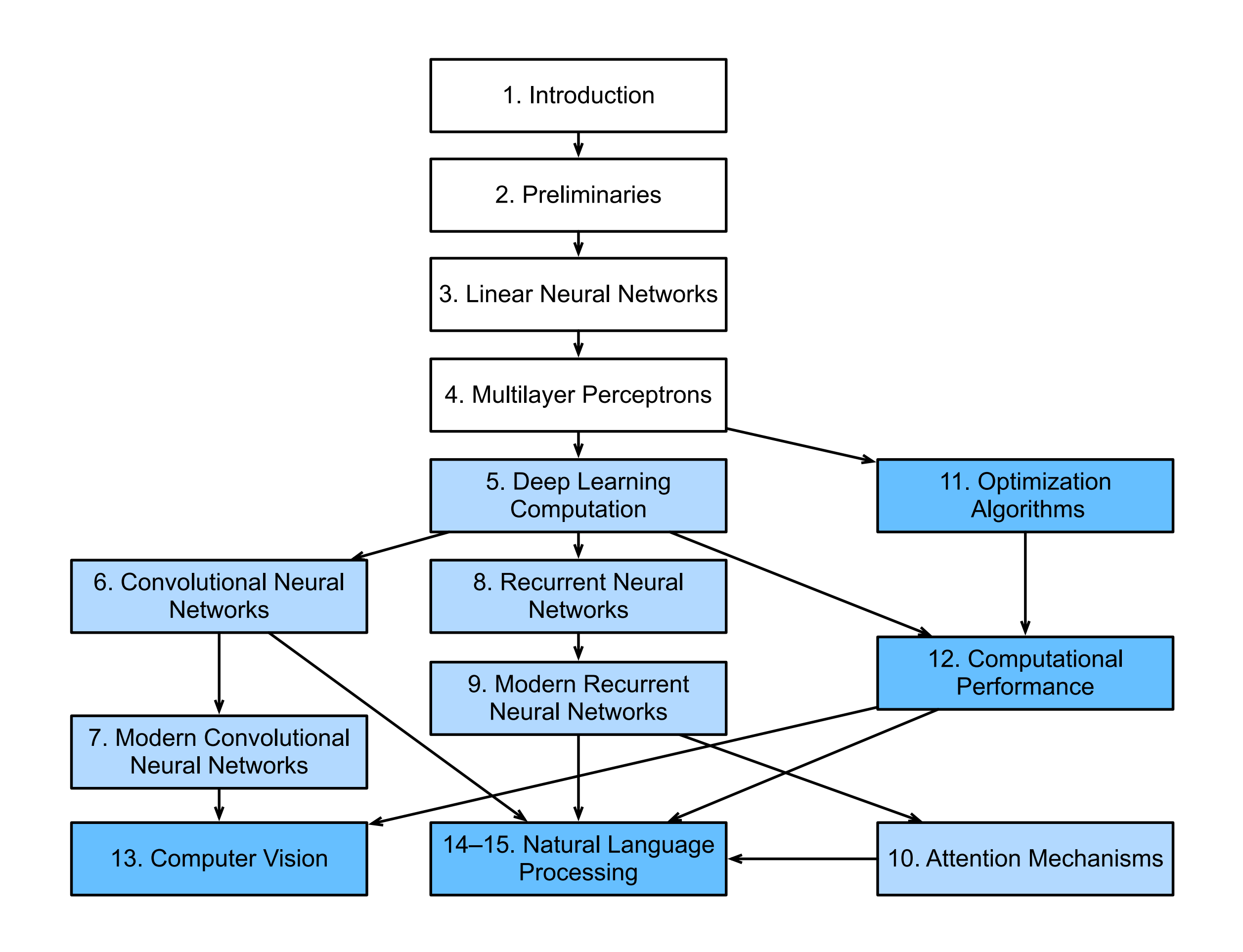
1. More pythonic (good class definitions)
2. Easier to write (subjective?)
3. Not so much diversity in platform support
4. Serving libraries are newer

TensorFlow

1. Less pythonic (functions separated from objects)
2. Harder to write (?)
3. More diverse platform support (.js, embedded)
4. More mature serving libraries

Note: I will be using PyTorch, but it is important to learn both, since it will allow you to read more repos.

Pathway


1. Introduction
2. Preliminaries
3. Linear Neural Networks
4. Multilayer Perceptrons
5. Deep Learning Computation
11. Optimization Algorithms
6. Convolutional Neural Networks
8. Recurrent Neural Networks
12. Computational Performance
7. Modern Convolutional Neural Networks
9. Modern Recurrent Neural Networks
13. Computer Vision
14–15. Natural Language Processing
10. Attention Mechanisms

1. Week 1 – Chapter 2, 3,
2. Week 2 – Chapter 4, 5
3. Week 3 – Chapter 6, 7
4. Week 4 – Chapter 8, 9
5. Week 5 – Chapter 10, 11,
6. Week 6 – Chapter 12
7. Week 7 – Chapter 13
8. Week 8 – Chapter 14, 15
9. Week 9 – Chapter 16,
10. Week 10 – Chapter 17

Note: Image source – d2l.ai

We will use and build on this reference for almost the entire study of Deep Learning: https://d2l.ai/

The Basics

What do you need to know?

Some **Python** (probably easier that way)

References

Look here first:

https://docs.google.com/document/d/1Cw6fA5Ne7gGnLA2NM_Fy1KJuOZEBW5JToXSvFfxl_Lw/edit?usp=sharing

Ask me/host for the slides if you want them.
It will probably **end up on some git repo** along with the code.

Repo:
https://github.com/kradkahaddi/snakesandmachines

The Basics

Method

I will (usually) introduce certain topics via recorded videos and host further explanation on weekends. (recorded)

A volunteer can revise the topics and dig deeper during the following week (recorded)

It will be good for discussions if everyone has read the material and perspectives to share and questions to ask

Opensource license for all additional code and materials: MIT or similar
(free usage, distribution, and modification)

The Basics

What do you need to Volunteer?

1. A better than zero understanding of the topic.
   My example (could be better for sure): https://www.youtube.com/watch?v=w46AX1jQhjM

2. Some Prep
   1. The **main jupyter notebook** with properly flowing code sections.
   2. A **requirements.txt** for your code environment.
   3. (Optional) A second code notebook if you want to do scratch work
   4. (Optional) A set of .py files to support as imports.
   5. (Optional) A slide deck if you want to give an overview of the topic before heading to the code.

3. A Sanity Check
   1. Don't over think what you're going to do.
   2. Ask someone to review your work before you present.
   3. Nobody succeeds alone, be open to receiving help.

This command should work for most environments:

```
pip freeze > requirements.txt
```

Note: You **MUST** have the main jupyter notebook and requirements.txt to present.