02 | Building Blocks of Deep Learning

Microsoft

Stephen F Elston | Principle Consultant, Quantia Analytics, LLC

Background for Deep Learning

- Review of multi-class distributions
- Tensors for deep learning
- Basics of image data
- Introduction to Keras

Review of Binary Classification

Bernoulli distribution for probability of success

Observation: ν = 1:

$$p(\nu = 1) = \Theta$$

$$\textit{where}$$

$$\nu = \textit{an observation}$$

$$\Theta = \textit{probability parameter}$$

Review of Binary Classification

We extend the Bernoulli distribution for multiple trials with the **Binomial distribution** for k successes in n trials:

$$p(\nu = k \mid \Theta) = \binom{n}{k} \Theta^k (1 - \Theta)^{n-k}$$

Were the Binomial coefficient, pronounced n choose k is:

$$\binom{n}{k}$$

Review of Binary Classification

How do perform classification with the Bernoulli distribution?

Use the **logistic** or **sigmoid** function

$$\sigma(x) = \frac{L}{1 + e^{-k(x_0 - x)}}$$

$$where$$

$$L = max\ value$$

$$k = slope$$

$$x_0 = sigmoid\ midpoint$$

Review of Binary Classification

Simplify the logistic function if k = 1, L = 1 and x_o = 0:

$$\sigma(x) = \frac{1}{1 + e^{-x}} = \frac{e^x}{1 + e^x}$$

Classification with the Categorical Distribution

- What are the distribution for multi-class problem?
- Use the **Categorical distribution** with probability mass function:

$$p(x = i \mid \boldsymbol{\theta}) = \theta_i$$

Where:

$$\boldsymbol{\theta} = (\theta_1, \ldots, \theta_k)$$

And:

$$\sum_{i=1}^{k} \theta_i = 1.0$$

Classification with the Categorical Distribution

- How do we create a categorical classifier?
- Use a softmax function:

$$\sigma(z_i) = \frac{e^{z_i}}{\sum_{k=1}^{K} e^{z_k}}$$

The normalization, $\sum_{k=1}^{K} e^{z_k}$, ensures the probabilities sum to 1.0

Classification with the Categorical Distribution

- What is the output of softmax?
- One value for each category
  - For example, if we have 10 categories, there are 10 output values
  - Take the max as the most probable category
- Label must be one-hot encoded
  - Binary value for each possible category
  - Only one 1, others 0

Tensors for Deep Learning

For deep learning models we need a way to represent high dimensional values

- Input data
- Model parameters
- Transformations
- Output

Tensors are a generalization of multi-dimensional arrays

Not to be confused with tensors in physics and engineering

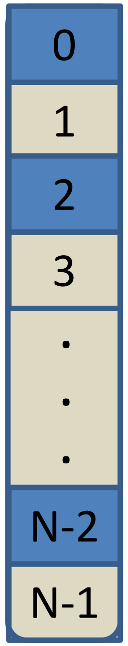
Tensors for Deep Learning

Zero dimensional tensor = scalar

0

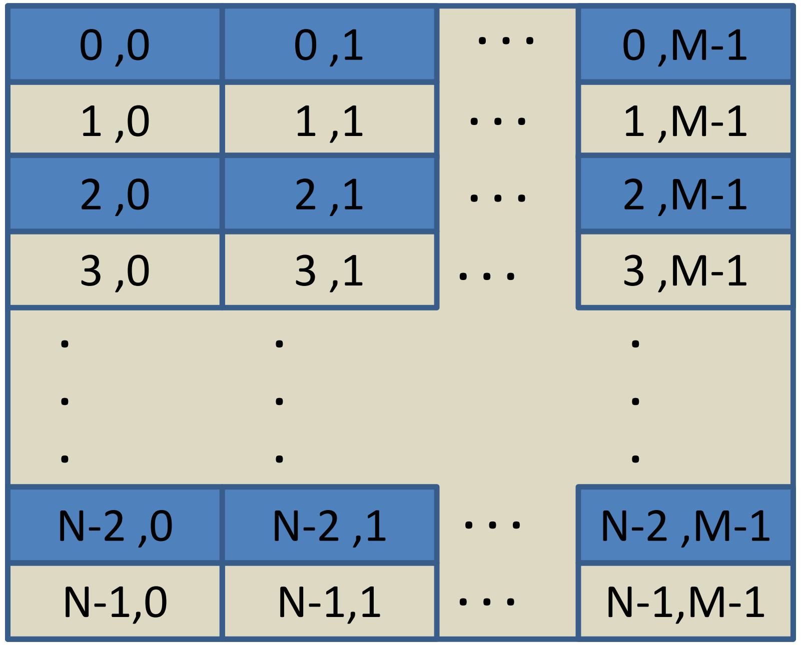
Tensors for Deep Learning

One dimensional tensor = vector



Tensors for Deep Learning

Two dimensional tensor = matrix



0 ,0	0 ,1	· · ·	0 ,M-1
1 ,0	1 ,1	· · ·	1 ,M-1
2 ,0	2 ,1	· · ·	2 ,M-1
3 ,0	3 ,1	· · ·	3 ,M-1
⋮	⋮		⋮
N-2 ,0	N-2 ,1	· · ·	N-2 ,M-1
N-1,0	N-1,1	· · ·	N-1,M-1

Working with Image Data

Deep learning has revolutionized image analysis and understanding

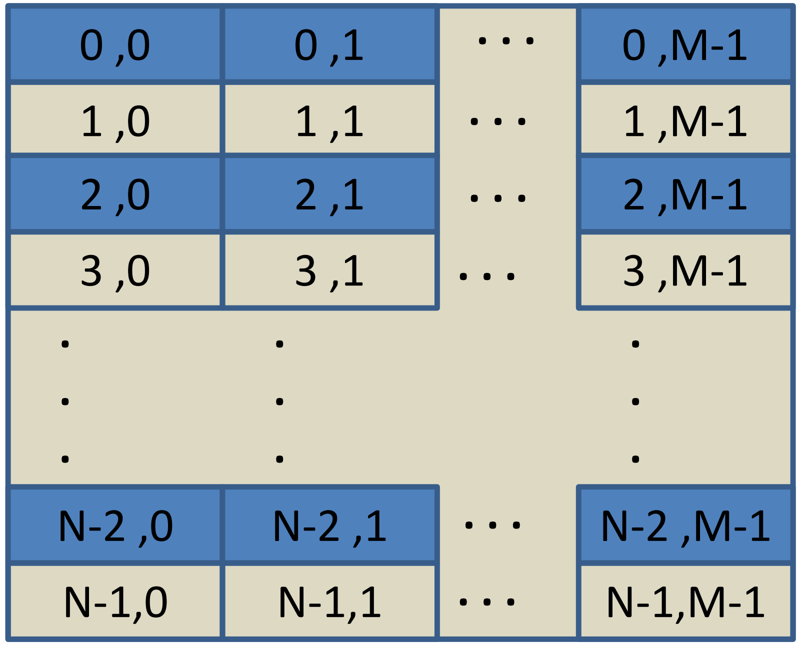
Common types of image data:

- Grey scale
- Color images
- Color video

Image data is represented by tensors

Working with Image Data

A grey scale image is represented as a two dimensional tensor of pixel values

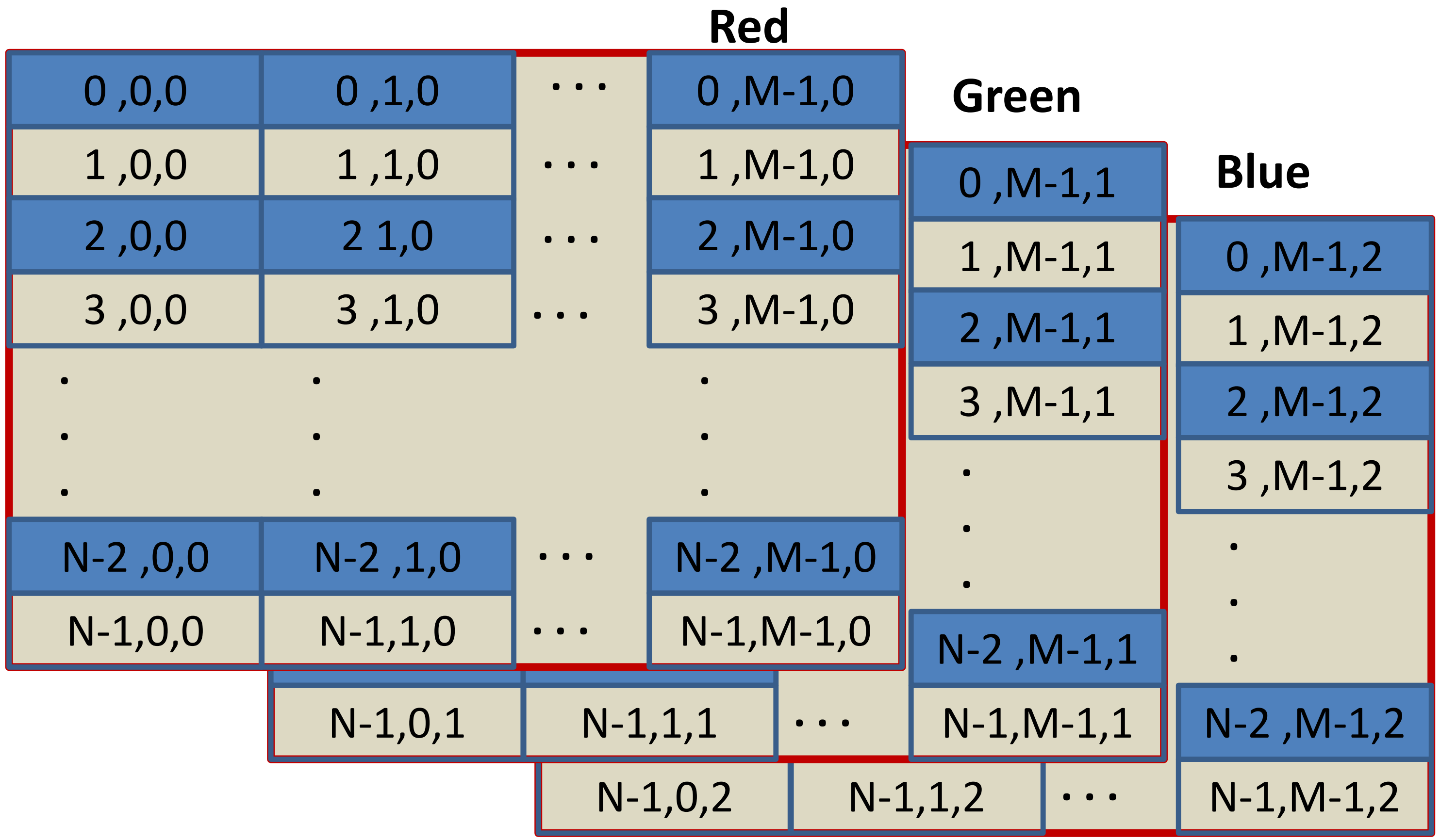


0 ,0	0 ,1	⋯	0 ,M-1
1 ,0	1 ,1	⋯	1 ,M-1
2 ,0	2 ,1	⋯	2 ,M-1
3 ,0	3 ,1	⋯	3 ,M-1
⋮	⋮		⋮
N-2 ,0	N-2 ,1	⋯	N-2 ,M-1
N-1,0	N-1,1	⋯	N-1,M-1

Pixel values have range {0-255}; white-black

Working with Image Data

A color image is a 3-dimensional tensor representing the three color channels



Red

0 ,0,0	0 ,1,0	···	0 ,M-1,0
1 ,0,0	1 ,1,0	···	1 ,M-1,0
2 ,0,0	2 1,0	···	2 ,M-1,0
3 ,0,0	3 ,1,0	···	3 ,M-1,0
⋮	⋮		⋮
N-2 ,0,0	N-2 ,1,0	···	N-2 ,M-1,0
N-1,0,0	N-1,1,0	···	N-1,M-1,0

Green

- 0 ,M-1,1
- 1 ,M-1,1
- 2 ,M-1,1
- 3 ,M-1,1
- ⋮
- N-2 ,M-1,1
- N-1,0,1 | N-1,1,1 | ··· | N-1,M-1,1

Blue

- 0 ,M-1,2
- 1 ,M-1,2
- 2 ,M-1,2
- 3 ,M-1,2
- ⋮
- N-2 ,M-1,2
- N-1,0,2 | N-1,1,2 | ··· | N-1,M-1,2

Intensity of color in each channel is on a scale of {0-255}

Working with Image Data

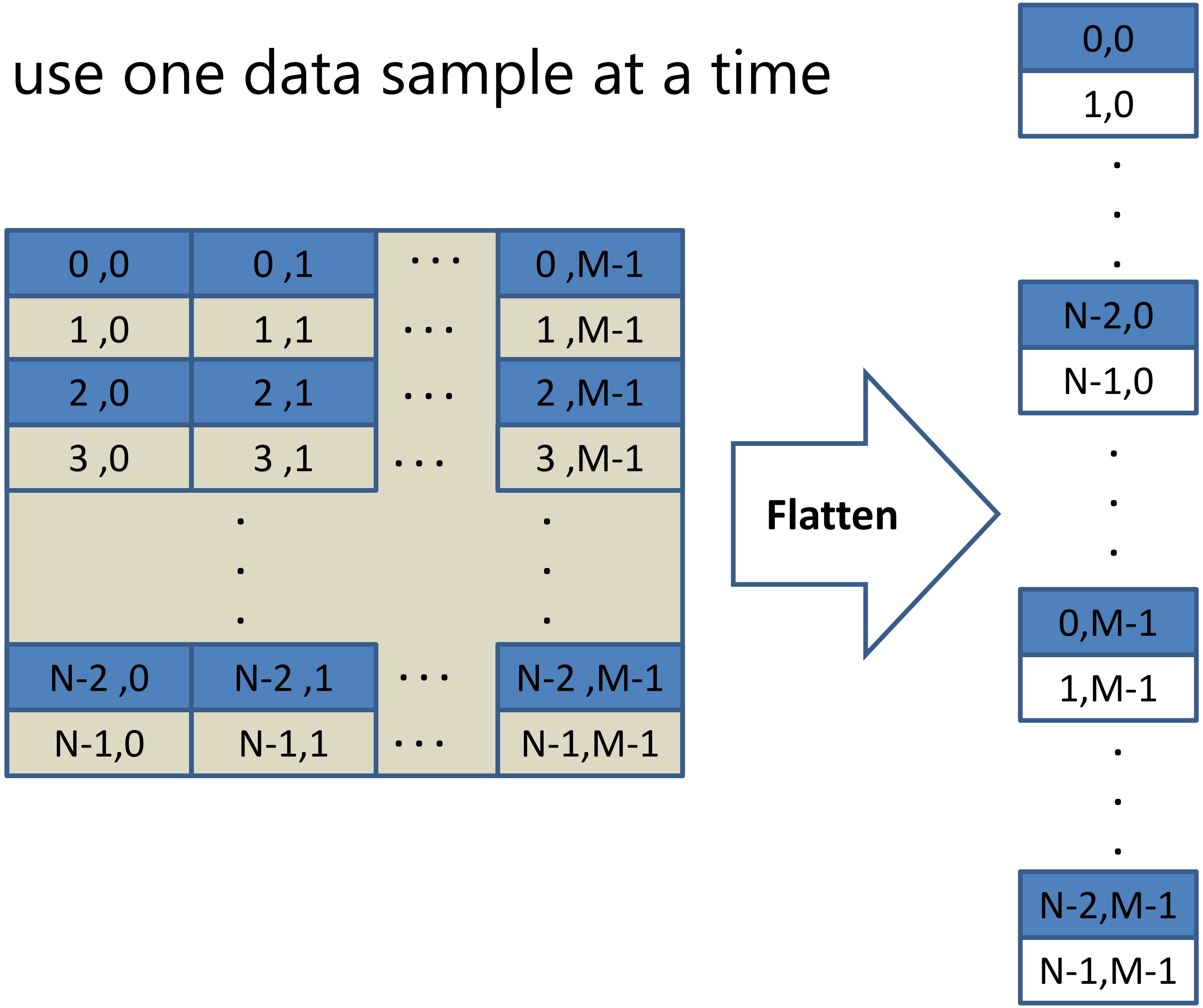
Video is represented as a 4-dimensional tensor

- 2 dimensions for each color channel
- 3rd dimension is color channel
- 4th dimension is the time sequence of images

Working with Image Data

Machine learning algorithms use one data sample at a time

For image data, we must **flatten** the image:

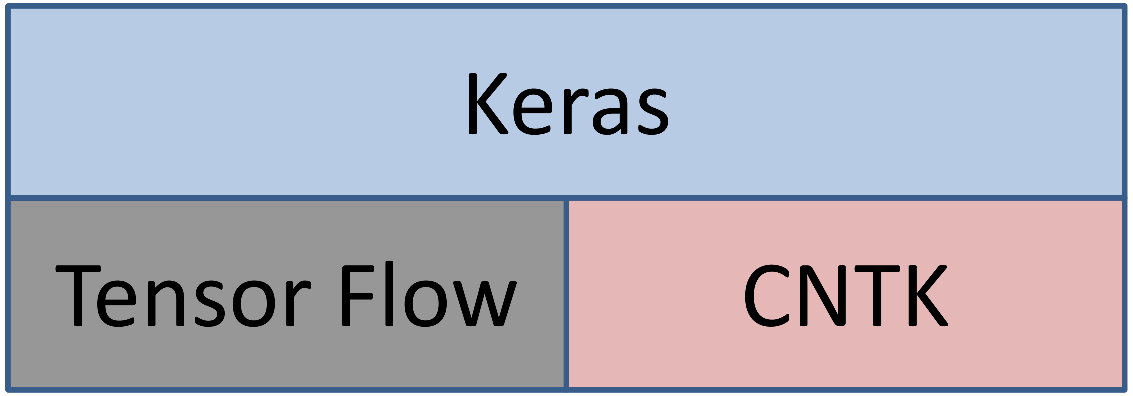


0 ,0	0 ,1	· · ·	0 ,M-1
1 ,0	1 ,1	· · ·	1 ,M-1
2 ,0	2 ,1	· · ·	2 ,M-1
3 ,0	3 ,1	· · ·	3 ,M-1
⋮			⋮
N-2 ,0	N-2 ,1	· · ·	N-2 ,M-1
N-1,0	N-1,1	· · ·	N-1,M-1

0,0
1,0
⋮
N-2,0
N-1,0
⋮
0,M-1
1,M-1
⋮
N-2,M-1
N-1,M-1

Introduction to Keras

- Keras abstracts common deep learning model definitions and operations
- Use on top of deep learning framework:



Keras

Tensor Flow

CNTK

- Extensive documentation at www.keras.io

Microsoft

The information herein is for informational purposes only and represents the current view of Microsoft Corporation as of the date of this presentation. Because Microsoft must respond to changing market conditions, it should not be interpreted to be a commitment on the part of Microsoft, and Microsoft cannot guarantee the accuracy of any information provided after the date of this presentation. MICROSOFT MAKES NO WARRANTIES, EXPRESS, IMPLIED OR STATUTORY, AS TO THE INFORMATION IN THIS PRESENTATION.