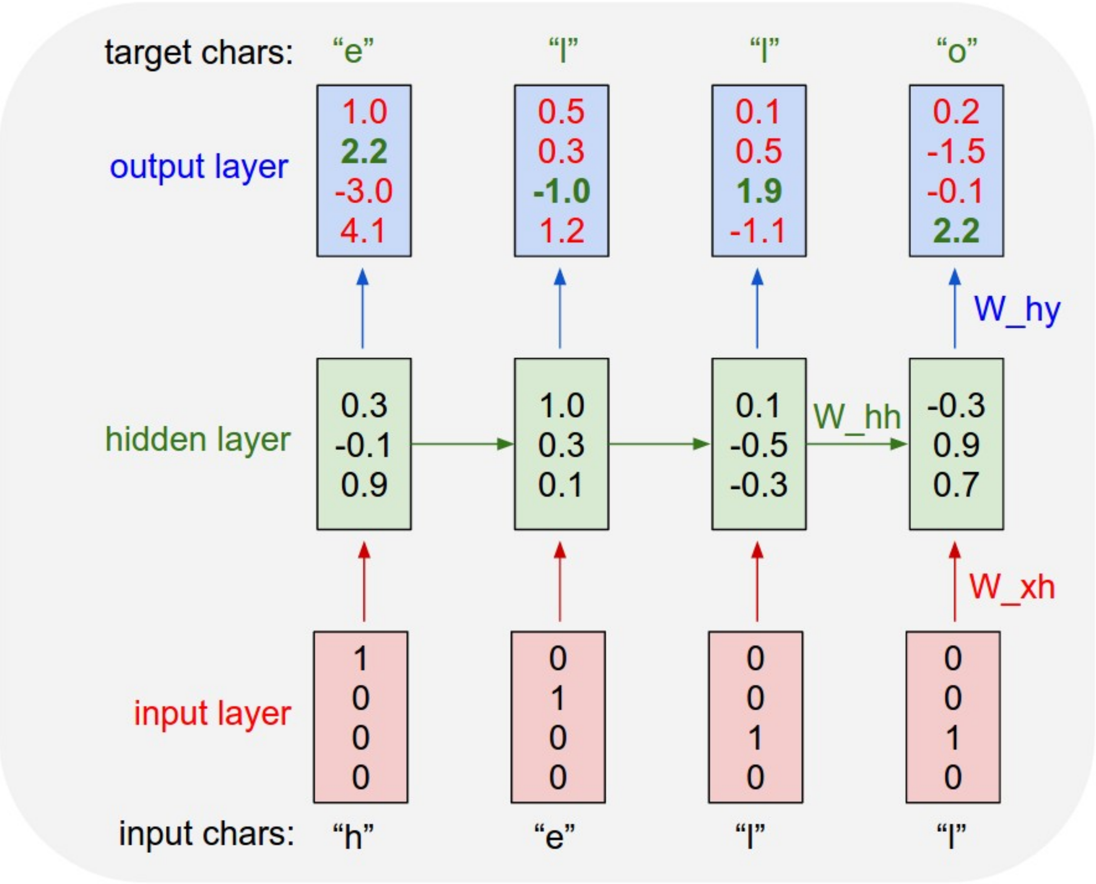
NLP with Deep Learning

- Let’s explore how to work with text data in conjunction with deep learning!
- This is a natural extension of the time series and recurrent neural network topics we just discussed.

- We will create a neural network that will generate new text based on a corpus of text data.
- Check out “The Unreasonable Effectiveness of RNNs” by Andrej Karpathy
- So how will this work?

- Given an input string sequence, predict the sequence shifted forward 1 character.
    - [“h” , ”e” , ”l” , ”l”]
    - [“e” , ”l” , ”l” , ”o”]

- The character based RNN will actually learn the structure of the text.
- In our example we will use the works of William Shakespeare.
- We will see the network clearly learn play writing structure and spacing, **just from a character level!**



target chars:	“e”	“l”	“l”	“o”
output layer	1.0 2.2 -3.0 4.1	0.5 0.3 -1.0 1.2	0.1 0.5 1.9 -1.1	0.2 -1.5 -0.1 2.2
hidden layer	0.3 -0.1 0.9	1.0 0.3 0.1	0.1 -0.5 -0.3	-0.3 0.9 0.7
input layer	1 0 0 0	0 1 0 0	0 0 1 0	0 0 1 0
input chars:	“h”	“e”	“l”	“l”

- Step 1: Read in Text Data
  - We can use basic built in python commands to read in a corpus of text as string data.
  - Note, you should have a large data set for this, at least 1 million characters for realistic results.

- Step 2: Text Processing and Vectorization
    - The neural network can’t take in raw strings, so we will encode them each to an integer.
        - A : 1
        - B : 2
        - C : 3
        - ? : 55

- Step 3: Creating Batches
    - We’ll use Tensorflow’s dataset object to easily create batches of text sequences.
        - [“h”, “e” , “l” , “l” , “o” , “ “, “m”]
        - [,“e” , “l” , “l” , “o” , “ “, “m”, “y”]

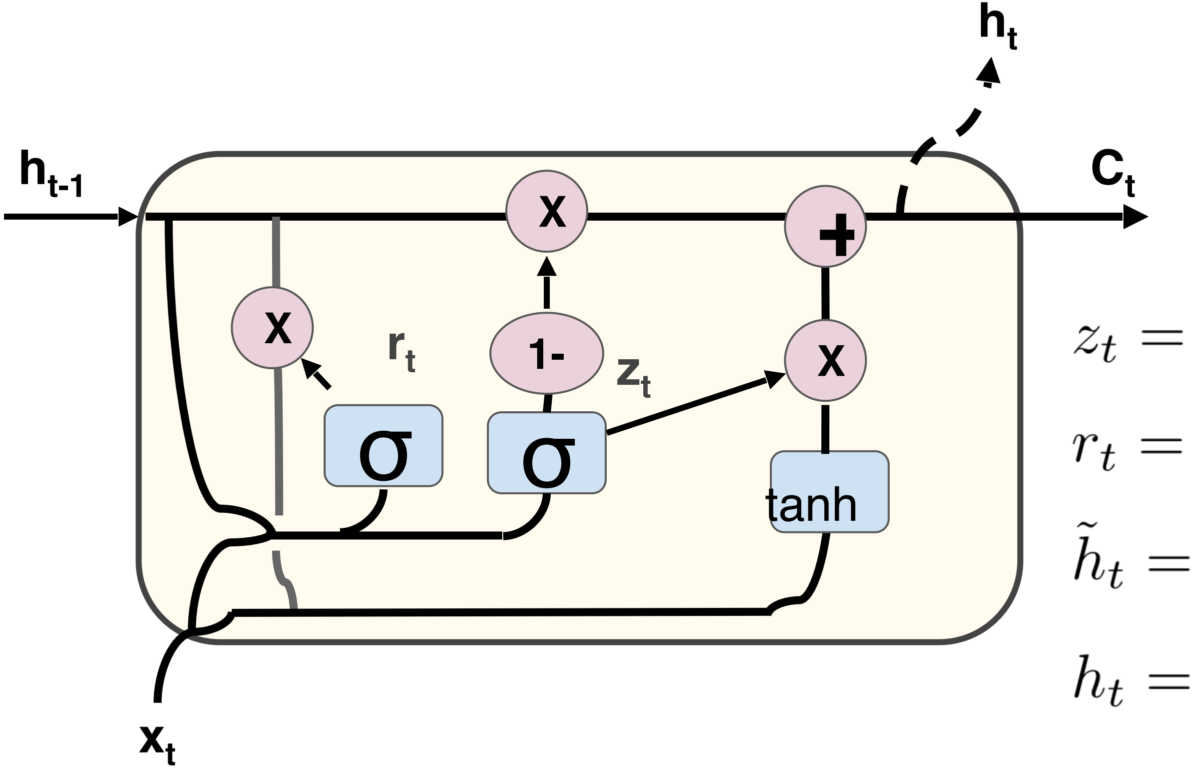
- Step 3: Creating Batches
  - We’ll want to use sequence lengths that are long enough to capture structure and previous words.
  - But not so long that the sequence is just historical noise.

- Step 4: Creating the Model
  - We’ll use 3 layers
    - Embedding
    - GRU
    - Dense

- Step 4: Creating the Model
  - Embedding Layer turns positive integers (indexes) into dense vectors of fixed size. eg. [[4], [20]] -> [[0.25, 0.1,0.3], [0.6, -0.2,0.9]]
  - Its up to the user to choose the number of embedding dimensions.

- Step 4: Creating the Model
  - GRU
    - Gated Recurrent Unit is a special type of recurrent neuron unit.
    - The GRU is like a long short-term memory (LSTM) with forget gate but has fewer parameters than LSTM, as it lacks an output gate.

Gated Recurrent Unit (GRU)



$$z_t = \sigma\left(W_z \cdot [h_{t-1}, x_t]\right)$$

$$r_t = \sigma\left(W_r \cdot [h_{t-1}, x_t]\right)$$

$$\tilde{h}_t = \tanh\left(W \cdot [r_t * h_{t-1}, x_t]\right)$$

$$h_t = (1 - z_t) * h_{t-1} + z_t * \tilde{h}_t$$

PIERIAN DATA

- Step 4: Creating the Model
  - Dense Layer
    - One neuron per character.
    - Character labels will be one hot encoded so the final dense layer produces a probability per character.

- Step 4: Creating the Model
  - Dense Layer
    - Probability per character means we can play around with “temperature”:
      - Choosing less probable characters more/less often

- Step 5: Training the Model
    - We’ll set up our batches and make sure to one-hot encode our character labels.

- Step 6: Generating new text
    - We’ll save our models weights and show you how to reload a model’s weights with a different batch size in order to pass in single examples.

Let’s get started!

Text Generation With Python and Keras

Part One

Deep Learning

- Part 1: The Data
  - Import main libraries
  - Importing Text
  - Understanding Characters

Text Generation With Python and Keras

Part Two

- Part 2: Text Processing
  - Vectorize the text
  - Create encoding dictionary

Text Generation With Python and Keras

Part Three

- Step 3: Creating Batches
  - Understand text sequences
  - Use Tensorflow datasets to generate batches
  - Shuffle batches

Text Generation
With Python and Keras

Part Four

- Step 4: Creating the Model
  - Set up loss function
  - Create Model
    - Embedding
    - GRU
    - Dense

Text Generation With Python and Keras

Part Five

- Step 5: Training the Model
  - We’ll quickly show an example of how to train the model.
  - We’ll also show you how to load our provided saved model file.

Text Generation With Python and Keras

Part Six

- Step 6: Generating Text
  - We’ll load our model
  - Adjust batch size to 1
  - Run a loop that generates new text