Training NNs and RNNs

Alexander Fraser

(Slides originally from Denis Peskov)

SS 2023

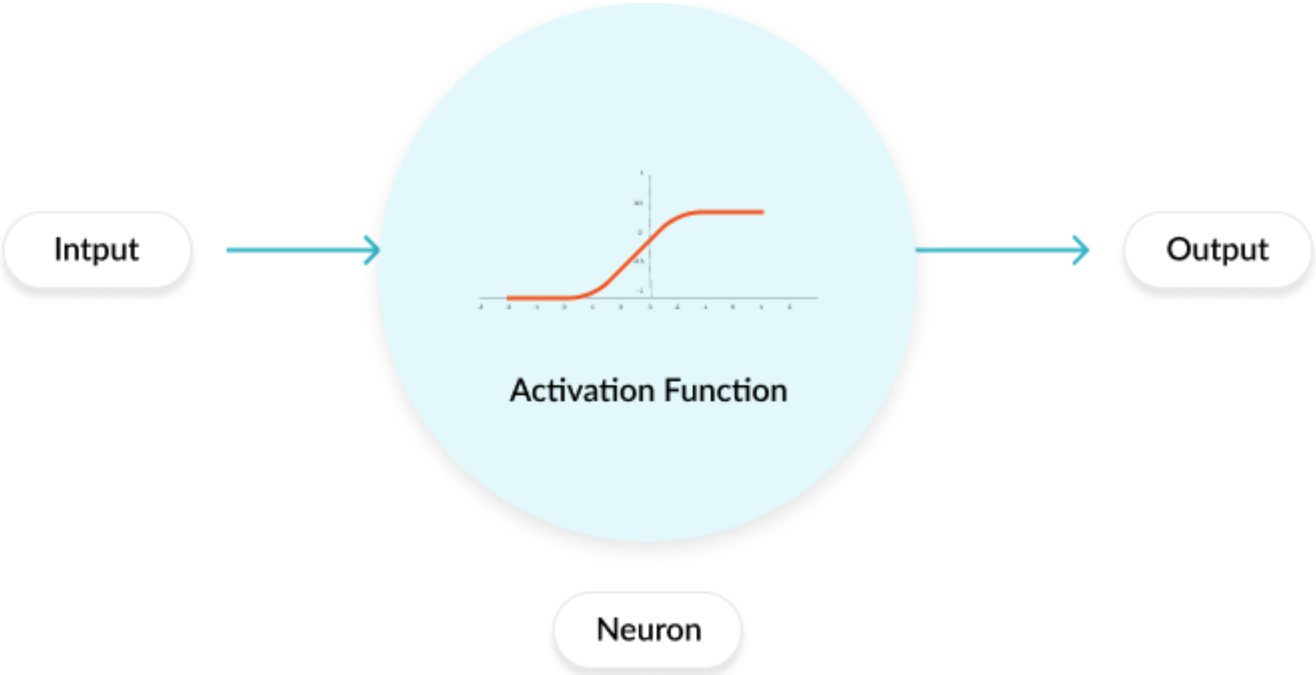
Topics

1. Activation functions
2. Training of neural networks
3. Recurrent networks
4. Subword tokenization (BPE)

Training a Neural Network

- Neurons
- Activation Functions
- Loss Functions
- Backpropogation
- Pragmatics

The Base Level: Neuron



Source: Missinglink.ai: 7 types of neural network activation functions (2019)

Activation Function

Obvious:

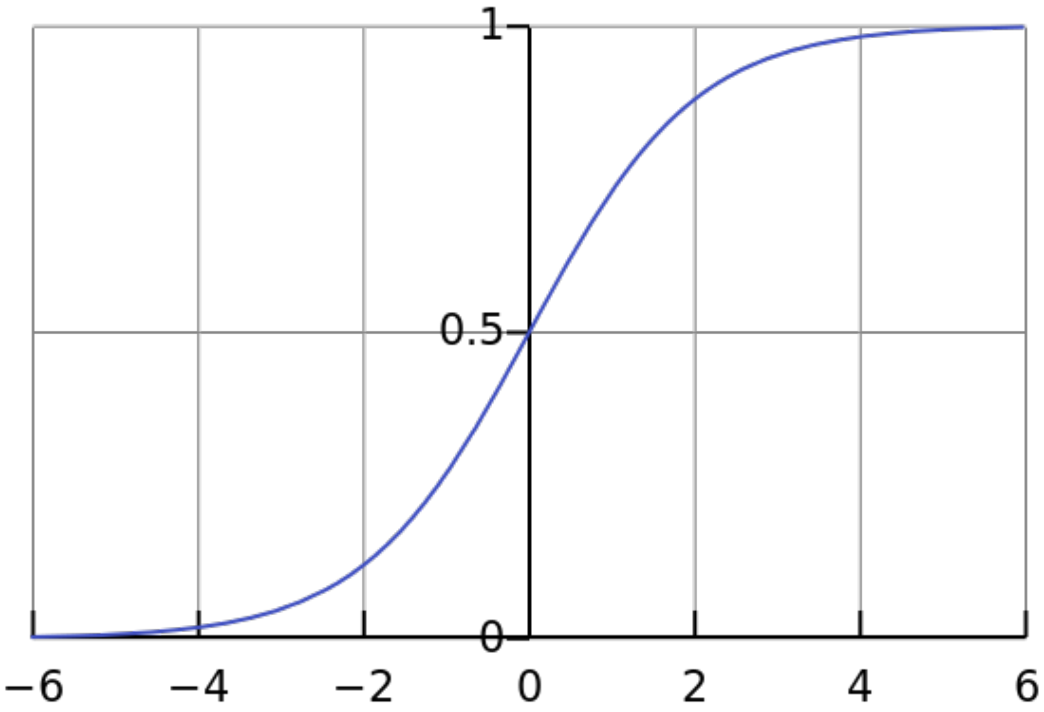
Linear: $A = cx$

But:

1. Derivative is constant
2. Stacking layers no longer works

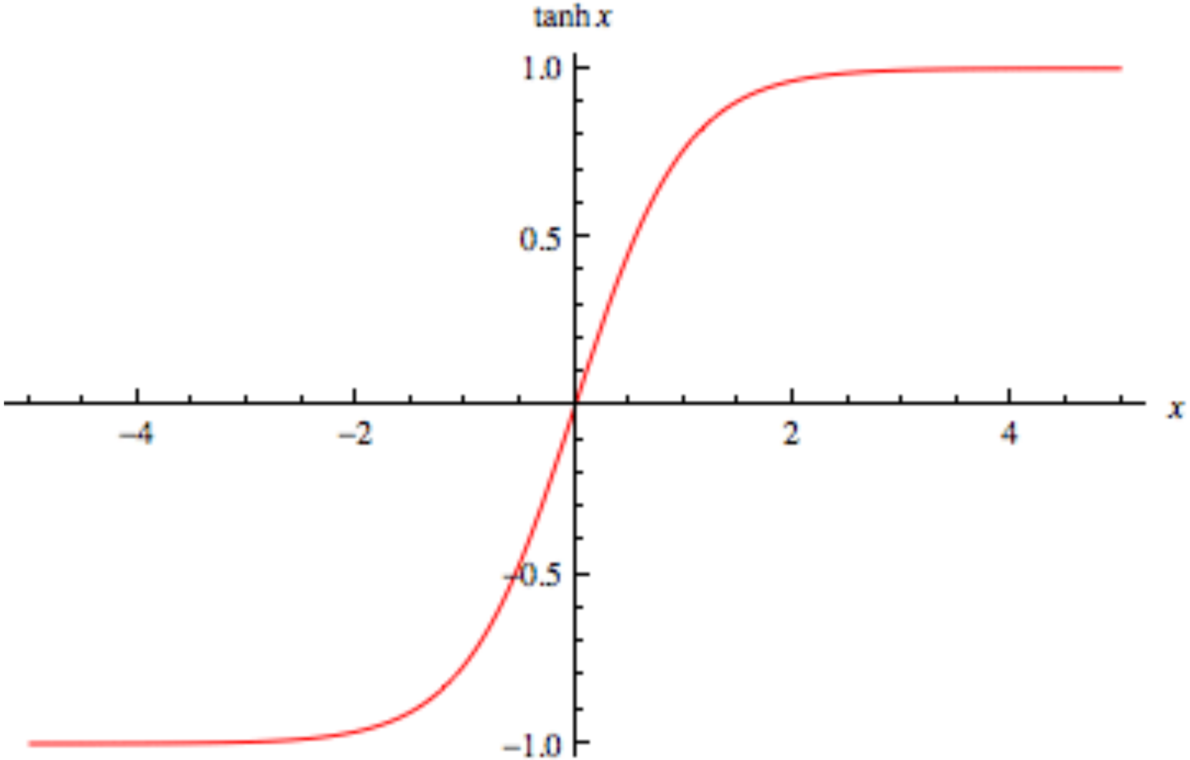
So we need **nonlinear** activation functions

sigmoid() (and softmax)



Sigmoid: https://en.wikipedia.org/wiki/Sigmoid_function

tanh()   (zero-centered)



Tanh: https://mathworld.wolfram.com/

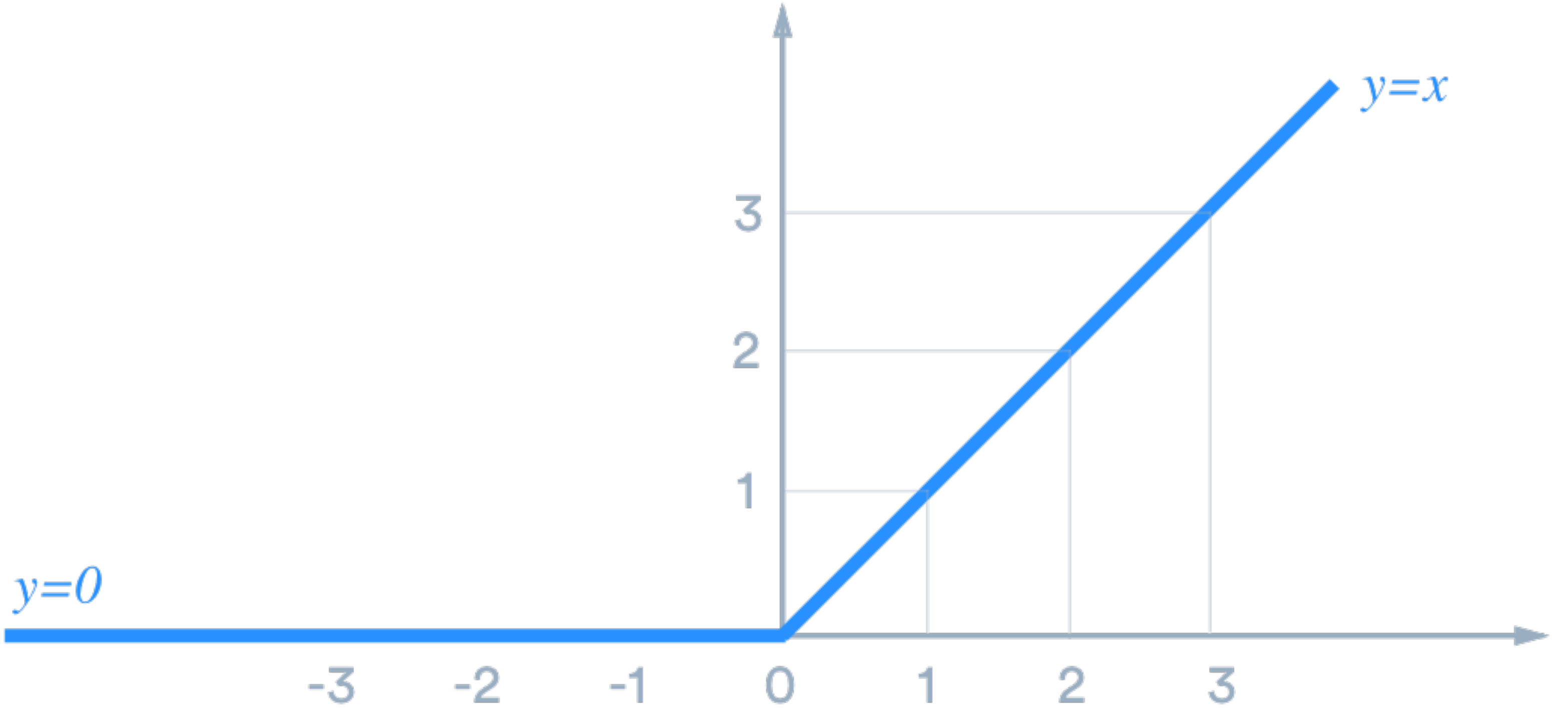
Vanishing Gradient

Gradient Descent is used for training Neural Networks

$$o(x) = f_n(f_{n-1}(\ldots.f_1(x))$$

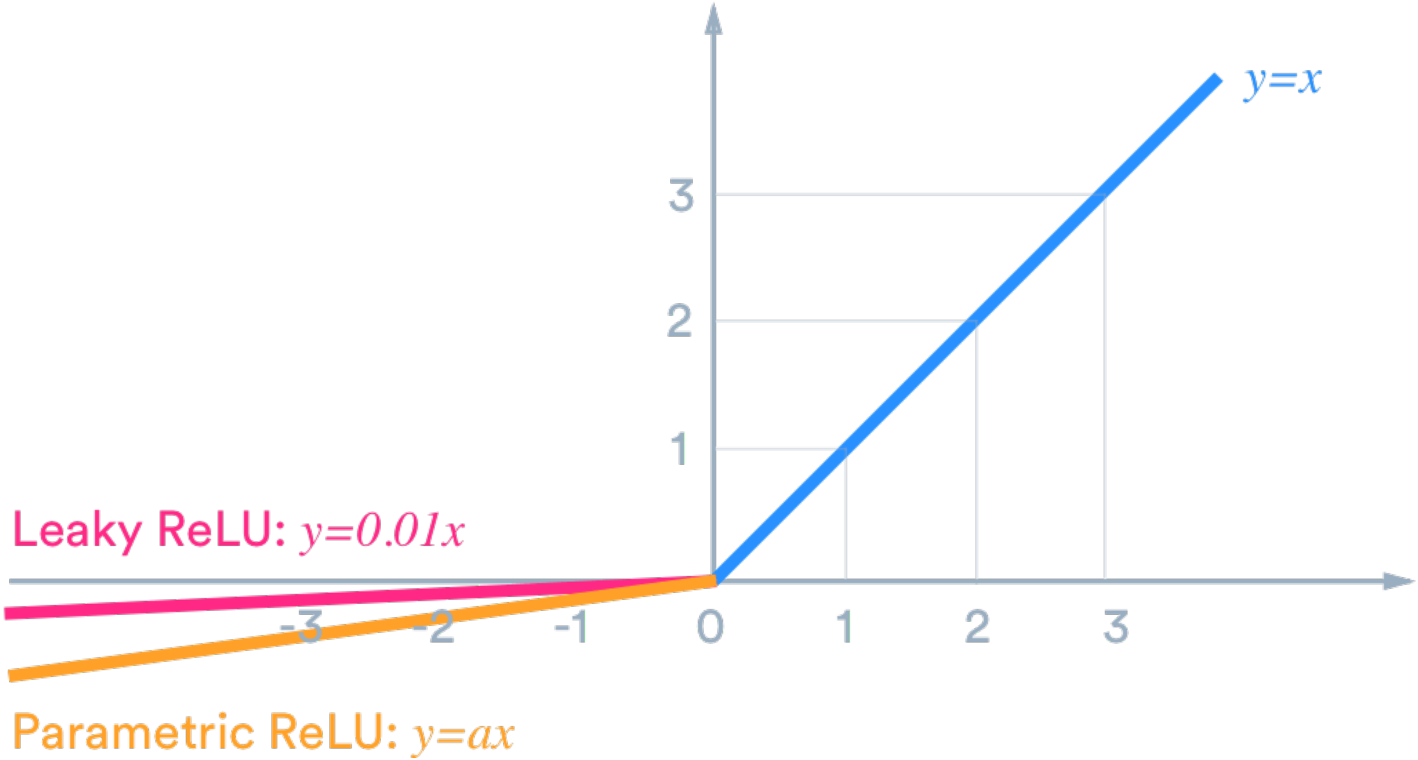
Multiplying values < 1 across multiple layers causes **VANISHING GRADIENT**

Avoid Vanishing Gradient with ReLU



Activation Functions: Sigmoid, ReLU, Leaky ReLU and Softmax basics for Neural Networks and Deep Learning. Himanshu S. Jan 19, 2019.

Avoiding “dying neurons”: Leaky ReLU



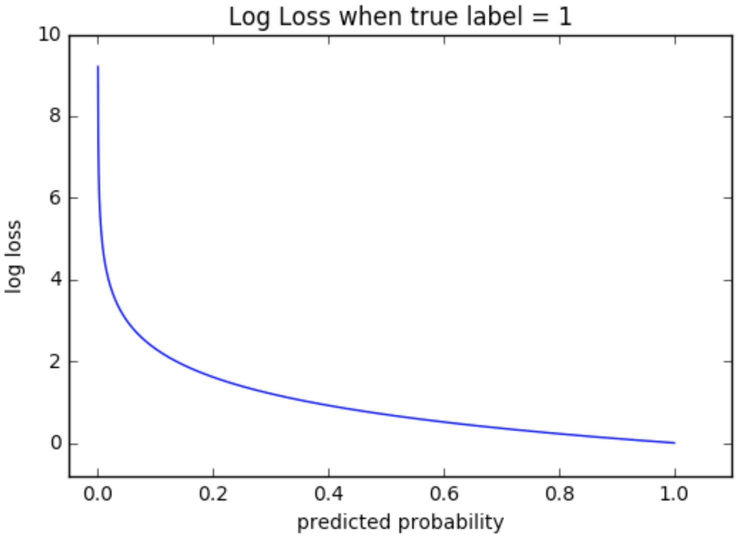
Activation Functions: Sigmoid, ReLU, Leaky ReLU and Softmax basics for Neural Networks and Deep Learning. Himanshu S. Jan 19, 2019.

Loss Function

Mean Squared Error:

$$MSE = \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{n}$$

Cross Entropy Loss:

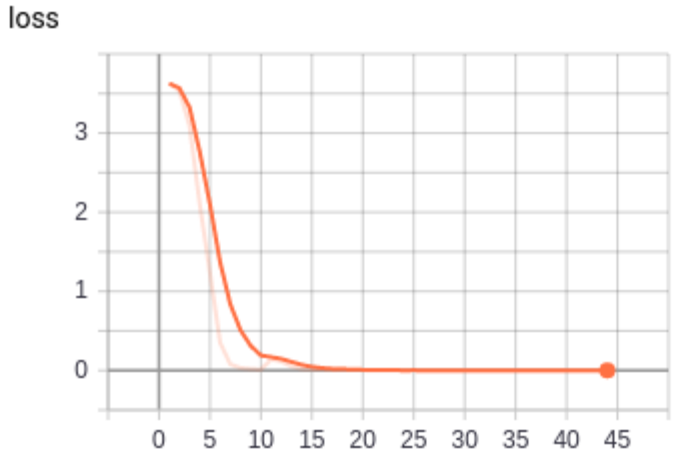


https://ml-cheatsheet.readthedocs.io/en/latest/loss_functions.html (ML Cheatsheet 2020)

How to use loss?

Train your network while loss is decreasing.

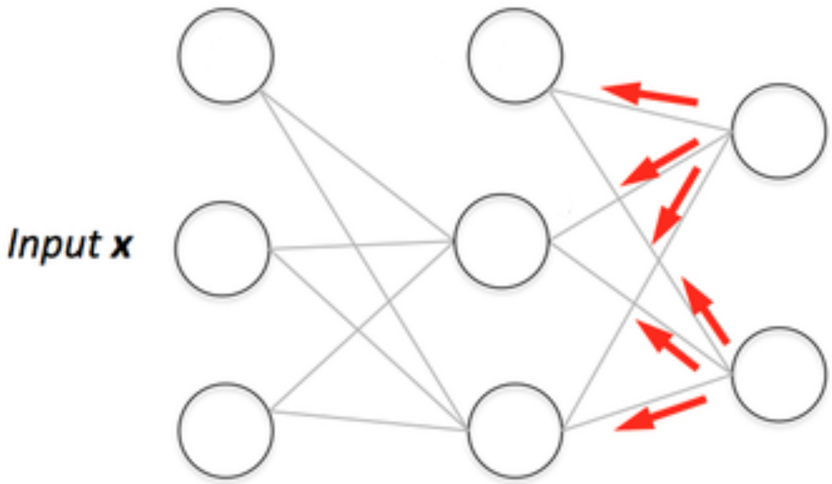
Perfect probability = loss of 0.



Loss isn't very intuitive.   Better to report train accuracy or similar.

Backpropagation

1. Forward pass
2. Error Calculation
3. Backwards Pass



How are Neural Networks Trained in Practice (NLP)?

Data

- Large!

Tools

- PyTorch
- AllenNLP

Hardware

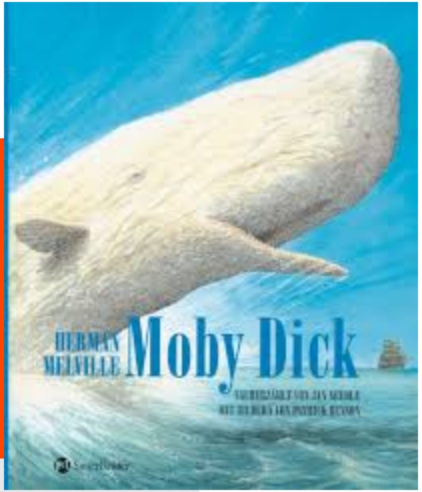
- GPUs (and even TPUs)

Neural Network Data for NLP

1. Neural networks often have thousands of parameters.
2. Law of large numbers avoids data inconsistency.
3. Beware of biases.
4. Denis Peskov: on **small** datasets in my own work, *I've* personally had close results with logistic regression models.
5. But in machine translation, logistic regression is not usually an option

Where do you get NLP data?

- Internet
- Books
- Crowd-Sourcing
- Artificial modifications
- Specialized Communities



But how do you guarantee quality of NLP data?

- Interannotator Agreement
- Think about biases:
  - Label: you only learn what's in the training data
  - Language: skewed towards popular languages
  - Text: text data requires less space than audio/video data and can be older
- Visualization

But what about language?

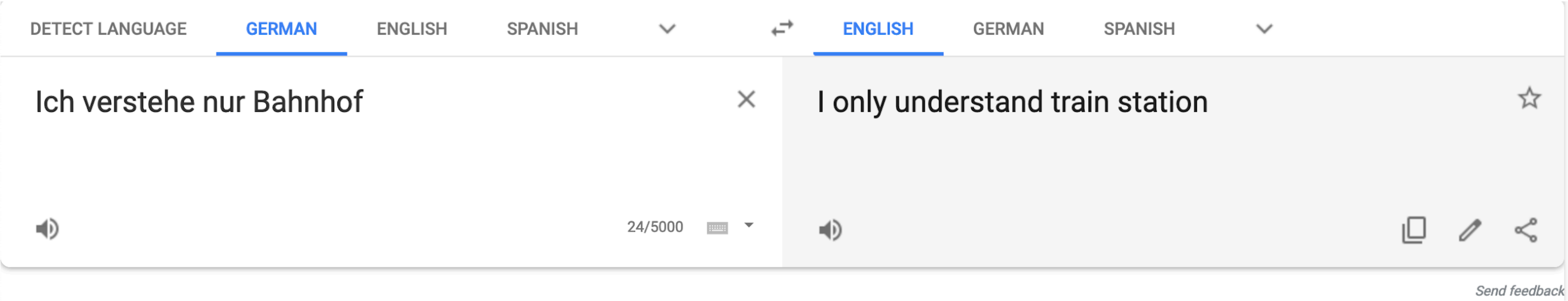
Neural Networks were a big leap in accuracy for **VISION.** Pixels were high in dimensionality, and difficult to interpret.

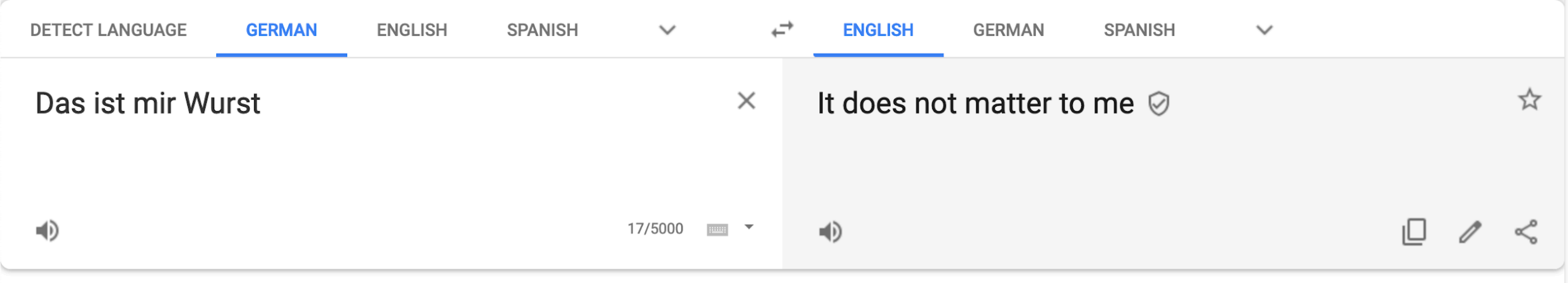
Human interpretable

Levels:

1. Character
2. Word
3. Phrase/Sentence
4. Document

Context Matters

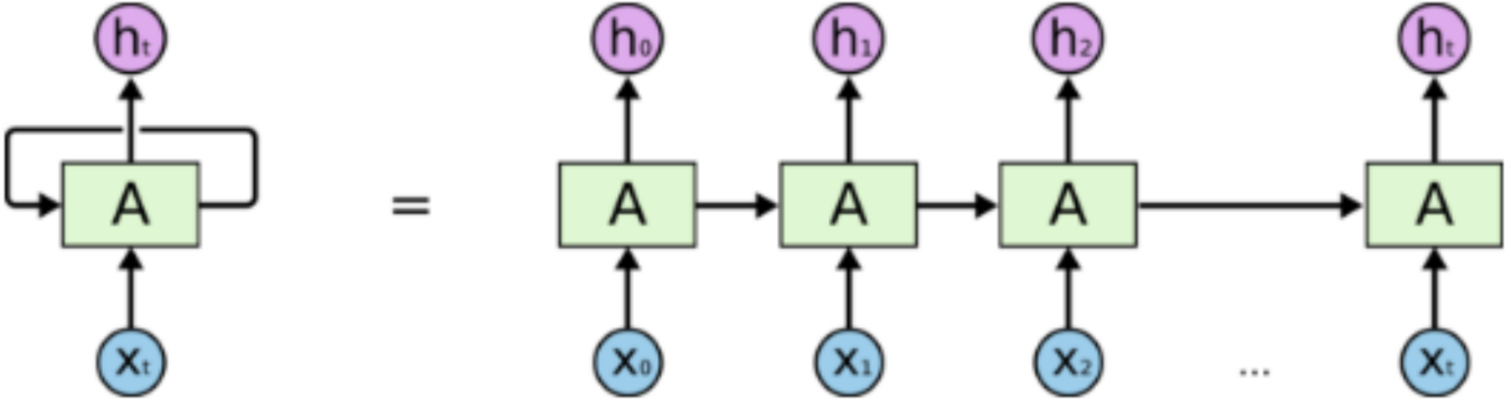




Pragmatics

1. Train/ Development/ Test splits
2. Batching
3. Random seed
4. Reasonable Significant Digits
5. Drop out data during training
6. Initialization
7. Human baselines & common sense
8. Monitor training loss

Recurrent Neural Networks (RNN)



An unrolled recurrent neural network.

Understanding RNN and LSTM. Aditi Mittal, Medium, 2020.
https://aditi-mittal.medium.com/understanding-rnn-and-lstm-f7cdf6dfc14e

Limitations

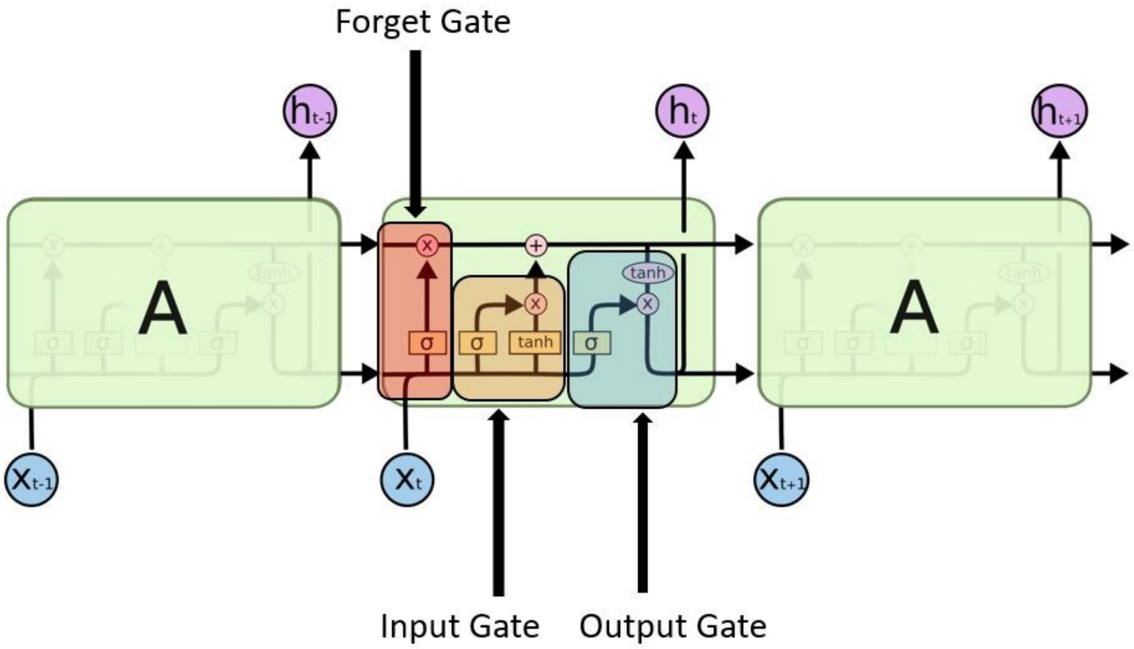
$$h_t = f(h_{t-1}, x_t)$$

$$h_t = \tanh\left(W_{hh}h_{t-1} + W_{xh}x_t\right)$$

W is *weight*, **h** is the *single hidden vector,* $\mathbf{W_{hh}}$ is *the weight at previous hidden state,* $\mathbf{W_{xh}}$ is the *weight at current input state,* **tanh** is the *Activation function,* which implements a non-linearity that squashes the activations to the range [-1, 1]

- Vanishing Gradient Losing Long Term Information
- Computation

Add Gates: Long Short-Term Memory (LSTM)



Input: Is this relevant?

Cell State: What to add?

Output: Where to send next?

Understanding RNN and LSTM. Aditi Mittal, Medium, 2020.
https://aditi-mittal.medium.com/understanding-rnn-and-lstm-f7cdf6dfc14e

Customers Review 2,491

Thanos

September 2018

Verified Purchase

Amazing! This box of cereal gave me a perfectly balanced breakfast, as all things should be. I only ate half of it but will definitely be buying again!

A Box of Cereal

$3.99

Customers Review 2,491

Thanos

September 2018

Verified Purchase

Amazing! This box of cereal gave me a **perfectly balanced breakfast**, as all things should be. I only ate half of it but **will definitely be buying again!**

A Box of Cereal

$3.99

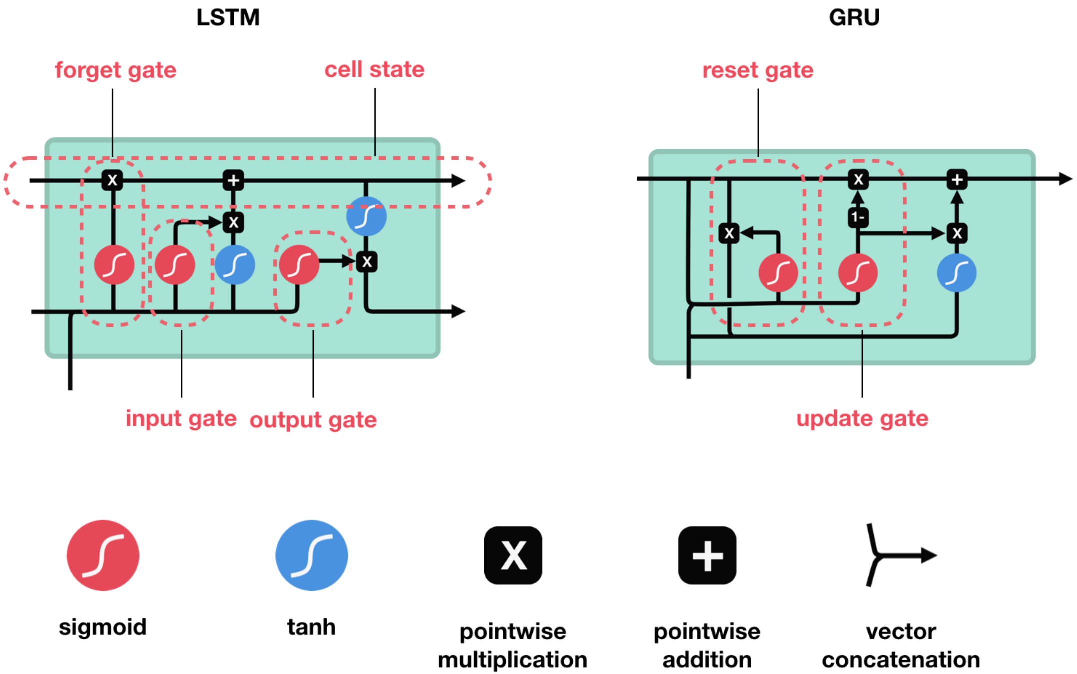
What’s the connection to LANGUAGE?

Language is both:

- Dependent on overall context
- Often short-term sequential

Other Variation: GRU

Optimizing the memory by forgetting leads to a Gated Recurrent Unit (GRU)



Michael Phi (2018). An Illustrated Guide to LSTMs and GRUs, a step-by-step explanation.

Role	Turn	Annotations
A	Hey there! Good morning. You're connected to LMT Airways. How may I help you?	DA = { elicitgoal }
C	Hi, I wonder if you can confirm my seat assignment on my flight tomorrow?	IC = { SeatAssignment }
A	Sure! I'd be glad to help you with that. May I know your last name please?	DA = { elicitslot }
C	My last name is Turker.	IC = { contentonly }, SL = {Name : Turker }
A	Alright Turker! Could you please share the booking confirmation number?	DA = { elicitslot }
C	I believe it's AMZ685.	IC = { contentonly }, SL = { Confirmation Number : AMZ685 }
...	...	...

Table 1: A segment of a dialogue from the airline domain annotated at the turn level. This data is annotated with agent dialogue acts (DA), customer intent classes (IC), and slot labels (SL). Roles C and A stand for "Customer" and "Agent", respectively.

Peskov et al (2019). Multi-Domain Goal-Oriented Dialogues (MultiDoGO). EMNLP

Subword Tokenization

- MT systems can have large vocabularies, but this requires large amounts of RAM.
- There is also the problem of OOV (out of vocabulary) words
- Both of these were solved for Neural Machine Translation by Sennrich, Haddow, Birch in 2016
- The critical idea is to split words into known subwords as a preprocessing step used at both training and test time

BPE, Wordpiece, Unigram

- Sennrich et al's BPE algorithm is simple: begin by splitting words into characters (i.e., "dog" to the three tokens "d" "o" "g"), then merge frequent pairs of tokens iteratively
- This is typically applied to the concatenation of the source and target corpus in MT, to ensure that identical names are split identically in the two languages
- Usually the total vocabulary is limited to something like 16000 or 30000
- Wordpiece is a Google in-house variant of BPE, used in, e.g., BERT, while "Unigram" (Kudo, 2018) is a newer variant that uses unigram probabilities of the resulting tokens

BPE vs Unigram vs ...

Bostrom und Durrett (Findings EMNLP 2020) show that Unigram may be better:

Original:	furiously	**Original:**	tricycles	**Original:**	nanotechnology
BPE:	`_fur iously`	**BPE:**	`_t ric y cles`	**BPE:**	`_n an ote chn ology`
Uni. LM:	`_fur ious ly`	**Uni. LM:**	`_tri cycle s`	**Uni. LM:**	`_nano technology`

Original:	`Completely preposterous suggestions`
BPE:	`_Comple t ely _prep ost erous _suggest ions`
Unigram LM:	`_Complete ly _pre post er ous _suggestion s`

Original:	`corrupted`	**Original:**	`1848 and 1852,`
BPE:	`_cor rupted`	**BPE:**	`_184 8 _and _185 2,`
Unigram LM:	`_corrupt ed`	**Unigram LM:**	`_1848 _and _1852 ,`

In my group we have a number of papers showing the effectiveness of stemming, morphological analysis and even morphological generation in NMT (and also SMT). See work by Marion Weller-Di Marco, for instance.

Questions?