INF Informatik
Hochschule Reutlingen

Intelligente Informationssysteme
Block 1 – Language Models

Dominik Neumann

Language Models - Agenda

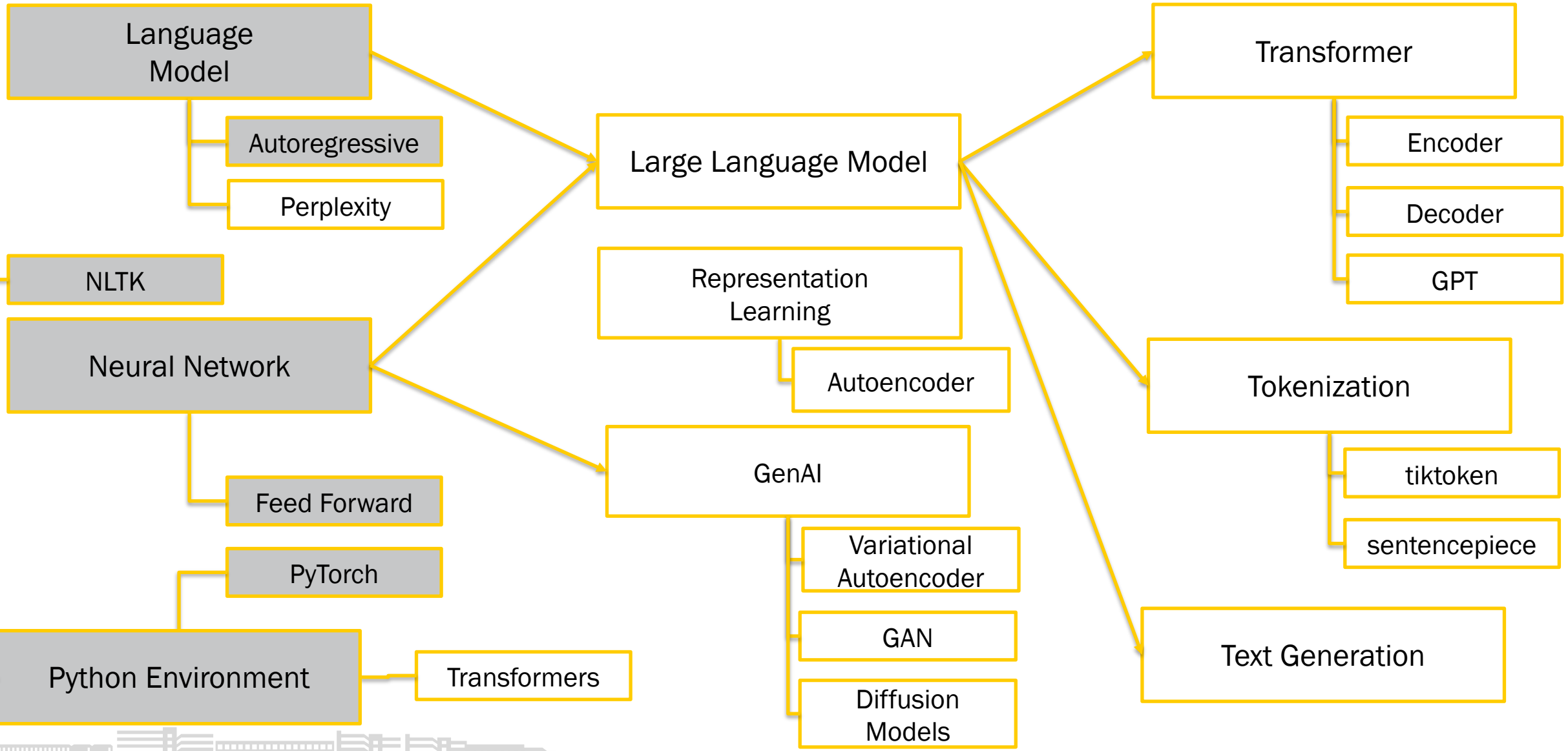
Hochschule Reutlingen

Language Model

2. What is a **model**?
1. Why is **language** so important in Artificial Intelligence?

3. What is a Language Model?
4. How to build a Language Model?

Overview


Language Model
Autoregressive
Perplexity
NLTK
Neural Network
Feed Forward
PyTorch
Python Environment
Transformers
Large Language Model
Representation Learning
Autoencoder
GenAI
Variational Autoencoder
GAN
Diffusion Models
Transformer
Encoder
Decoder
GPT
Tokenization
tiktoken
sentencepiece
Text Generation

01
Language is the jewel in the crown of cognition

Steven Pinker

INF Informatik
Hochschule Reutlingen

Language

Hochschule Reutlingen

1. Language is a system of communication that uses symbols in a regular way to create meaning.
2. Language gives us the ability to communicate to others by talking, reading, and writing.
3. Human language is powerful and has been transformative to our species because it gives groups of people a way to network human brains together.

INF
Informatik
Hochschule Reutlingen

02
Concept of a Model in Machine Learning

What does it mean to “model something”?

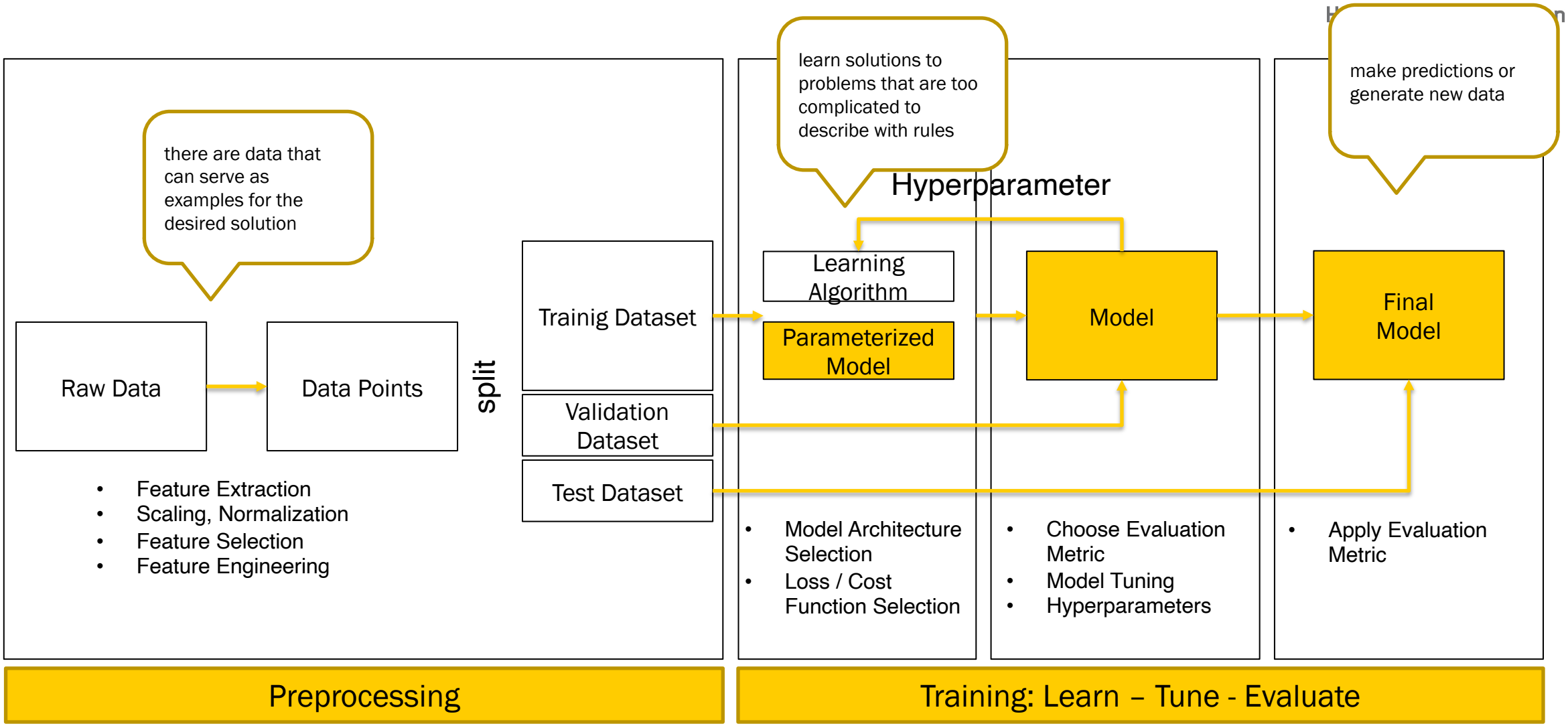
Imagine that we have a model of a physical world.

What do you expect it to be able to do?

- A good model would predict what happens next, given some description of “context”.
- A good model would simulate the behavior of the real word.
- It would “understand” which events are in better agreement with the world. Which of them are more likely.

Concept of a Model in Machine Learning

- learned in some machine learning process



Concept of a Model in Machine Learning

Informatik
Hochschule Reutlingen

As a mathematical parameterized model that explain some data

Machine learning (ML) develops, researches and uses statistical algorithms, also known as learning algorithms.

Learning algorithms can learn solutions to problems that are too complicated to describe with rules, but there are data that can serve as examples for the desired solution.

A learning algorithm maps given example data onto a mathematical parameterized model. The learning algorithm adapts the parameters of the model so that it can generalize from the example data to new cases. This process is called training. After training, the solution found is stored in the model. It is not explicitly programmed.

The trained model can make predictions or generate new data.

There are always three ingredients for Machine Learning:

1. some data
2. a parameterized model that can explain the data, given an appropriate choice of parameters.
3. at least an evaluation procedure, the loss function, that allows us to adjust the parameters of our model.

Concept of a Model in Machine Learning

- as a „black box“ function with input and output performing some task

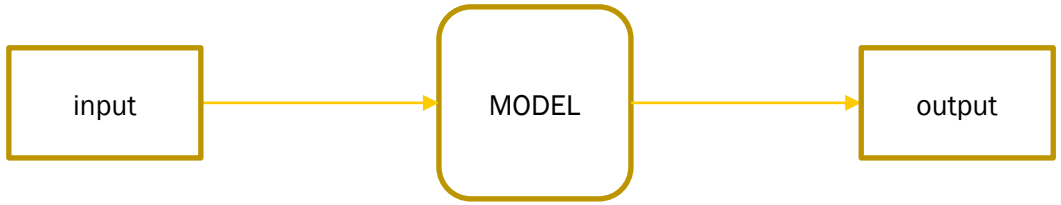
Informatik
Hochschule Reutlingen

The universal Function Approximation Theorem tells us, that every function can be approximated by a deep neural network:

A ML task is a type of prediction or inference being made by the model. It is based on the problem or question that is being asked, and the available data.

For example, the classification task assigns data to categories, and the clustering task groups data according to similarity.

A Model is like a function with input and output performing some task.



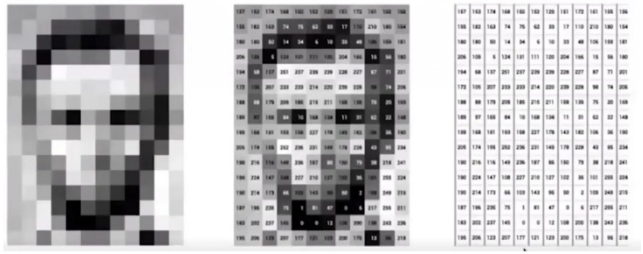
Concept of a Model in Machine Learning

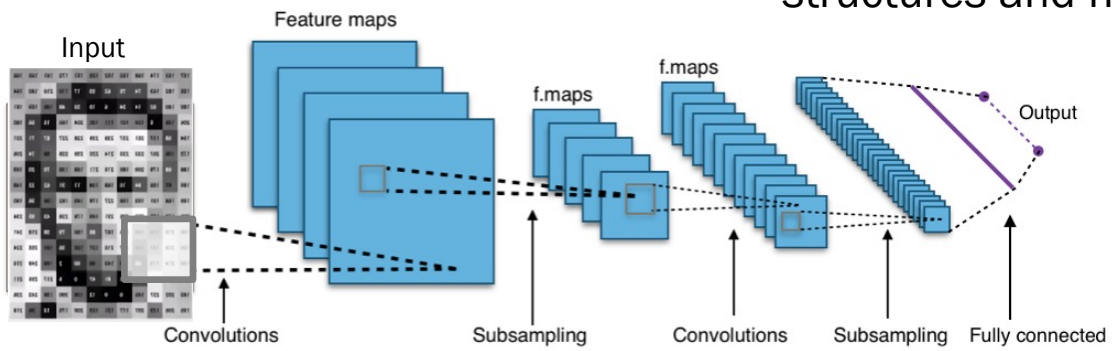
Models and their Deep Learning Network Architectures

Informatik
Hochschule Reutlingen

While designing a model, we make assumptions about our data depending on the geometry of the data: called Inductive Bias

1. Images are represented as Matrix or Tensor





Inductive Bias

- Nearby pixels have similar colors
- Use convolutional filters to extract structures and hidden features

Source: Wikipedia - https://commons.wikimedia.org/w/index.php?curid=45679374

Concept of a Model in Machine Learning

INF Informatik
Hochschule Reutlingen

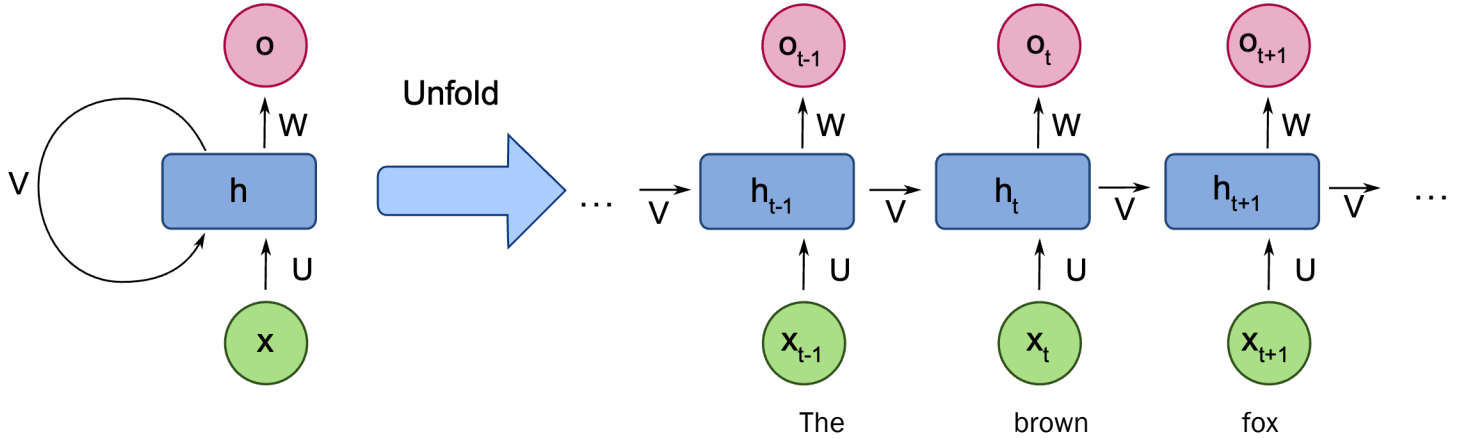
While designing a model, we make assumptions about our data depending on the geometry of the data: called Inductive Bias

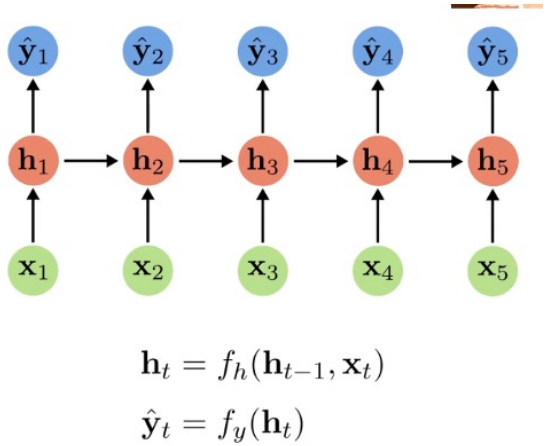
2. Language is represented as a Sequence:

["The", "brown", "fox", ...]

Inductive Bias

- Nearby words belong to the same semantic meaning
- sequence-2-sequence models like recurrent neural networks could be used to model context with hidden states





$$\mathbf{h}_t = f_h(\mathbf{h}_{t-1}, \mathbf{x}_t)$$
$$\hat{\mathbf{y}}_t = f_y(\mathbf{h}_t)$$

Source: Wikipedia - https://commons.wikimedia.org/w/index.php?curid=60109157

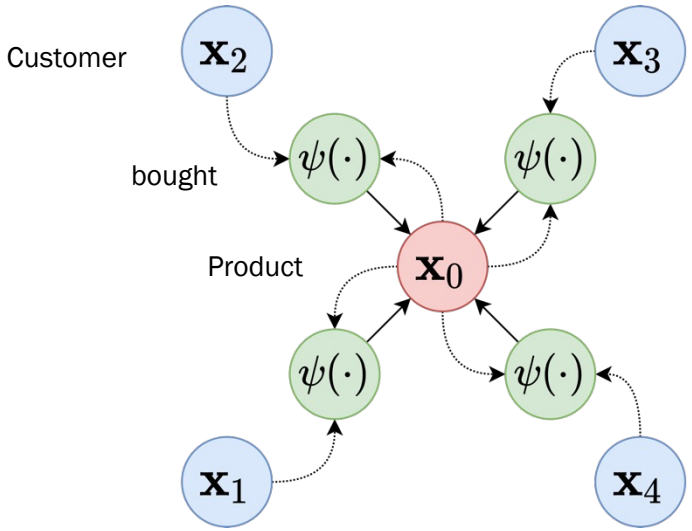
Concept of a Model in Machine Learning

Models and their Deep Learning Network Architectures

Hochschule Reutlingen

While designing a model, we make assumptions about our data depending on the geometry of the data: called Inductive Bias

3. Data like objects with relationships to each other can be represented as a graph



Inductive Bias

- Use the graph structure of the data to build a graph neural network.
- In a Message Passing GNN the network nodes update their representations by aggregating the messages received from their neighbors.
- Node representation could be used to downstream task, like classification, clustering or link prediction.

Source: Wikipedia - https://commons.wikimedia.org/w/index.php?curid=119852416

INF
Informatik
Hochschule Reutlingen

03
Language Models

Language Models

Hochschule Reutlingen

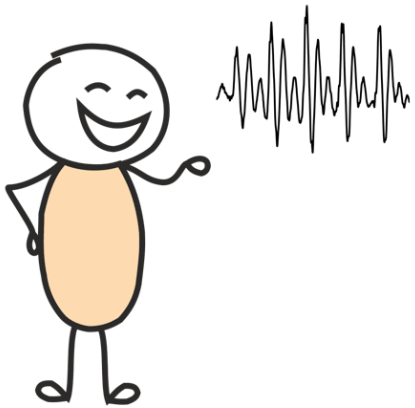
In language, an event is a linguistic unit: text, sentences, word, token, symbol

The goal of a language model is to estimate the probabilities of these events.

Definition: A Language model is a probability distribution over linguistic units.
It estimate the probability of different linguistic units: symbols, tokens, token sequences.

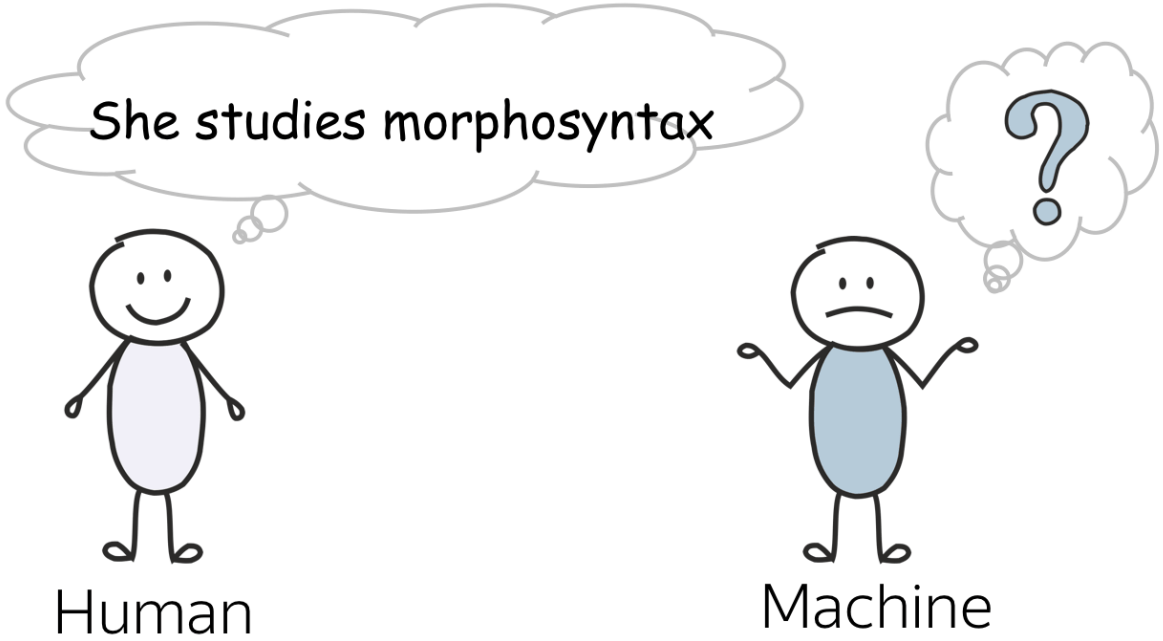
Hochschule Reutlingen

What is easy for humans, can be very hard for machines



Similarly sounding options

She studies morphosyntax
She studies more faux syntax
She studies morph or syntax
....



Source: the morphosyntax example is from the slides by Alex Lascarides and Sharon Goldwater, Foundations of Natural Language Processing course at the University of Edinburgh.

Language Models

Hochschule Reutlingen

But how a machine is supposed to understand?

- A machine needs a language model, which estimates the probabilities of sentences.

- If a language is good, it will assign a larger probability to a correct sentence.

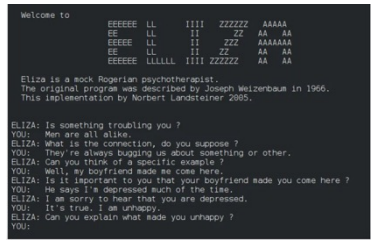
  P(„this is a correct sentence.“) > P(„htis is a coreckt sentence“.)

Language Models

Informatik
Hochschule Reutlingen

History of (Large) Language Models

ELIZA
(Joseph Weizenbaum)



N-gram:

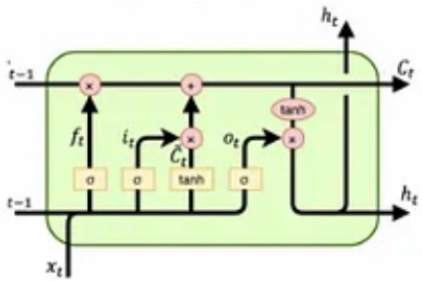
N-gram models emphasized the importance of context in language

P(word | given some history)

(Sufficient digitally available text)

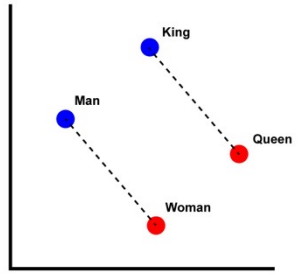
LSTM:

Long Short-Term Memory
Sepp Hochreiter
Jürgen Schmidhuber



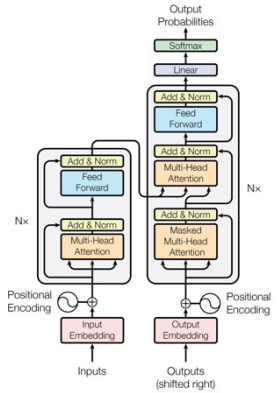
Word2Vec:

Word Embeddings
Thomas Mikolov
(Google)



Google's Neural Machine Translation System

Transformer
Attention is all you need
(Google)



1991 - The World Wide Web provides a Massive Source of Data

1966 | 1990 | 1995 | 2013 | 2016 | 2017

Rule-based models | Statistical language models | Neural Networks

04
How to build a Language Model

How to build a Language Model - General Framework

Informatik
Hochschule Reutlingen

A language model is a probability distribution over sequences of a vocabulary.

General Framework:

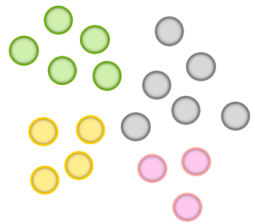
- Our goal is to estimate probabilities of text fragments.
- Without loss of generality (w.l.o.g.) we deal with sentences.
- We want these probabilities to reflect knowledge of a language.
- We want sentences that are “more likely” to appear in a language to have larger probability according to our language model.

Source: https://lena-voita.github.io/nlp_course/language_modeling.html

How to build a Language Model - General Framework

Could simple probability theory help?

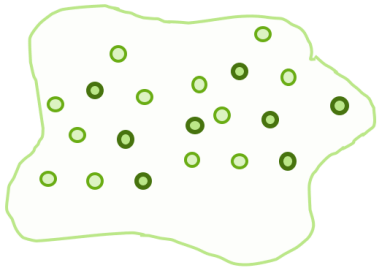
What is the probability to pick a green ball?



$$\frac{5}{5+6+4+3} = \frac{5}{18}$$

The probability to pick a ball of a certain color (let's say green) from this basket is the frequency with which green balls occur in the basket.

Can we do the same for sentences?



Text corpus

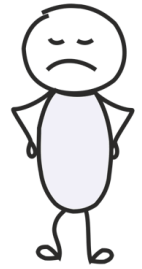
$$P(\text{the mut is tinming the tebn}) = \frac{0}{|\text{corpus}|} = 0$$

$$P(\text{mut the tinming tebn is the}) = \frac{0}{|\text{corpus}|} = 0$$

With this approach, sentences that never occurred in the corpus will receive zero probability

Since we can not possibly have a text corpus that contains all sentences in a natural language, a lot of sentences will not occur in the corpus.
While among these sentences some are clearly more likely than the others, all of them will receive zero probability.

But the first sentence is "more likely" than the second!
This method is not good!



Source: https://lena-voita.github.io/nlp_course/language_modeling.html

How to build a Language Model - General Framework

Hochschule Reutlingen

Solution: Decompose a sentence into smaller parts!

We can not reliably estimate sentence probabilities if we treat them as atomic units. Instead, let's decompose the probability of a sentence into probabilities of smaller parts.

For example, let's take the sentence **I saw a cat on a mat** and imagine that we read it word by word. At each step, we estimate the probability of all seen so far tokens.

$$P(\text{I}) =$$

$$\underbrace{P(\text{I})}_{\text{Probability of I}}$$

Source: https://lena-voita.github.io/nlp_course/language_modeling.html

How to build a Language Model - General Framework

Hochschule Reutlingen

Formally, let be *(x_1, ..., x_n)* tokens in a sentence, and *P (x_1, ..., x_n)* the probability to see all these tokens (in this order). Using the product rule of probability (aka the chain rule), we get:

$$P(x_1, x_2, \dots x_n) = P(x_1) * P(x_2|x_1) * P(x_3|\, x_1, x_2) * \cdots * P(x_n|x_1, x_2, \dots, x_{n-1})$$
$$= \prod_{t=1}^{n} P(x_t|x_{<t}).$$

- We decomposed the probability of a text into conditional probabilities of each token given the previous context.
- What we got is the **autoregressive** left-to-right language modeling framework.

Source: https://lena-voita.github.io/nlp_course/language_modeling.html

How to build a Language Model - General Framework

Need to define, how to compute $P(x_n|x_1, x_2, \ldots, x_{n-1})$.

- N-Gram
- Neural Models (neural networks)

Source: https://lena-voita.github.io/nlp_course/language_modeling.html

How to build a Language Model - General Framework

Hochschule Reutlingen

Generate text using a Language Model?

I ____

Once we have a language model, we can use it to generate text.

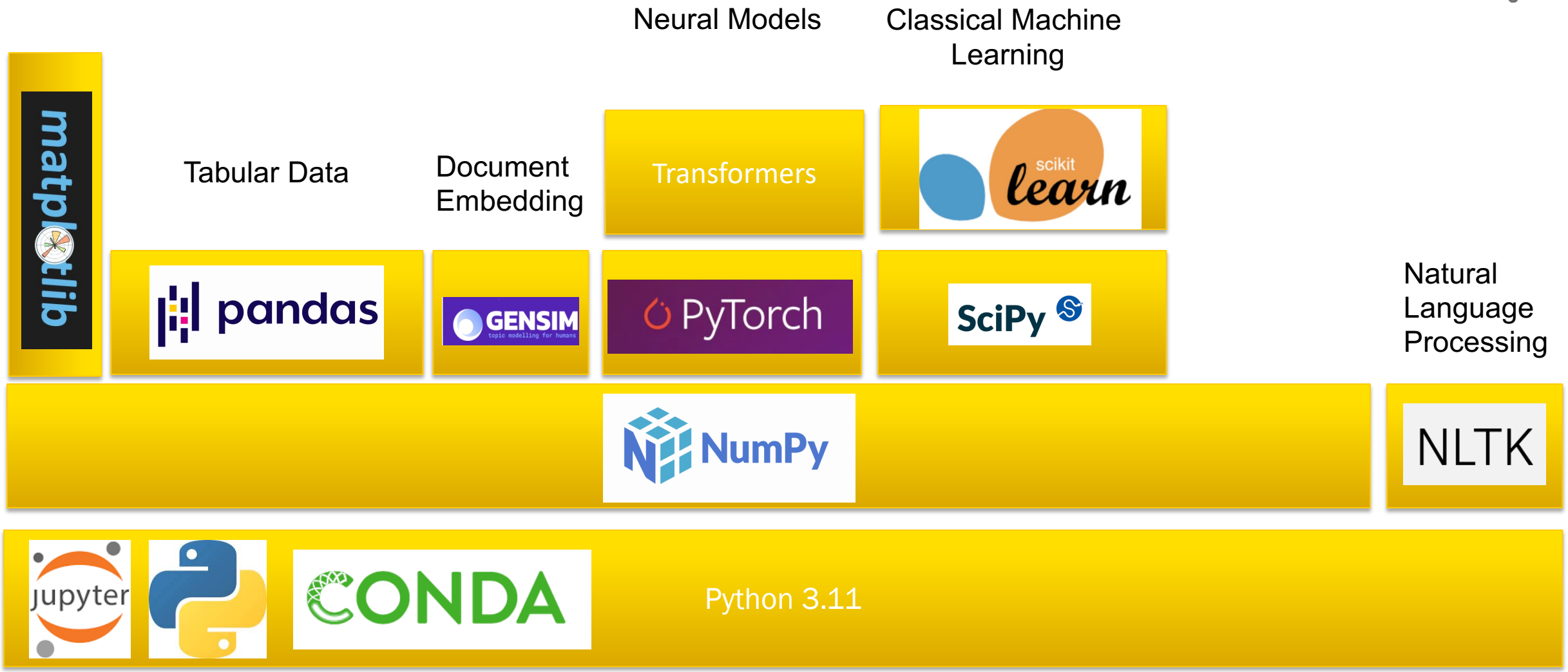
We do it one token at a time: predict the probability distribution of the next token given previous context, and sample from this distribution.

Source: https://lena-voita.github.io/nlp_course/language_modeling.html

05
Build a N-Gram Language Model with Python

INF Informatik
Hochschule Reutlingen

Install Python Environment



Install Python

INF Informatik
Hochschule Reutlingen

- Python: https://www.python.org
- Python on Mac with homebrew: https://docs.python-guide.org/starting/install3/osx/
- Python on Windows: https://docs.python-guide.org/starting/install3/win/

Python Environment Management

Hochschule Reutlingen

Create for each project its own Python environment to omit dependency conflicts:

$ **python –m venv** *envpath* creates a virtual environment (in *envpath* directory)

Beispiel:

```
(base) done@Dominiks-MBP ~ % python -m venv Documents/tmpenv/test1
(base) done@Dominiks-MBP ~ % cd Documents/tmpenv/test1
(base) done@Dominiks-MBP test1 % ls -l
total 8
drwxr-xr-x  12 done  staff  384  2 Okt 10:46 bin
drwxr-xr-x   2 done  staff   64  2 Okt 10:46 include
drwxr-xr-x   3 done  staff   96  2 Okt 10:46 lib
-rw-r--r--   1 done  staff   89  2 Okt 10:46 pyvenv.cfg
```

Python Environment Management

Hochschule Reutlingen

```
(base) done@Dominiks-MBP ~ % tree -dL 4 Documents/tmpenv/test1

Documents/tmpenv/test1
├── bin   ← Link to system python
├── include
└── lib
    └── python3.10
        └── site-packages
            ├── _distutils_hack
            ├── pip
            ├── pip-22.3.1.dist-info
            ├── pkg_resources
            ├── setuptools
            └── setuptools-65.5.0.dist-info
```

Python Environment Management

Informatik
Hochschule Reutlingen

Activate and deactivate the virtual environement

On Unix/MacOS:
$ **source** *envpath*/**bin/activate** activates the environment
$ **source** *envpath*/**bin/deactivate** deactivates the environment

On Windows:
C:\> *envpath*/**Scripts/activate.bat**

Activation does many things:

1. Add virtualenv‘s *bin* directory at the beginning of shell PATH
2. Defines a deactivate command to return the environment to ist former state
3. Modifies shell prompt: puts the environment name in front of it
4. Defines a VIRTUAL_ENV environment variable as the path to virtual environments root directory.

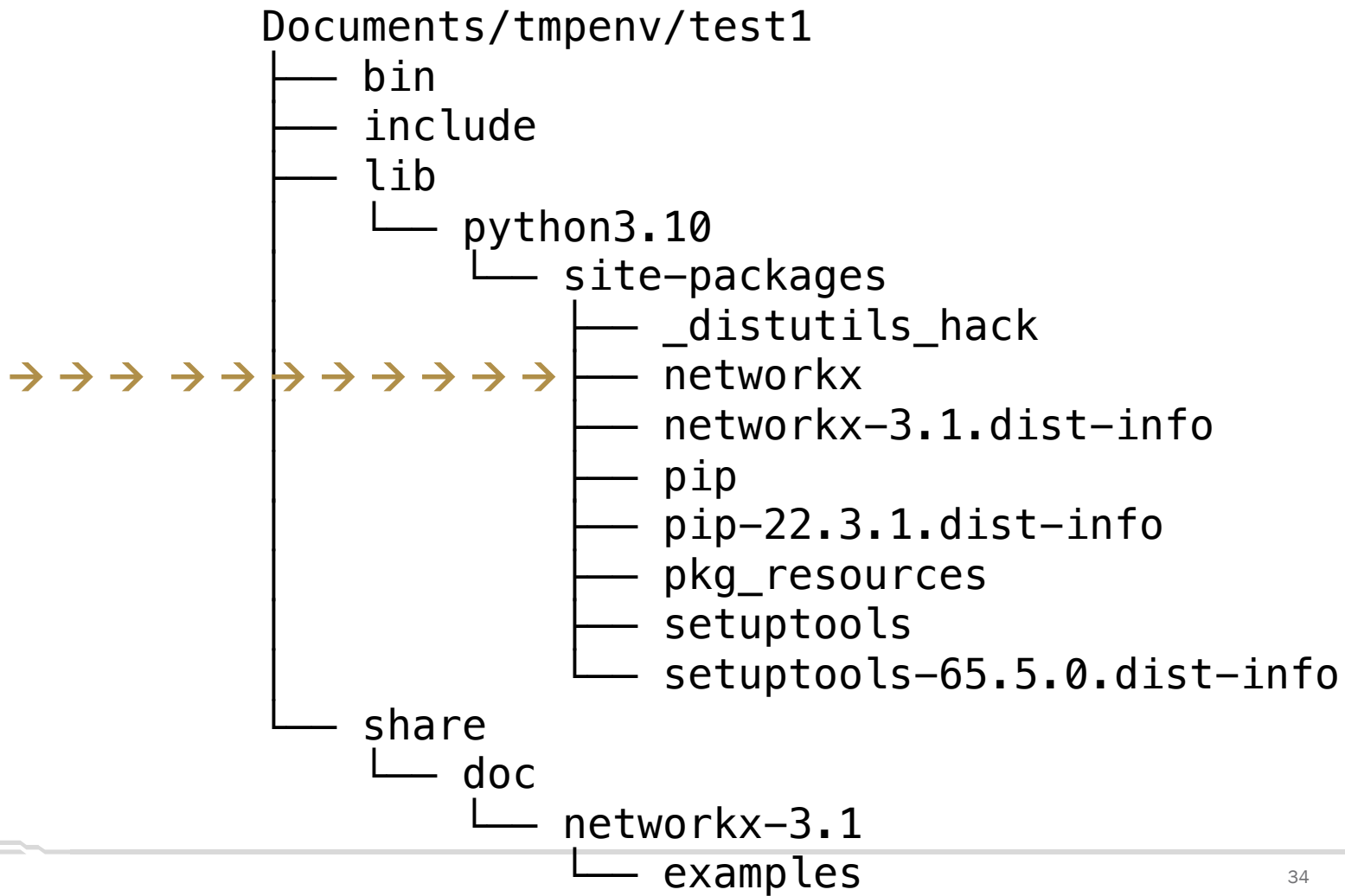
Hochschule Reutlingen

Install a p:ackage from https://pypi.org into the virtual environment:

pip install *packegname*

Example:

```
pip install networkx
```



Python Environment Management

Informatik
Hochschule Reutlingen

A smart way to manage environments is `conda`

https://docs.conda.io/en/latest/

```
conda create --name >envname< python=3.11
conda activate >envname<
conda deactivate >envname<
```

CONDA https://docs.conda.io/projects/conda/en/latest/index.html

Python Environments

Informatik
Hochschule Reutlingen

- https://pypi.org
- https://jupyter.org — pip install jupyter
- https://www.nltk.org — pip install nltk
- https://pytorch.org — pip install torch
- https://huggingface.co — pip install transformers

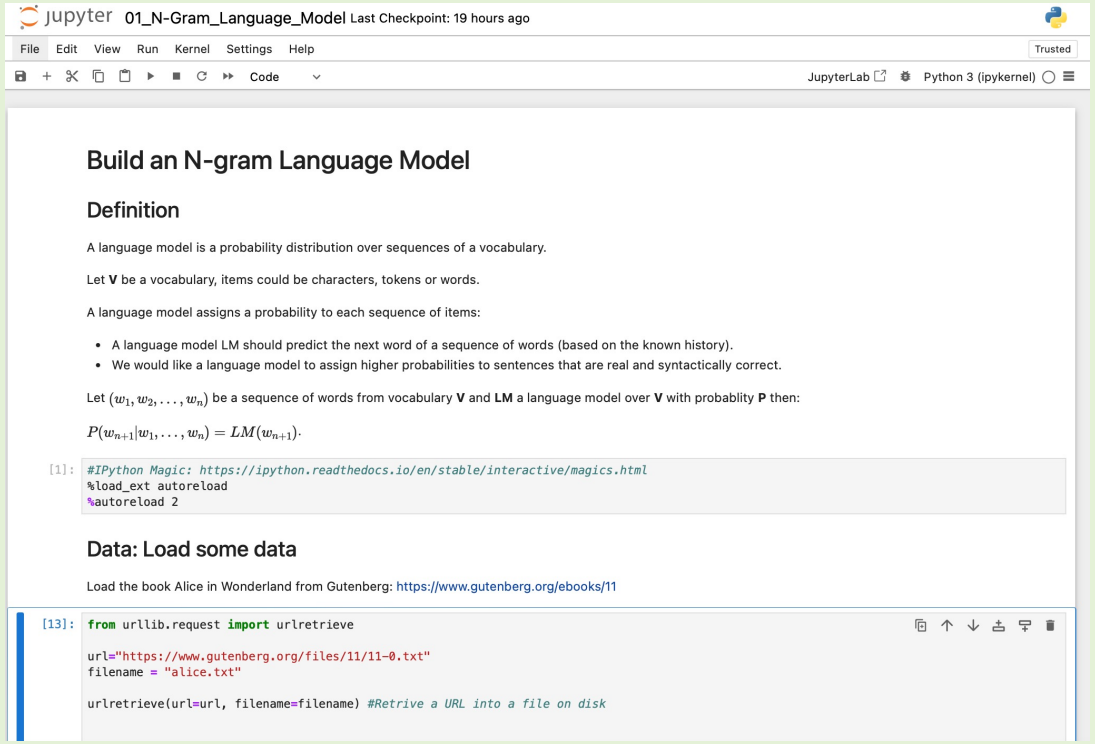
Github Repository

Our github Repo: https://github.com/dneumnn/ws24

N-Gram Notebook

Informatik
Hochschule Reutlingen

01_N-Gram_Language_Model.ipynb



06
Neural Language Models

INF Informatik
Hochschule Reutlingen

Neutral Language Models

Hochschule Reutlingen

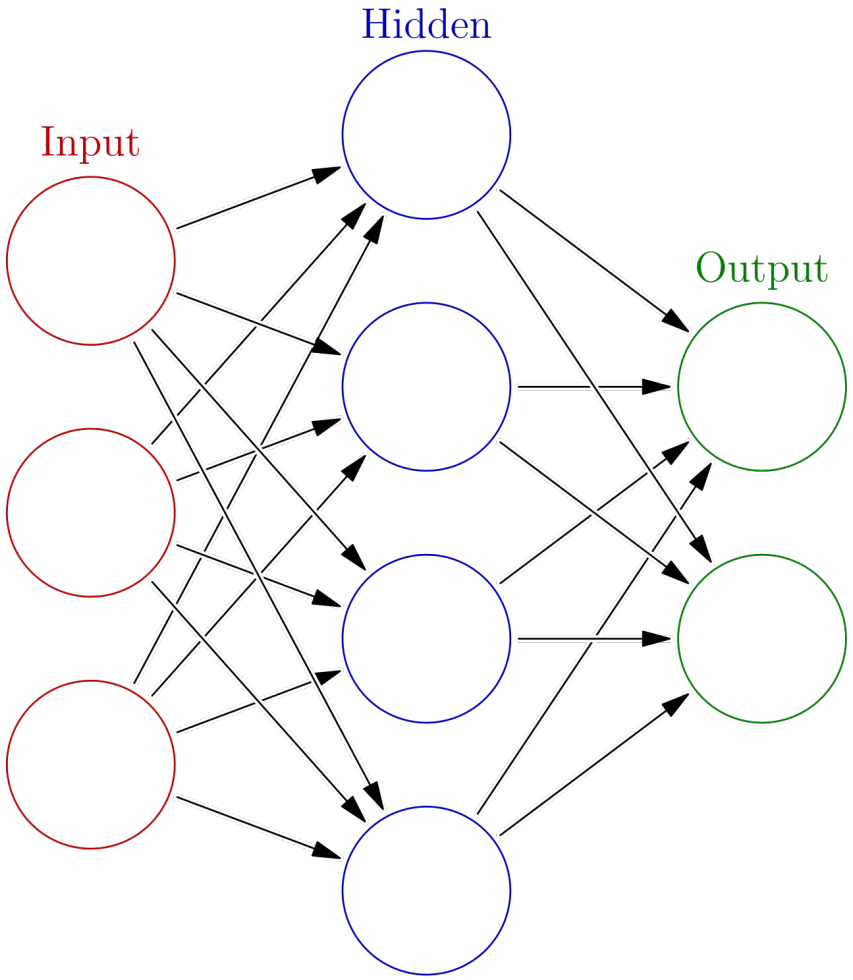
Recap: Need to define, how to compute $P(x_n|x_1, x_2, \ldots, x_{n-1})$.

history or context

- N-Gram
- Neural Models (neural networks)

Source: https://lena-voita.github.io/nlp_course/language_modeling.html

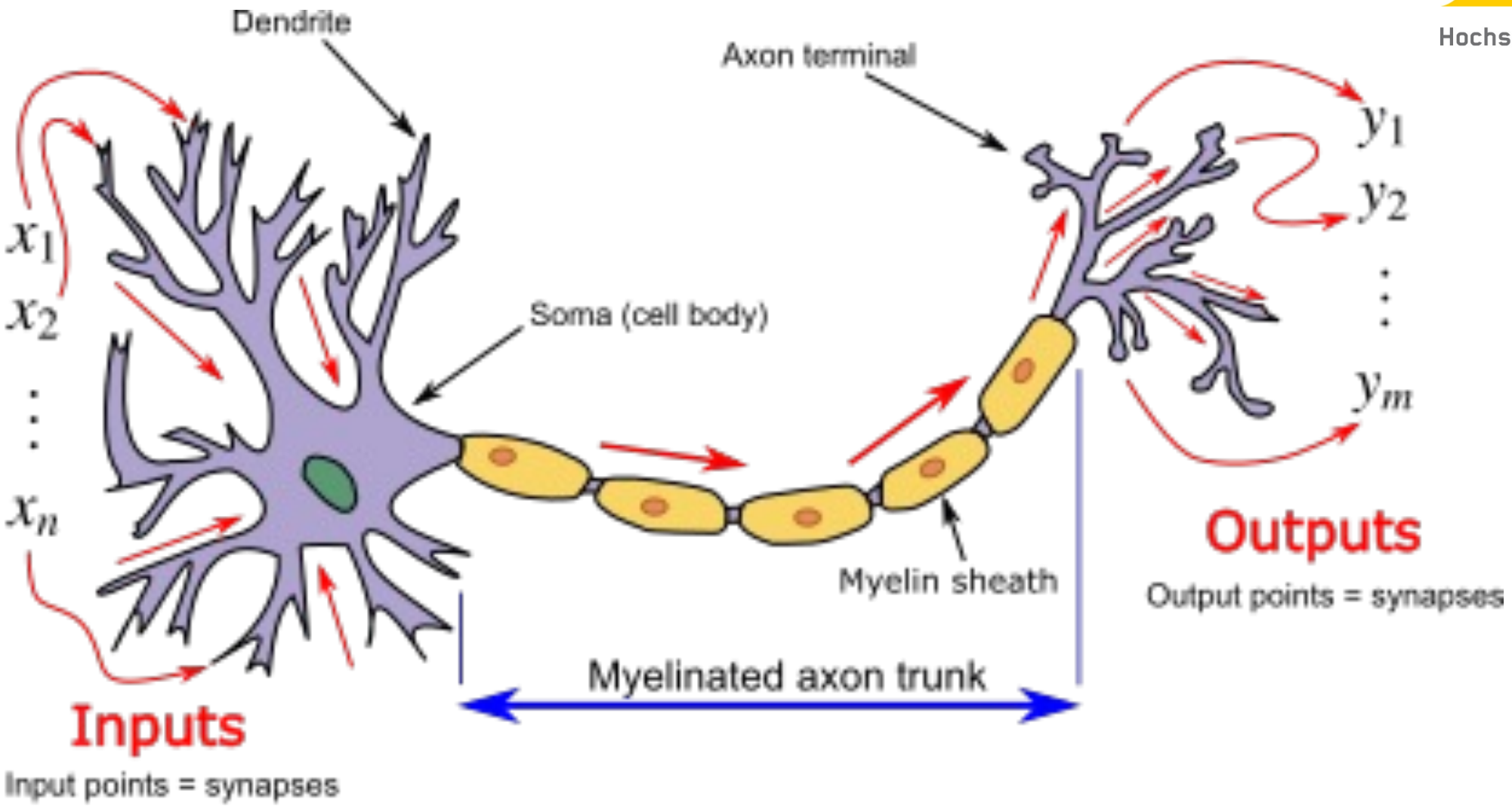
A very short Introduction into Neural Networks



Source: https://en.wikipedia.org/wiki/Neural_network_(machine_learning)

- In machine learning, a neural network (NN) is a model inspired by the structure and function of biological neural networks in animal brains.
- An NN consists of connected units or nodes called artificial neurons, which loosely model the neurons in the brain.
- These are connected by edges, which model the synapses in the brain.
- Each artificial neuron receives signals from connected neurons, then processes them and sends a signal to other connected neurons. The "signal" is a real number, and the output of each neuron is computed by some non-linear function of the sum of its inputs, called the activation function.
- The strength of the signal at each connection is determined by a weight, which adjusts during the learning process.

A very short Introduction into Neural Networks

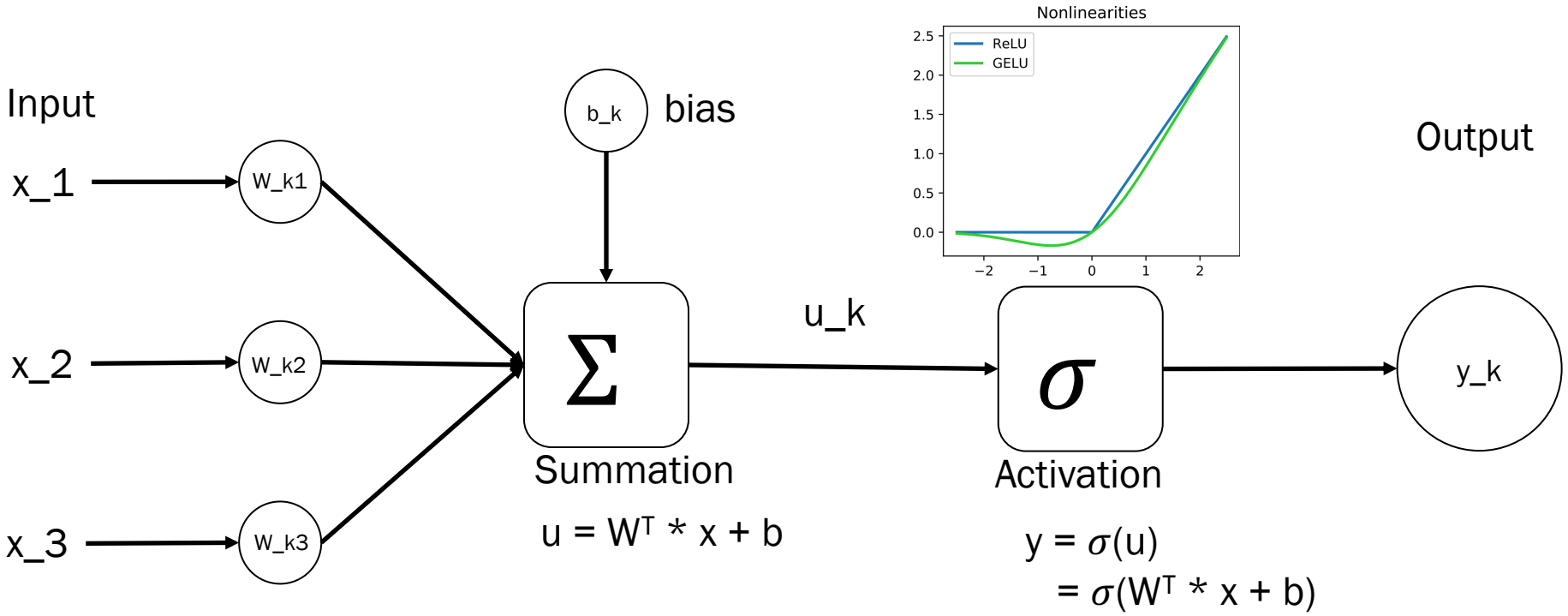


Source: https://en.wikipedia.org/wiki/Neural_network_(machine_learning)

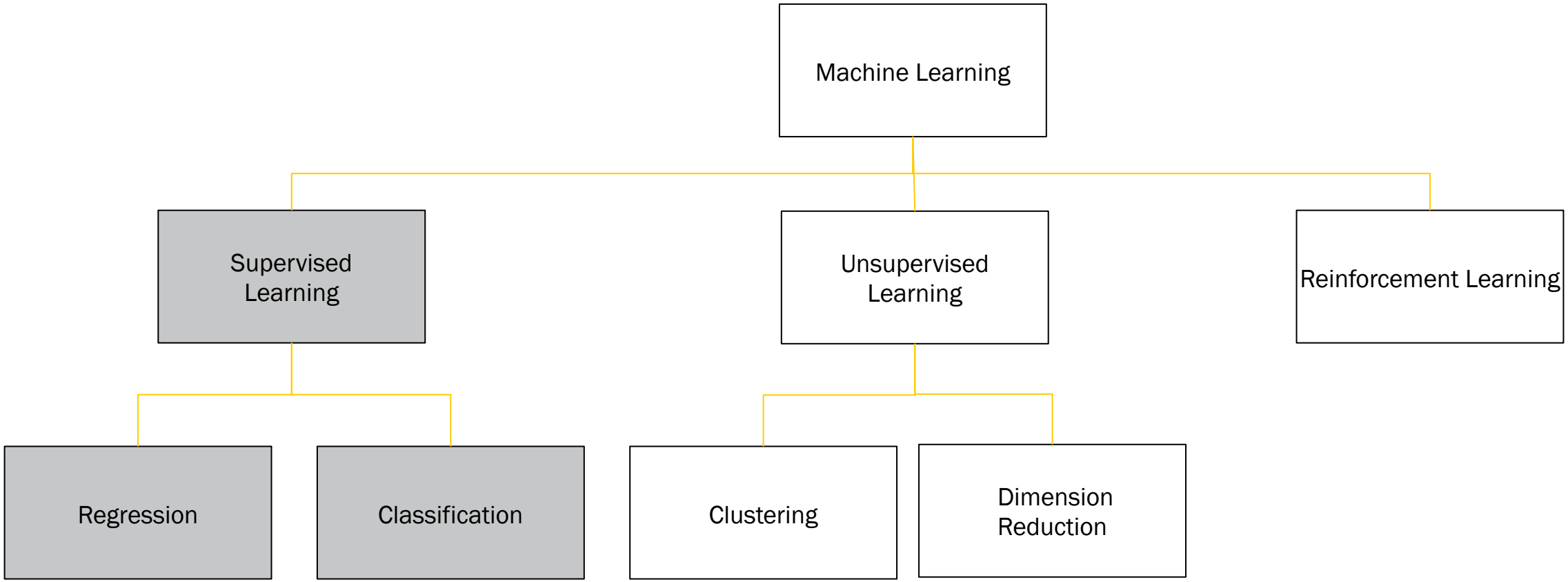
A very short Introduction into Neural Networks

Hochschule Reutlingen

Each neuron of the hidden layer does the following computation



A very short Introduction into Neural Networks



Supervised Learning

(2) Collect (labeled) Data

Training Data

Data Points

FEATURES

LABELS

(1) Define the problem to be solved

Unknown Pattern

$f: X \rightarrow Y: f(x) = y$

(3) Chose Algorithm and Optimization Metric

ML Algorithm
Model.fit(X,y)

Hyperparameter:
- Learning Rate
- epochs
…

Model
(Hypothesis)

$h: X \rightarrow Y: h(x;\ \theta) = \hat{y}$

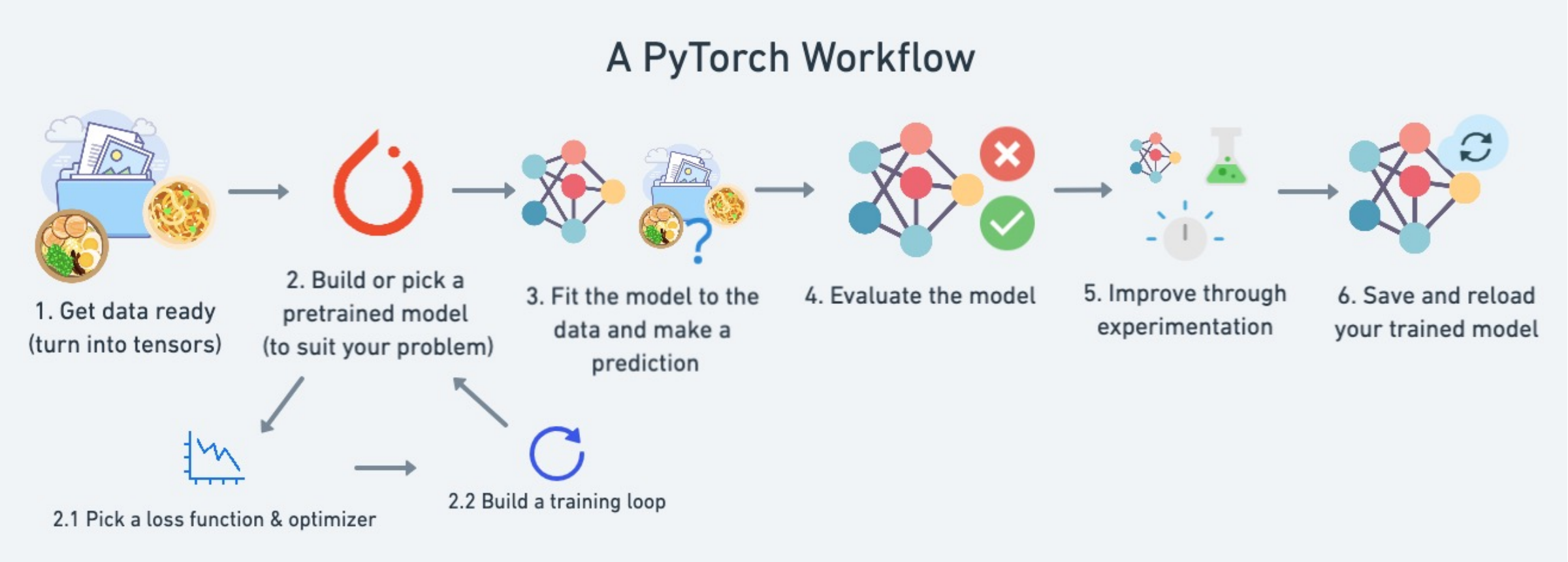
Cost / Loss
-function

$J(\theta) = \sum L(y, h(x;\ \theta))$

(4) Chose a Metric to evaluate the final model

New Data → Predictive Model **Model.predict(X)** → **Prediction**

Neural Networks with PyTorch

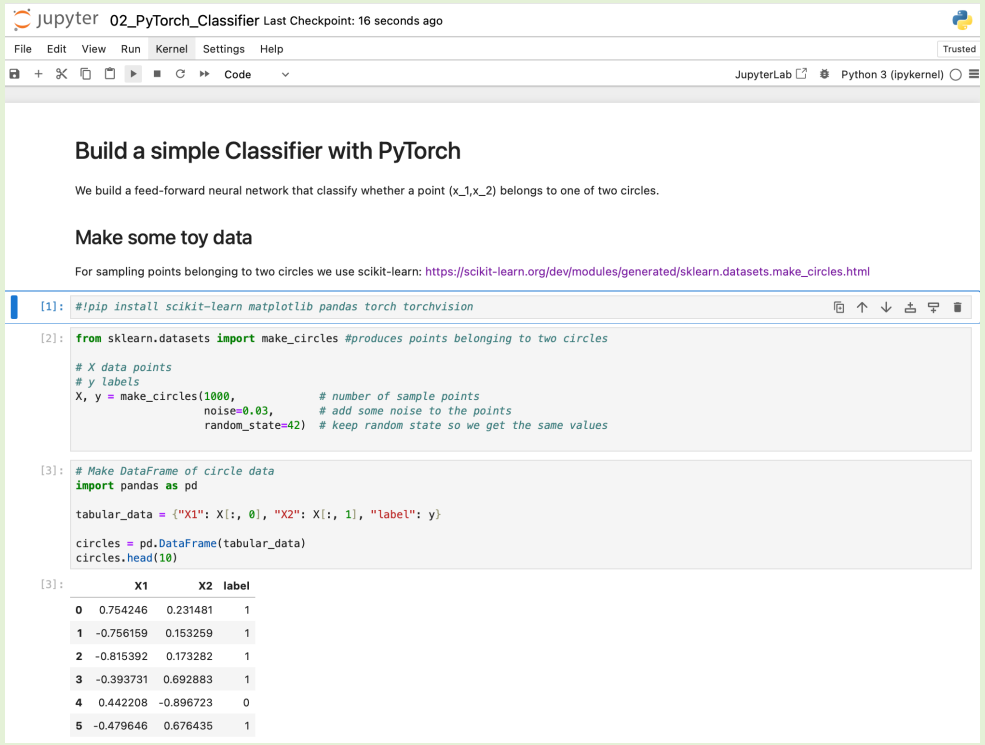


Source: https://www.learnpytorch.io/01_pytorch_workflow/

A very short Introduction into Neural Networks

Informatik
Hochschule Reutlingen

Build a Neural Classifier with PyTorch



Source: https://www.learnpytorch.io/01_pytorch_workflow/

Neutral Language Models

Informatik
Hochschule Reutlingen

Intuitively, neural Language Models do two things:

1. process context $(x_1, x_2, \ldots, x_{n-1})$ → model-specific

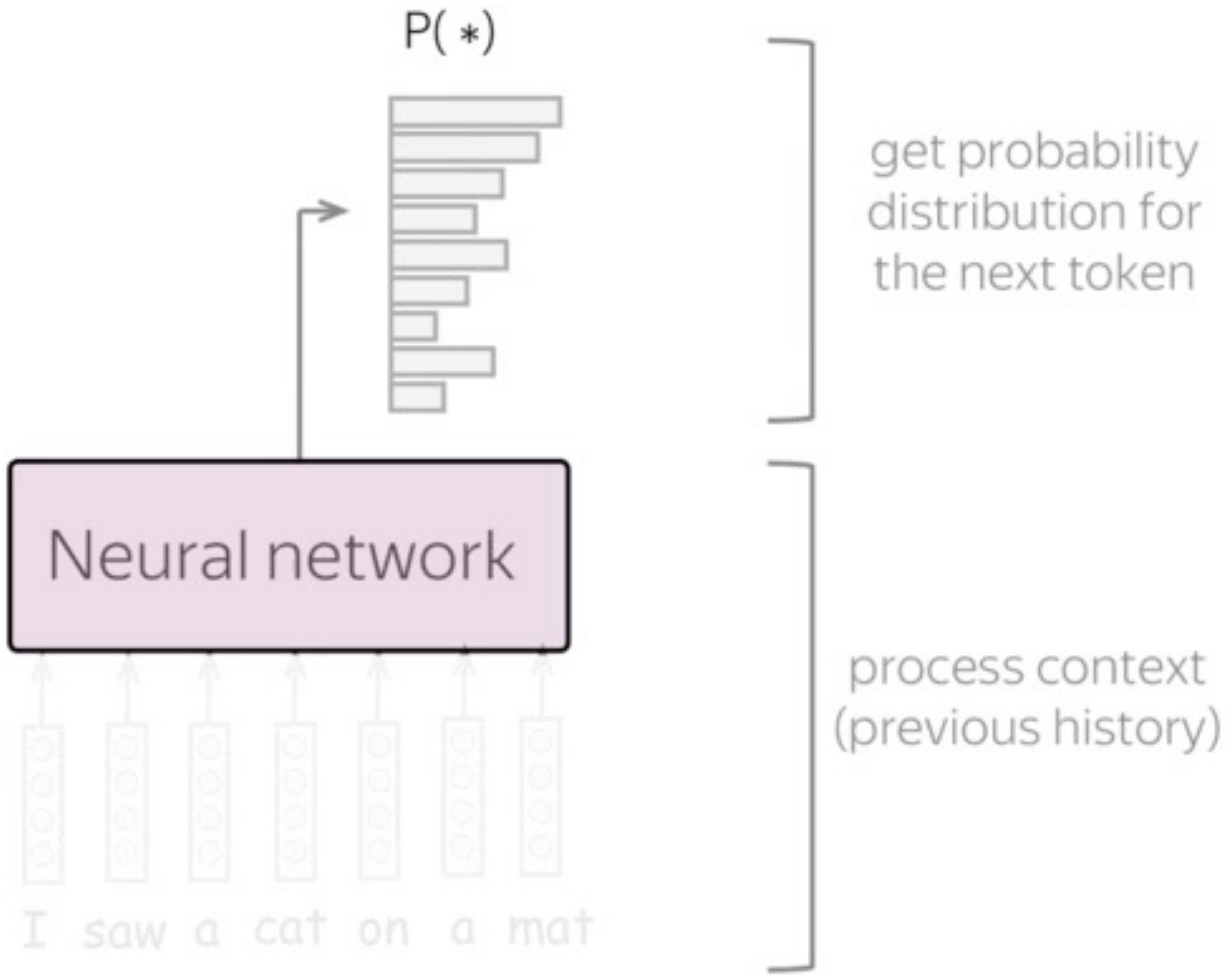
   The main idea here is to get a vector representation for the previous context. Using this representation, a model predicts a probability distribution for the next token. This part could be different depending on model architecture (e.g., RNN, CNN, whatever you want), but the main point is the same - to encode context.

2. generate a probability distribution for the next token (x_n) → model-agnostic

   Once a context has been encoded, usually the probability distribution is generated in the same way - see below.

Source: https://lena-voita.github.io/nlp_course/language_modeling.html

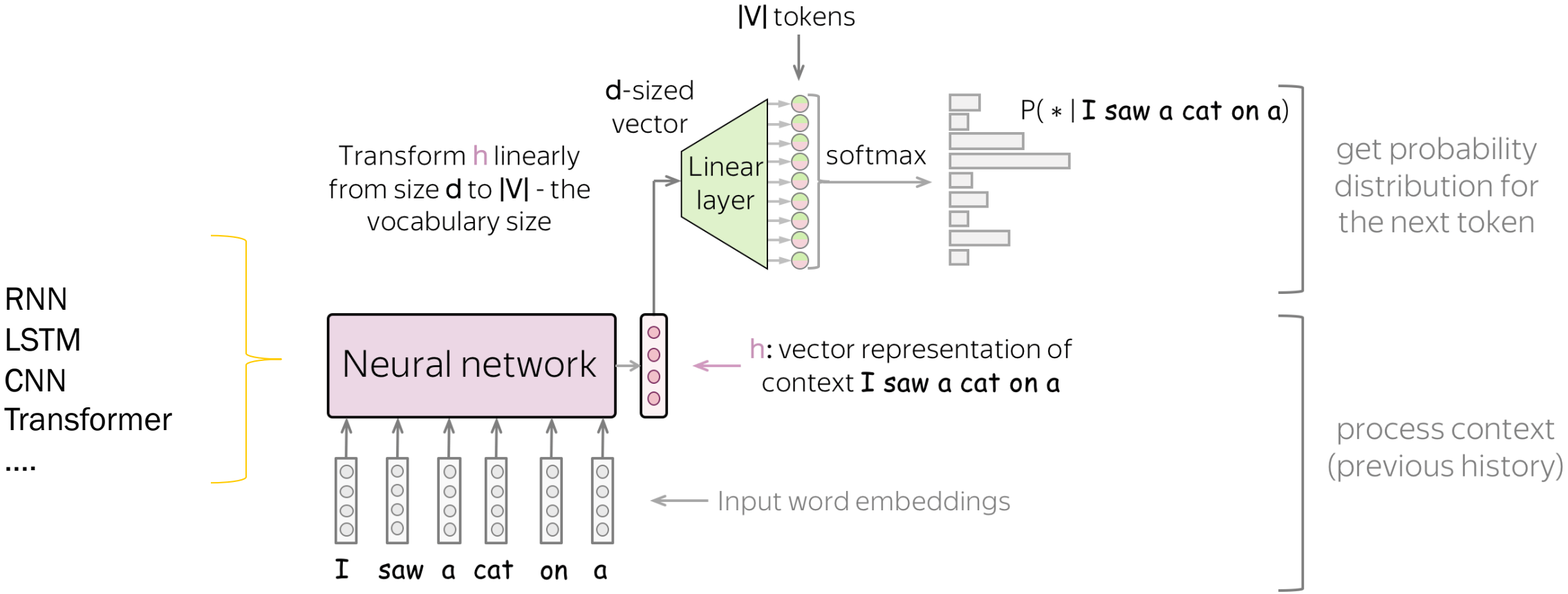
Neutral Language Models



Source: https://lena-voita.github.io/nlp_course/language_modeling.html

Neutral Language Models

We can think of neural language models as neural classifiers, where the classes are |V| vocabulary tokens.



Source: https://lena-voita.github.io/nlp_course/language_modeling.html

Neutral Language Models

Hochschule Reutlingen

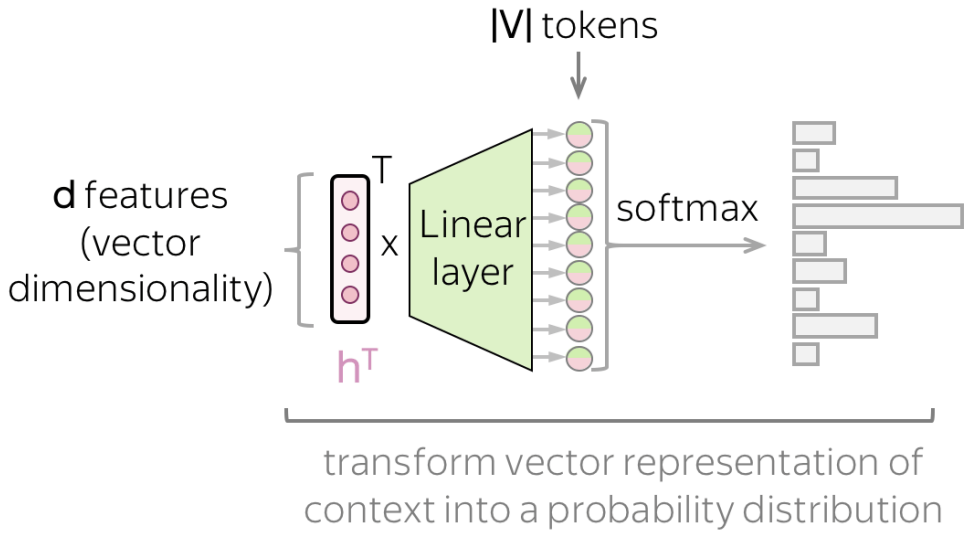
Pipeline

- feed word embedding for previous (context) words into the network
- get vector representation of context back from the network
- from this vector representation, predict a probability distribution for the next token.

We have:
- h - vector of size **d**

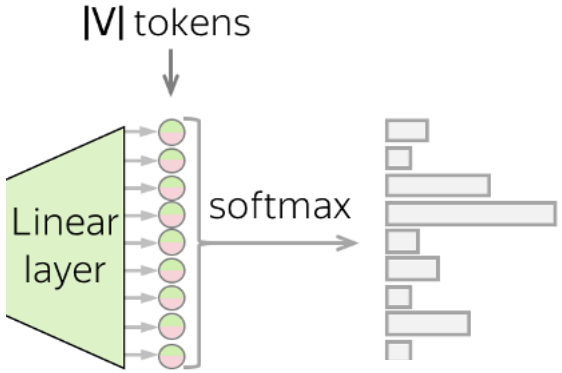
We need:
- vector of size |V| – probabilities for all tokens in the vocabulary

→ Transform linearly from size **d** to size |V|



Source: https://lena-voita.github.io/nlp_course/language_modeling.html

Using Softmax to convert logits into a probabilities:



$$\mathbf{z} \longmapsto \sigma(\mathbf{z})_i = \frac{e^{z_i}}{\sum_{j=1}^{|V|} e^{z_j}} .$$

```
import numpy as np

def softmax(z, beta=1.0):
    return np.exp(beta * z)/ np.sum(np.exp(beta * z))

z = np.array([1.0, 2.0, 3.0, 4.0, 1.0, 2.0, 3.0])
p = softmax(z)
p_max = np.argmax(p)
```

the softmax applies the standard exponential function to each element z_i of the input vector z (consisting of |V| real numbers), and normalizes these values by dividing by the sum of all these exponentials.

See: https://pytorch.org/docs/stable/generated/torch.nn.Softmax.html

Training and the Cross-Entropy Loss

See: https://lena-voita.github.io/nlp_course/text_classification.html

→ Equvalence to minimizing cross-entropy

Further Reading

Further Reading

Informatik
Hochschule Reutlingen

- Raymond Lee, “Natural Language Processing”, 2024, Springer
- Manning & Schütze, Foundations of Statistical Natural Language Processing, 1999, MIT Press
- Lena Voita, “Language Modeling” https://lena-voita.github.io/nlp_course/language_modeling.html
- Zero to Mastery Learn PyTorch for Deep Learning: https://www.learnpytorch.io
- Dive into Deep Learning: https://d2l.ai/index.html

INF Informatik
Hochschule Reutlingen

Dominik Neumann
Hochschule Reutlingen, Alteburgstraße 150, 72762 Reutlingen
www.reutlingen-university.de
T. +49 172 9861157
dominik.neumann@reutlingen-university.de
dominik.neumann@exxeta.com