Hochschule Karlsruhe
Technik und Wirtschaft
UNIVERSITY OF APPLIED SCIENCES

inovex

# AI Labor - Sommersemester 2020

## **N**atural **L**anguage **P**rocessing
## Review, Retro & Planning

Pascal Fecht, Maximilian Blanck

Karlsruhe, 15. Mai 2020

# Retrospektive

## max. 30 Minuten

- Was bewegt euch gerade?
- Was ist beim letzten Sprint gut gelaufen?
- Was können wir für die kommenden Aufgaben besser machen?
- Welche Erfahrungen könnt ihr den anderen Teams mitgeben?

https://scrumlr.io/#/join/-M7MSk7gb1EtmCzF3Wwd

# Natural Language Processing

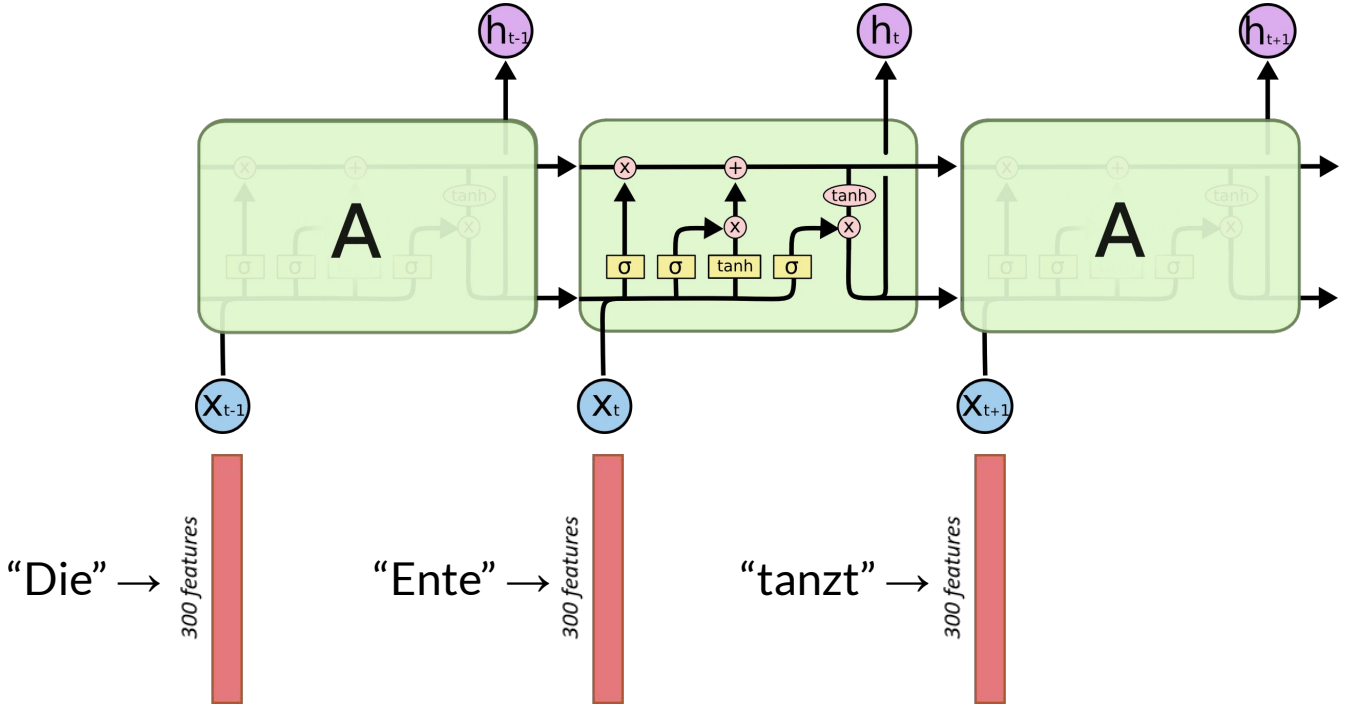
- **Theorie**
  - Sequence2Sequence Modelle
  - Transformer
  - BERT
- **Praxis**
  - Machine Translation (Englisch -> Deutsch) (Aufgabe 1)
  - Multiple-Choice Question Answering mit BERT (Aufgabe 2)

# **Aufgabe 1**
## Maschinelles Übersetzen

# RNNs

## Wie kann ein NN die Historie lernen?

# LSTM - Long Short Term Memory



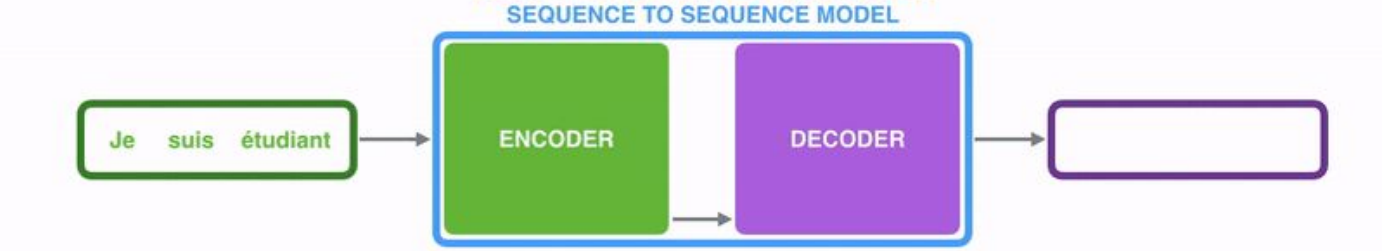



https://colah.github.io/posts/2015-08-Understanding-LSTMs/

# **Seq**uence**2Seq**uence

# Encoder-Decoder Architektur

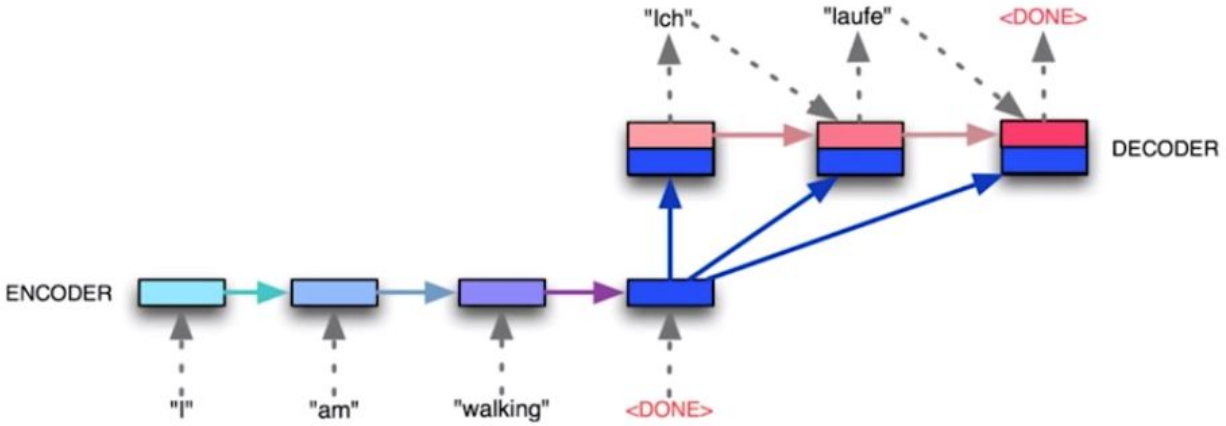
## Maschinelles Übersetzen




https://jalammar.github.io/visualizing-neural-machine-translation-mechanics-of-seq2seq-models-with-attention/

# Encoder-Decoder Architektur

## Maschinelles Übersetzen






inovex

# Batch Training von Texten

# Padding

## Problem: Nicht alle Sequenzen haben die gleiche Länge...

- **Beispiel**
  - Text_1: [“Die”, “Ente”, “tanzt”, “und” ,”quakt”]
  - Text_2: [“Die”, “Ente”, “schwimmt”]
- Die beiden Sätze sind in einem Batch und sollen an das NN gefüttert werden.
- Problem: Wir müssen mit Tensoren arbeiten, die die gleichen Dimensionen haben.
- Lösung Padding:
  - Text_1: [“Die”, “Ente”, “tanzt”, “und” ,”quakt”]
  - Text_2: [“Die”, “Ente”, “schwimmt”, ”PADDING”, ”PADDING”]
  - Batch : [[1,2,3,4,5],[1,2,6,0,0]]


colab

# Aufgabe 2

## Sequential Transfer Learning mit BERT

# Language Model

## Grundlage von BERT

word2vec, GLoVe sind **kontextfrei**

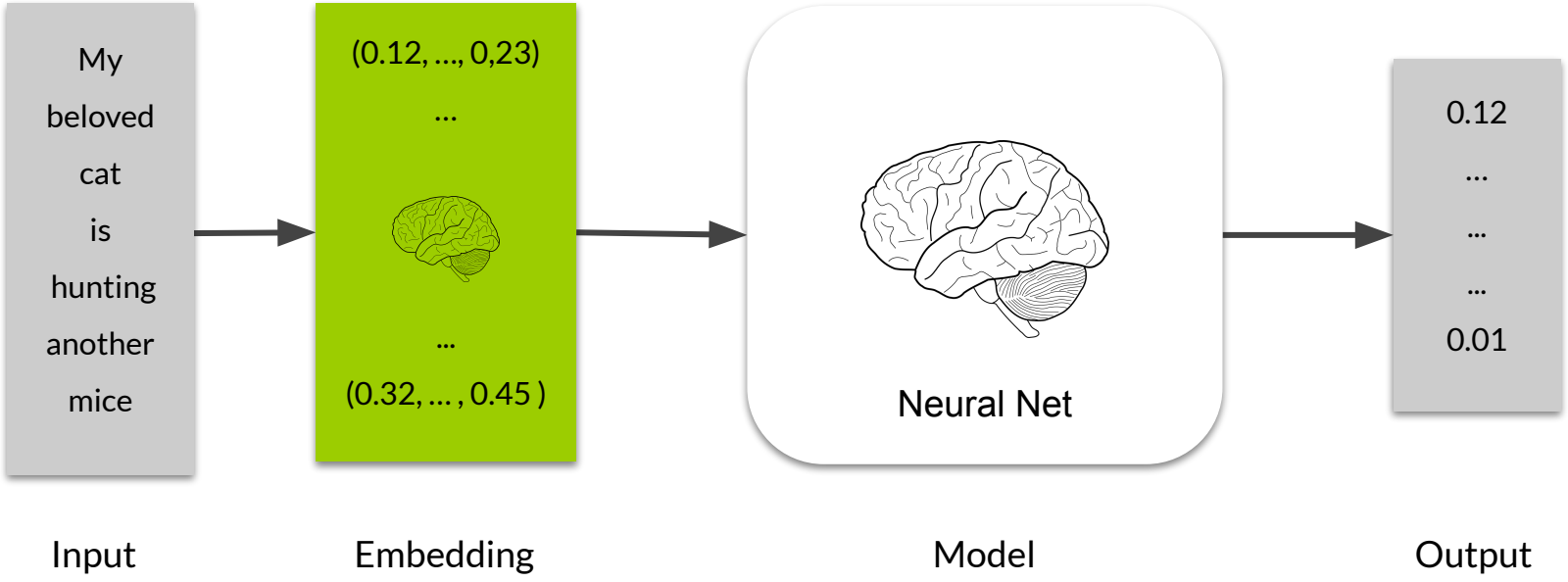
**Language Model (LM)**: Gegeben eines Kontexts, was ist das nächste Wort?

- $C_1$: “the weather was <*target*> .” ⇒ target = hot
- $C_2$: “the <*target*> dog was delicious.” ⇒ target = hot

⇒ Semi-Supervised Task
⇒ Große LMs sind Grundlage für Transfer Lernen in NLP
⇒ Sprachmodelle können bidirektional sein.

# Contextual Embeddings

## Integration in bestehende Architekturen






inovex

# Sequential transfer learning

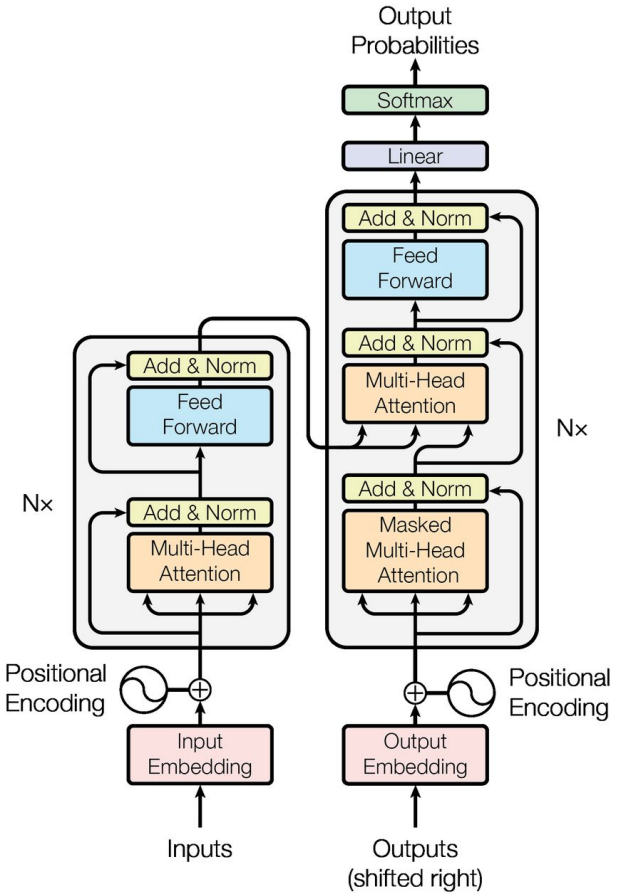
## Nutze die volle Power des vortrainierten Modells




inovex

# Transformers

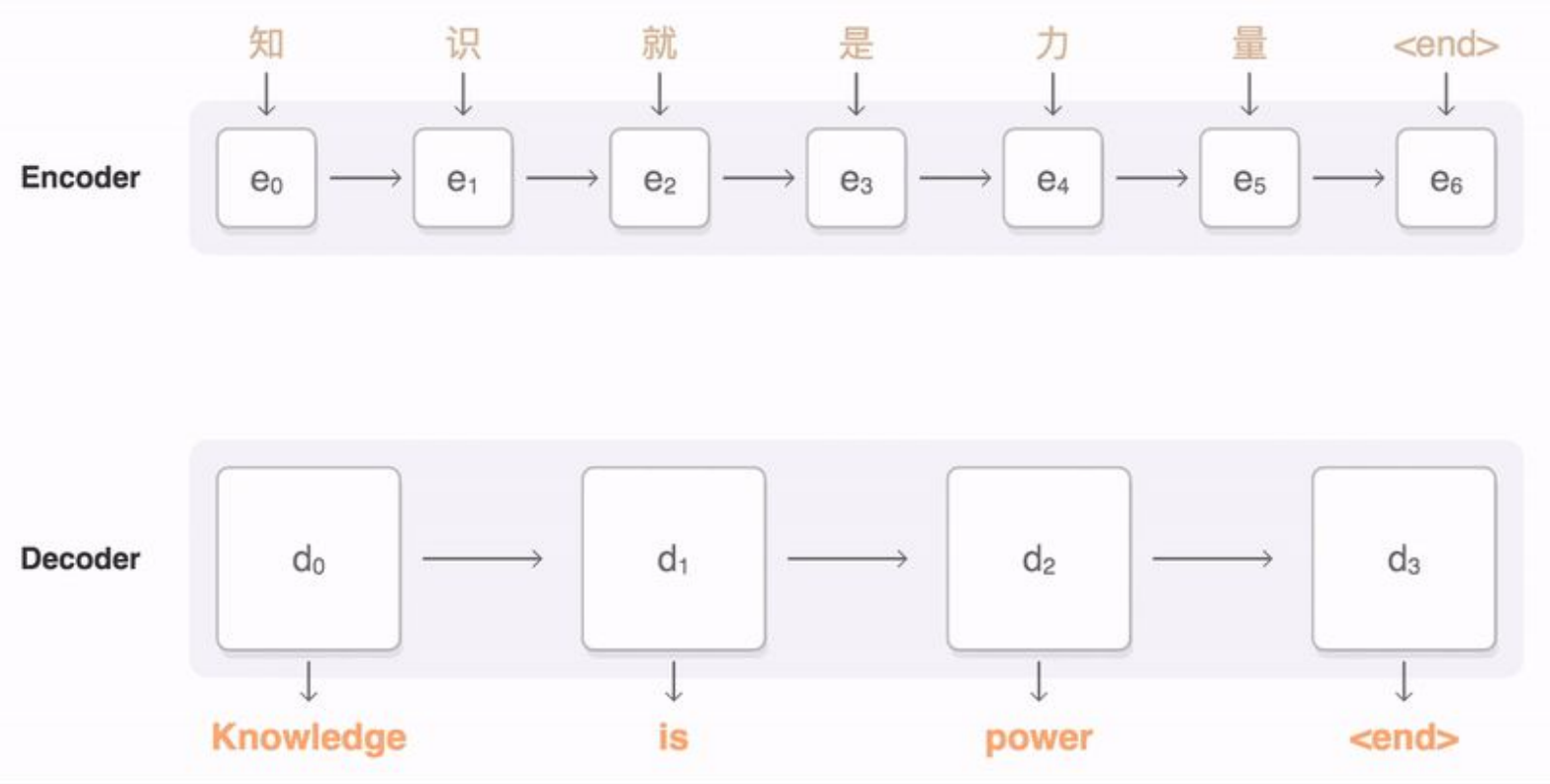
## Grundlage für BERT






https://arxiv.org/abs/1706.03762 | http://jalammar.github.io/illustrated-transformer/

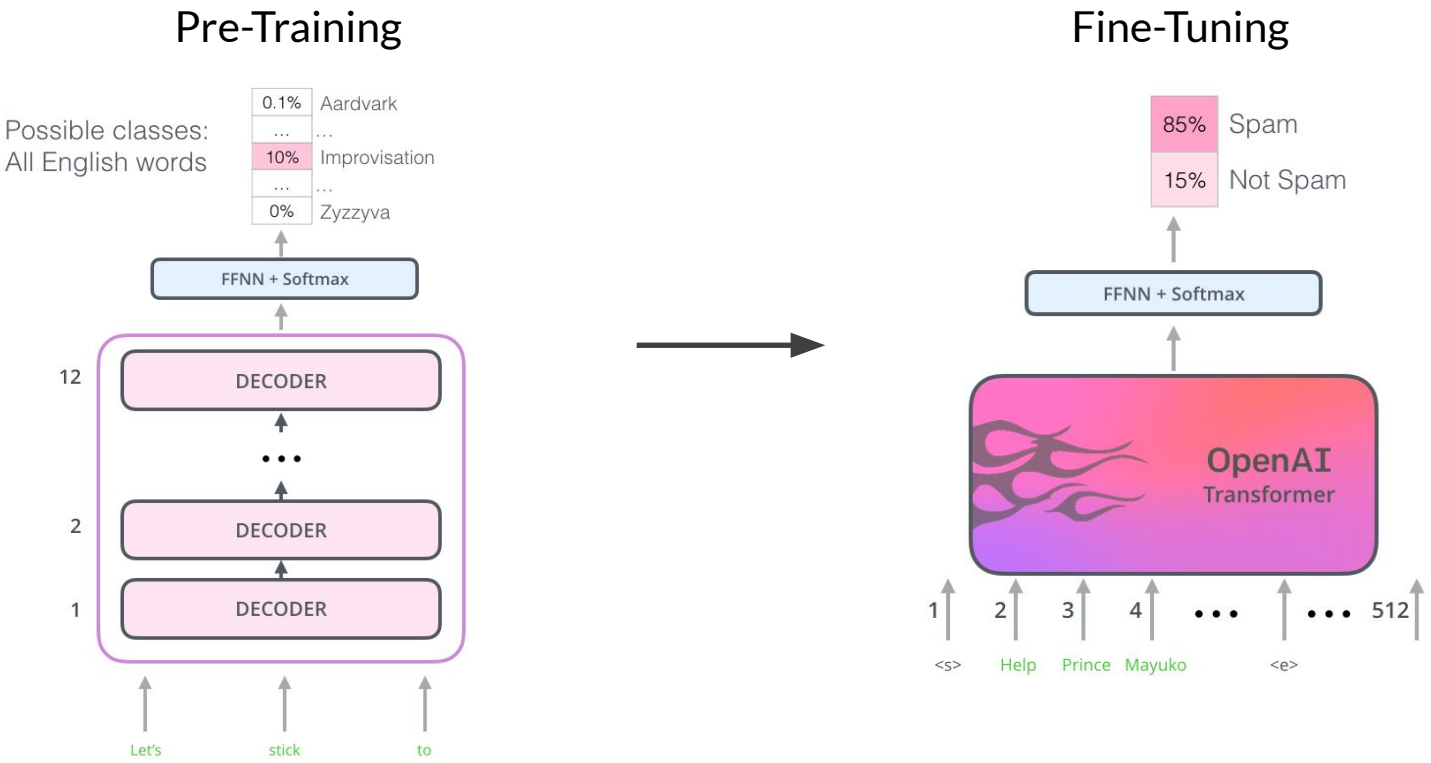
# Ausflug Attention






inovex

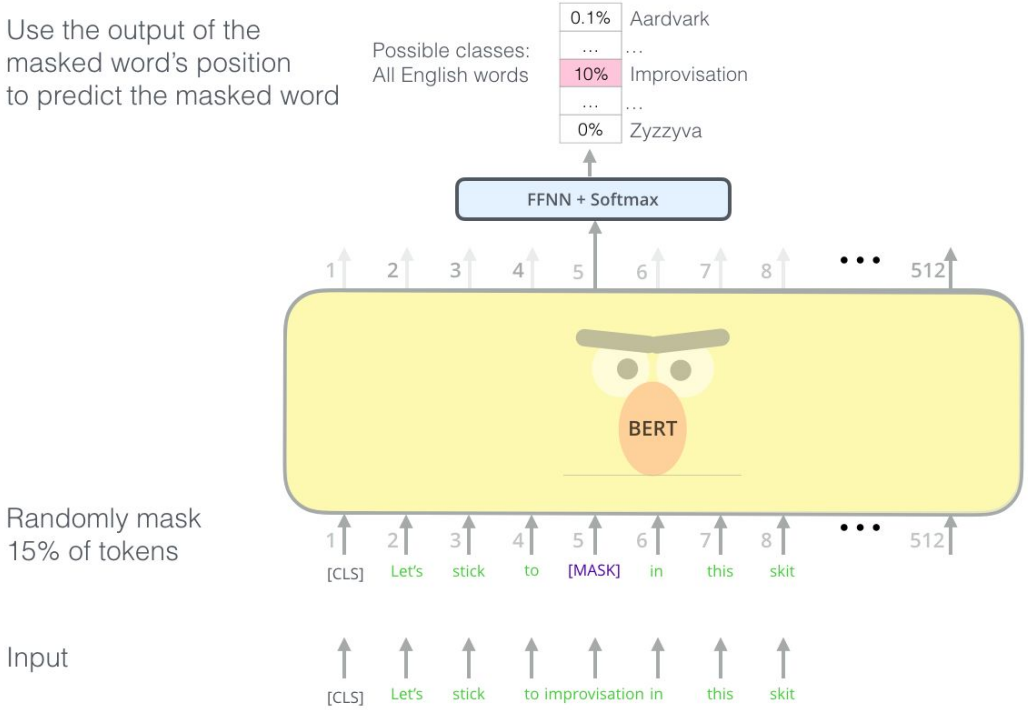
# OpenAI Transformer: Language Model






https://jalammar.github.io/illustrated-transformer/

# BERT






https://jalammar.github.io/illustrated-bert/

# BERT

Zwei Modelle: BASE (110M param) und LARGE (340M param)

Dank Google's Infrastruktur:

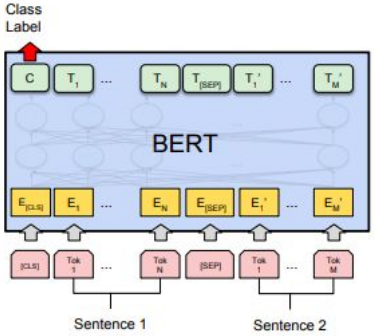
- BERT-BASE trainiert auf 4 Cloud TPUs (16 TPU chips total).
- BERT-LARGE trainiert auf 16 Cloud TPUs (64 TPU chips total)

⇒ Pre- training hat insgesamt **4 Tage** gedauert.

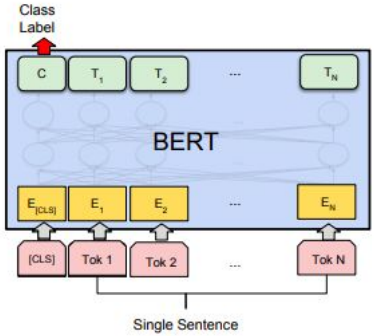


https://goo.gl/language/bert

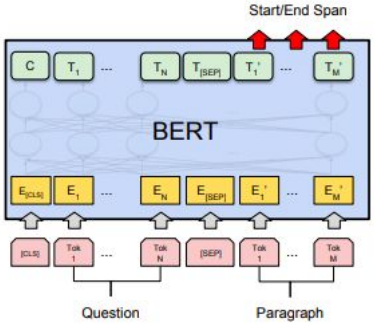
# Bert Anwendungen



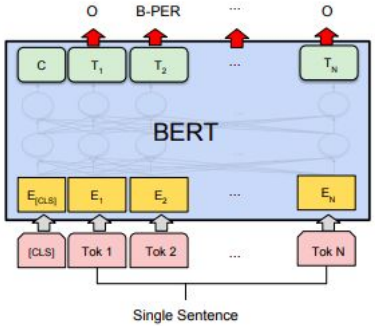

(a) Sentence Pair Classification Tasks: MNLI, QQP, QNLI, STS-B, MRPC, RTE, SWAG




(b) Single Sentence Classification Tasks: SST-2, CoLA




(c) Question Answering Tasks: SQuAD v1.1




(d) Single Sentence Tagging Tasks: CoNLL-2003 NER

Feature Extraktion (Wie ELMo)

Oder:

a) Sentence Pair classification

b) Sentence classification

c) Question answering

d) Sentence tagging

# BERT

## Adaptionen und Weiterentwicklungen

- Robustere BERT Modelle (z. B. ROBERTa)
- Verschiedene Sprachen (z.B. CamemBERT)
- Verkleinerung des Modells (z.B. DistillBERT)
- Tausende weitere (siehe https://github.com/tomohideshibata/BERT-related-papers)

# Aufgabe 2

# Huggingface

## Transforms

› Einfache API:

```
# Store the model we want to use
MODEL_NAME = "bert-base-cased"

# We need to create the model and tokenizer
model = AutoModel.from_pretrained(MODEL_NAME)
tokenizer = AutoTokenizer.from_pretrained(MODEL_NAME)
```

**Tokenizer**: Wandelt Eingabetext in Tokens und Metadaten um

**Modell**: Lädt das vortrainiertes Modell in TF oder pytorch

› Modell Name als ID ⇒ Hunderte Modelle für unterschiedliche Tasks und Sprachen (https://huggingface.co/models)

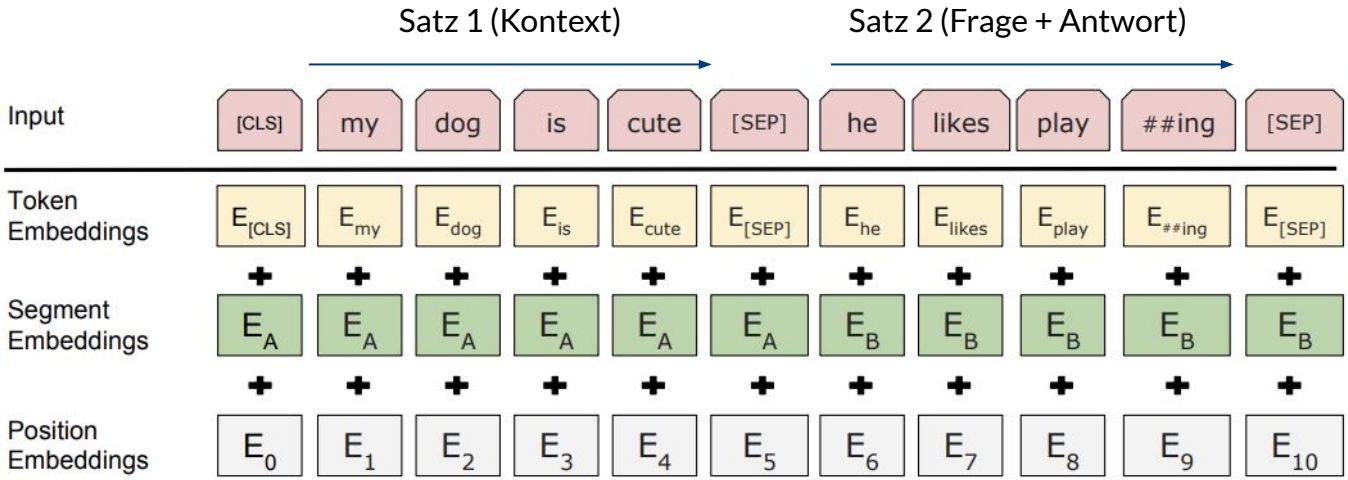
# SWAG Datensatz

## Beispiel

```
Context:        Standing on their hind legs, the bears swap a look.
Beginning:      He
Ending 0:       walks the boy through a door.
Ending 1:       flips the book back.
Ending 2:       leads her into the kitchen.
Ending 3:       goes back and forth.
```

# BERT

## Eingabe-Encoding






(Devlin et al., 2018) https://arxiv.org/abs/1706.03762


colab

# Feedback

https://forms.gle/sdikyENt3KKEo2x7A

# Vielen Dank

Pascal Fecht
pfecht@inovex.de

Maximilian Blanck
mblanck@inovex.de

Credits an:
Anna Weisshaar
Tilman Berger