Workshop #2

Amaru Cuba Gyllensten
amaru.cuba.gyllensten@ri.se

Ariel Ekgren
ariel.ekgren@ri.se

Översikt

1. Introduktion
2. Presentation
3. NLU-Colab
4. Pilot smartflow

Språkmodeller

Amaru Cuba Gyllensten
amaru.cuba.gyllensten@ri.se

Ariel Ekgren
ariel.ekgren@ri.se

Översikt

1. Vad är NLU?
2. Hur görs NLU?
   a. Transformers
   b. Förträning
   c. Finetuning
3. NLU i praktiken

1. **Vad är NLU?**
2. Hur görs NLU?
   a. Transformers
   b. Förträning
   c. Finetuning
3. NLU i praktiken

Natural Language Understanding

NLU är en gren inom ML för att lösa en mängd språkförståelse-relaterade problem:

- Klassifikation
- Sammanfattning
- Namnigenkänning
- Översättning

Natural Language Understanding

Idag används förtränade språkmodeller till nästan alla NLU-problem.

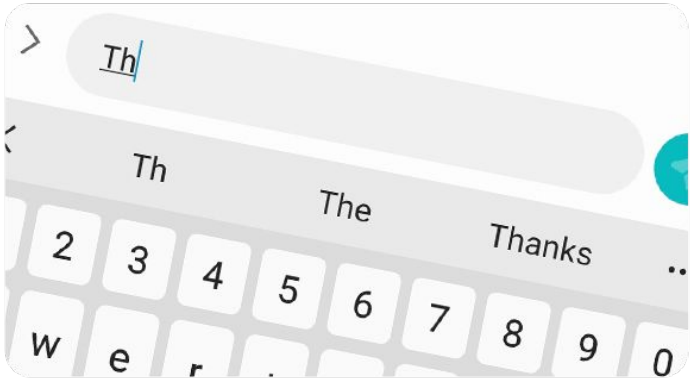
Vad är en språkmodell?

Språkmodeller i korthet

Abstrakt: Språkmodeller är modeller som lär sig språkets *distribution*.

Konkret: Språkmodeller är modeller som “gissar” ord, givet en kontext.

Språkmodeller i korthet



- Gissa nästa ord:

  “.. ett höganäskrus i en svångrem om **???**”

- Gissa ord i kontext:

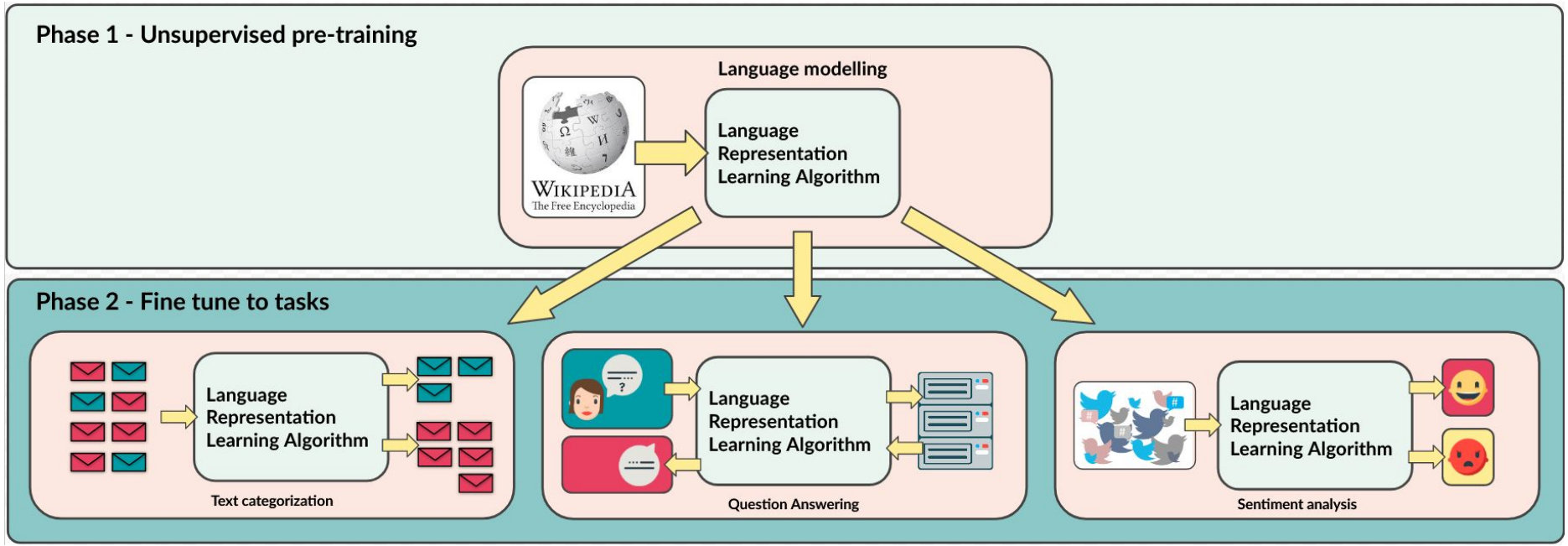
  “... ett **???** i en svångrem om halsen …”

Språkmodeller i korthet

Språkmodellen i sig är sällan intressant.

Språkmodellen är grunden med vilket vi bygger NLU-lösningar!

Varför språkmodeller?



Varför språkmodeller?

Språkförståelse är *nödvändigt* för dokument-klassificering, fråga-svarssystem, sentimentanalys et.c.

Språkförståelse är *nödvändigt* för språkmodellering.

En tillräckligt bra språkmodell har lärt sig “förstå språk” till den grad att den enklare kan anpassas till mer specifika problem.

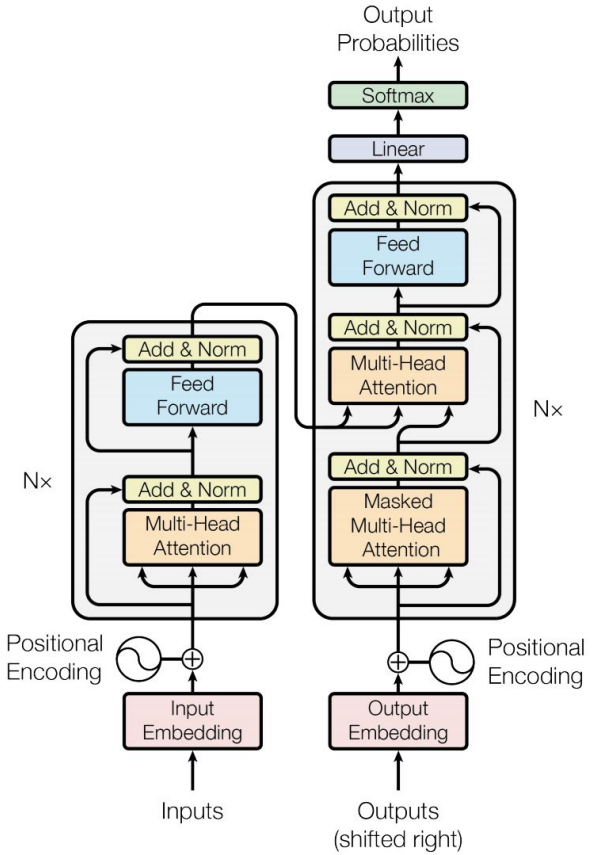
RI.
SE

1. Vad är NLU?
2. **Hur görs NLU?**
   a. **Transformers**
   b. Förträning
   c. Finetuning
3. NLU i praktiken

NLU ❤️ Transformers

Den “bästa” familjen av språkmodeller för tillfället är Transformers.

Transformer-modellen introducerades 2019 med artikeln Attention is All You Need.



Vaswani, Ashish, et al. "Attention is all you need." Advances in neural information processing systems. 2017.

Text -> Tokens

För att hantera text behöver man konvertera den till något ett neuralt nätverk kan använda.

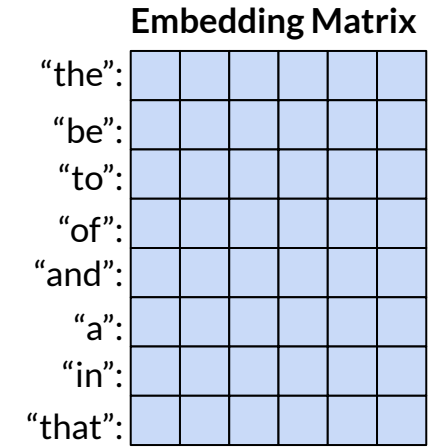
Detta görs genom tokenisering. Tokenisering styckar upp en text i delar, eller tokens. t.ex. “en katt satt på en hatt” -> [en, katt, satt, på, en, hatt].

Tokenisering ger oss också ett vokabulär av alla tokens som förekommer i datan.

Tokens -> Embeddings

Efter tokenisering har vi ett vokabulär av tokens.

För varje token skapar vi en “embedding”. Detta är en parameteriserad vektor, som stoppas in som input i neurala nätverk.

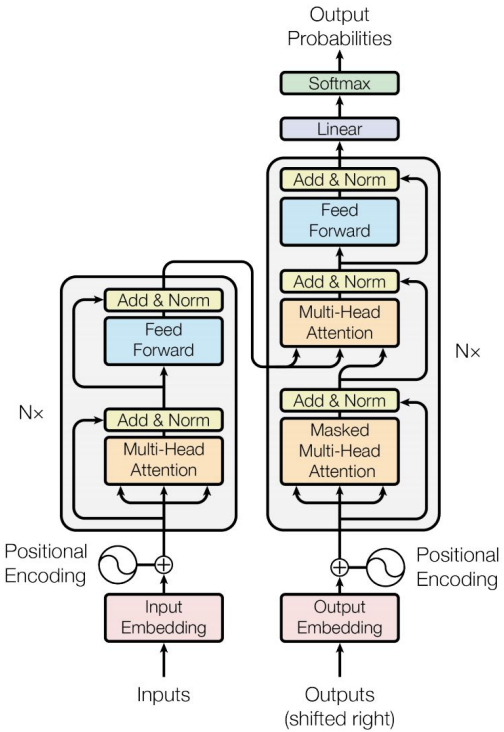


vocabulary size **x** embedding dimension

Embeddings -> Representation

Embeddings stoppas sedan in som input i själva transformern.

Output är omvandlade embeddings.



“En katt satt på en hatt.”								
<B>	En	katt	satt	på	en	hatt	.	<E>
2	59	7808	1549	68	59	9226	7	3
								
								
								

1. Vad är NLU?
2. **Hur görs NLU?**
   a. Transformers
   b. Förträning
   c. Finetuning
3. NLU i praktiken

Förträning

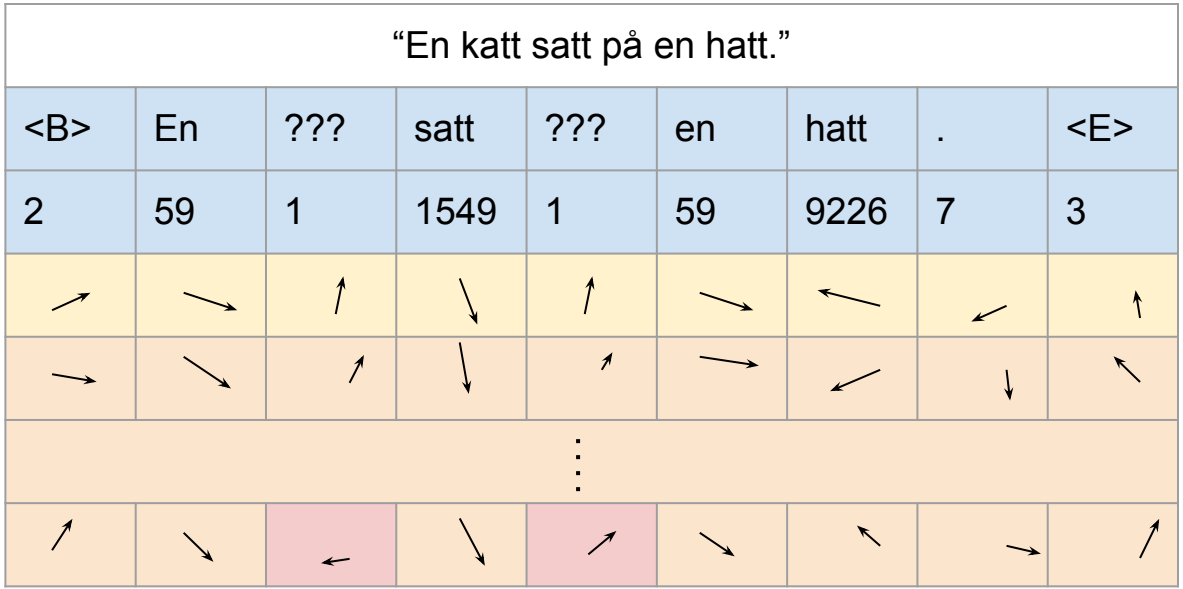
Förträning går ut på att en låter algoritmen träna på en stor mängd icke annoterad textdata.

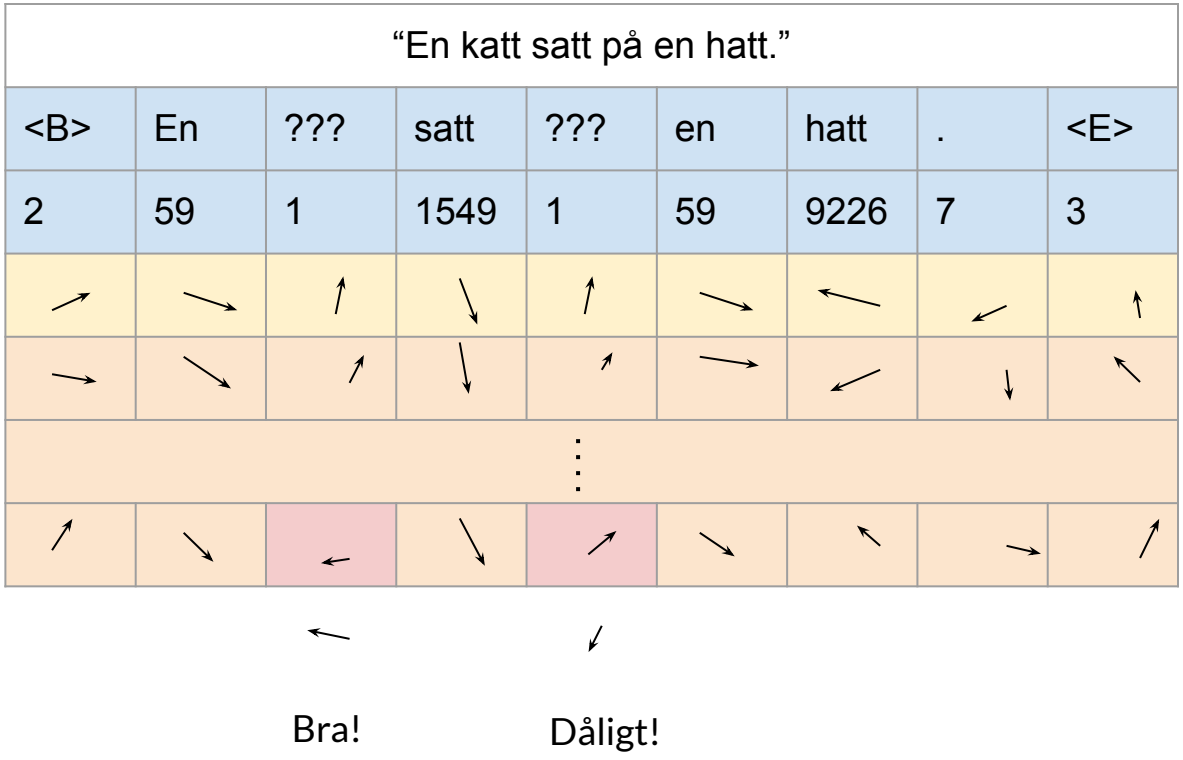
Till exempel text från Kungliga Biblioteket och wikipedia.

Detta görs genom att gissa ord i kontext.

Källa: https://xkcd.com/1838/

“En katt satt på en hatt.”								
<B>	En	katt	satt	på	en	hatt	.	<E>
2	59	7808	1549	68	59	9226	7	3
								
								
								


“En katt satt på en hatt.”
<B>
En
???
satt
???
en
hatt
.
<E>
2
59
1
1549
1
59
9226
7
3


“En katt satt på en hatt.”
<B>
En
???
satt
???
en
hatt
.
<E>
2
59
1
1549
1
59
9226
7
3
Bra!
Dåligt!

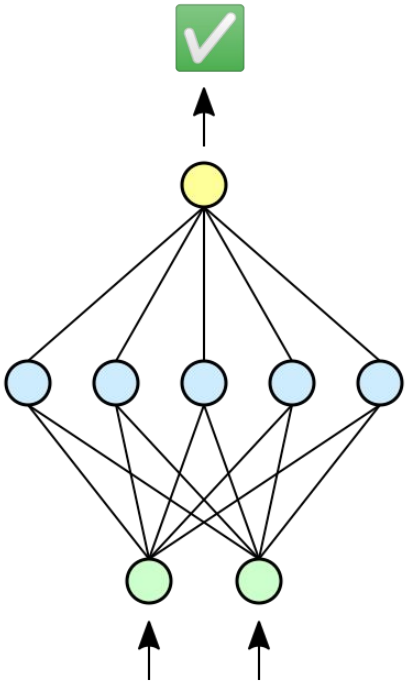
1. Vad är NLU?
2. **Hur görs NLU?**
   a. Transformers
   b. Förträning
   c. Finetuning
3. NLU i praktiken

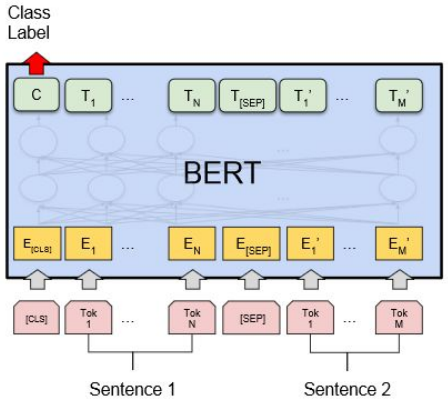
Fine-tuning

När man har en förtränad modell är det vanligaste att man finjusterar modellerna mot ett annoterat dataset. Med det förfarandet fortsätter man uppdatera modellerna mot ett specifikt mål.

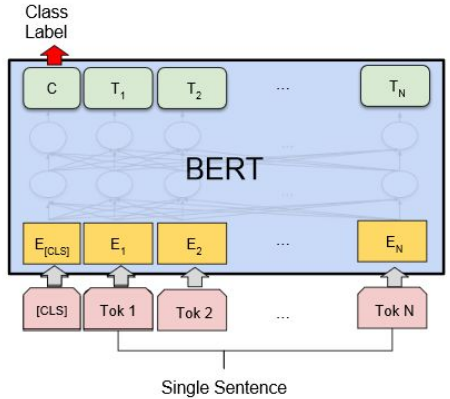
Det kräver dock tusentals annoterade exempel för att uppnå bra resultat.

Korrekt : “En katt satt på en hatt.”								
<B>	En	katt	satt	på	en	hatt	.	<E>
2	59	7808	1549	68	59	9226	7	3
⋮								

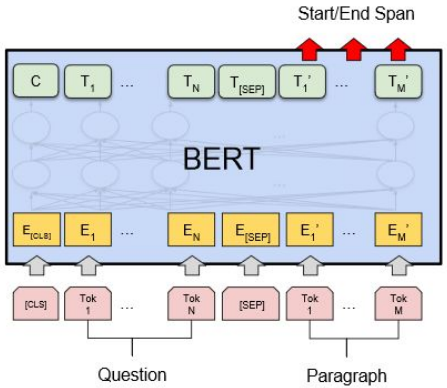




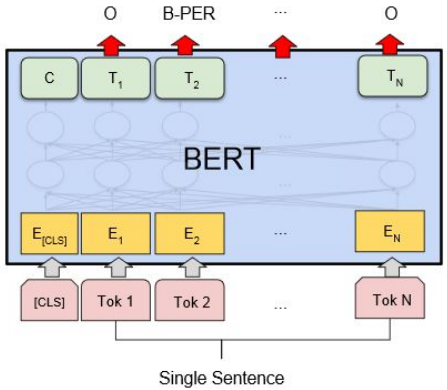
(a) Sentence Pair Classification Tasks: MNLI, QQP, QNLI, STS-B, MRPC, RTE, SWAG



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



(c) Question Answering Tasks: SQuAD v1.1

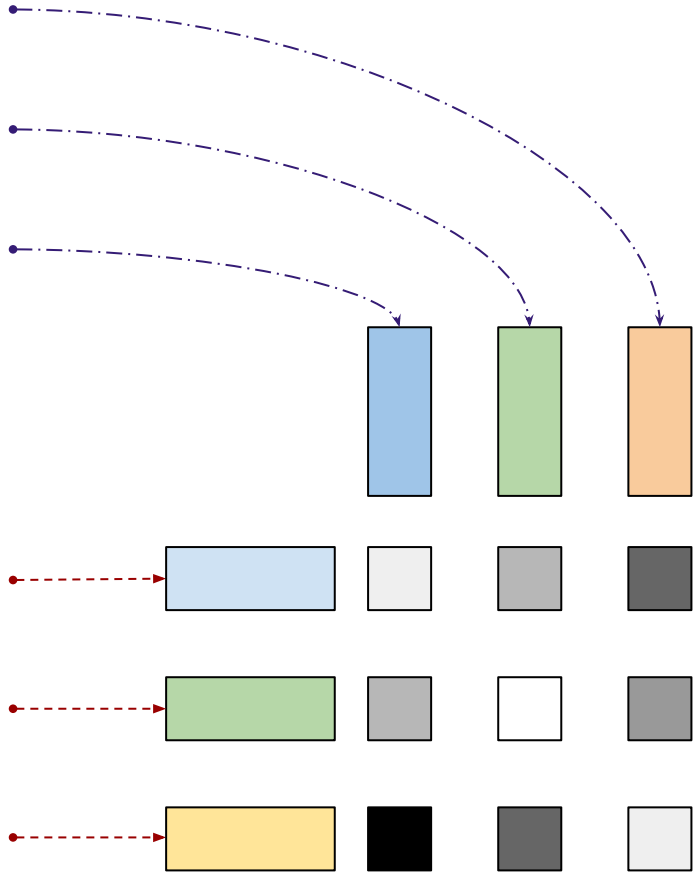


(d) Single Sentence Tagging Tasks: CoNLL-2003 NER

Hej! Återställ ditt lösenord via ...

Hej! Om du har ett öppet dokument ...

Hej! Gå till receptionen och anmäl ...



Hur återställer jag mitt lösenord?

Hur sparar jag filer i teams?

Jag har tappat bort min nyckel!

1. Vad är NLU?
2. Hur görs NLU?
   a. Transformers
   b. Förträning
   c. Finetuning
3. **NLU i praktiken**

Teckenkodning

Det finns många textkodningar:

ascii, latin-1, windows-1250, **unicode** (utf-8)

Unicode är bra (och standard)! men: **Normalisera**.

Source		NFD	NFC	Source		NFD	NFC
Å 212B	:	A ◌̊ 0041 030A	Å 00C5	Å 00C5	:	A ◌̊ 0041 030A	Å 00C5
Ω 2126	:	Ω 03A9	Ω 03A9	ô 00F4	:	o ◌̂ 006F 0302	ô 00F4

UTF-8

	_0	_1	_2	_3	_4	_5	_6	_7
(1 byte) 0_	NUL 0000	SOH 0001	STX 0002	ETX 0003	EOT 0004	ENQ 0005	ACK 0006	BEL 0007
(1) 1_	DLE 0010	DC1 0011	DC2 0012	DC3 0013	DC4 0014	NAK 0015	SYN 0016	ETB 0017
(1) 2_	SP 0020	! 0021	" 0022	# 0023	$ 0024	% 0025	& 0026	' 0027
(1) 3_	0 0030	1 0031	2 0032	3 0033	4 0034	5 0035	6 0036	7 0037
(1) 4_	@ 0040	A 0041	B 0042	C 0043	D 0044	E 0045	F 0046	G 0047
(1) 5_	P 0050	Q 0051	R 0052	S 0053	T 0054	U 0055	V 0056	W 0057
(1) 6_	` 0060	a 0061	b 0062	c 0063	d 0064	e 0065	f 0066	g 0067
(1) 7_	p 0070	q 0071	r 0072	s 0073	t 0074	u 0075	v 0076	w 0077
8_	• +00	• +01	• +02	• +03	• +04	• +05	• +06	• +07
9_	• +10	• +11	• +12	• +13	• +14	• +15	• +16	• +17
A_	• +20	• +21	• +22	• +23	• +24	• +25	• +26	• +27
B_	• +30	• +31	• +32	• +33	• +34	• +35	• +36	• +37
(2) C_	2 0000	2 0040	Latin 0080	Latin 00C0	Latin 0100	Latin 0140	Latin 0180	Latin 01C0
(2) D_	Cyril 0400	Cyril 0440	Cyril 0480	Cyril 04C0	Cyril… 0500	Armeni 0540	Hebrew 0580	Hebrew 05C0
(3) E_	Indic 0800	Misc. 1000	Symbol 2000	Kana… 3000	CJK 4000	CJK 5000	CJK 6000	CJK 7000
(4) F_	SMP… 10000	40000	80000	SSP… C0000	SPUA-B 100000	4 140000	4 180000	4 1C0000

Textdataformat

Text kan finnas i många format! PDF, Word, xml, et.c.

PDF: Jättebra utdataformat! Väldigt dåligt indataformat!

Som indata vill man helst ha strukturerad text i **unicode.** T.ex. json, csv, xml.

Huggingface

De facto standard för transformerlösningar. Huggingface är en open source lösning för språkmodeller av typen Transformers. Fungerar med både tensorflow och pytorch. De erbjuder också lösningar för att ladda ner förtränade modeller, tokenisering samt dataset.

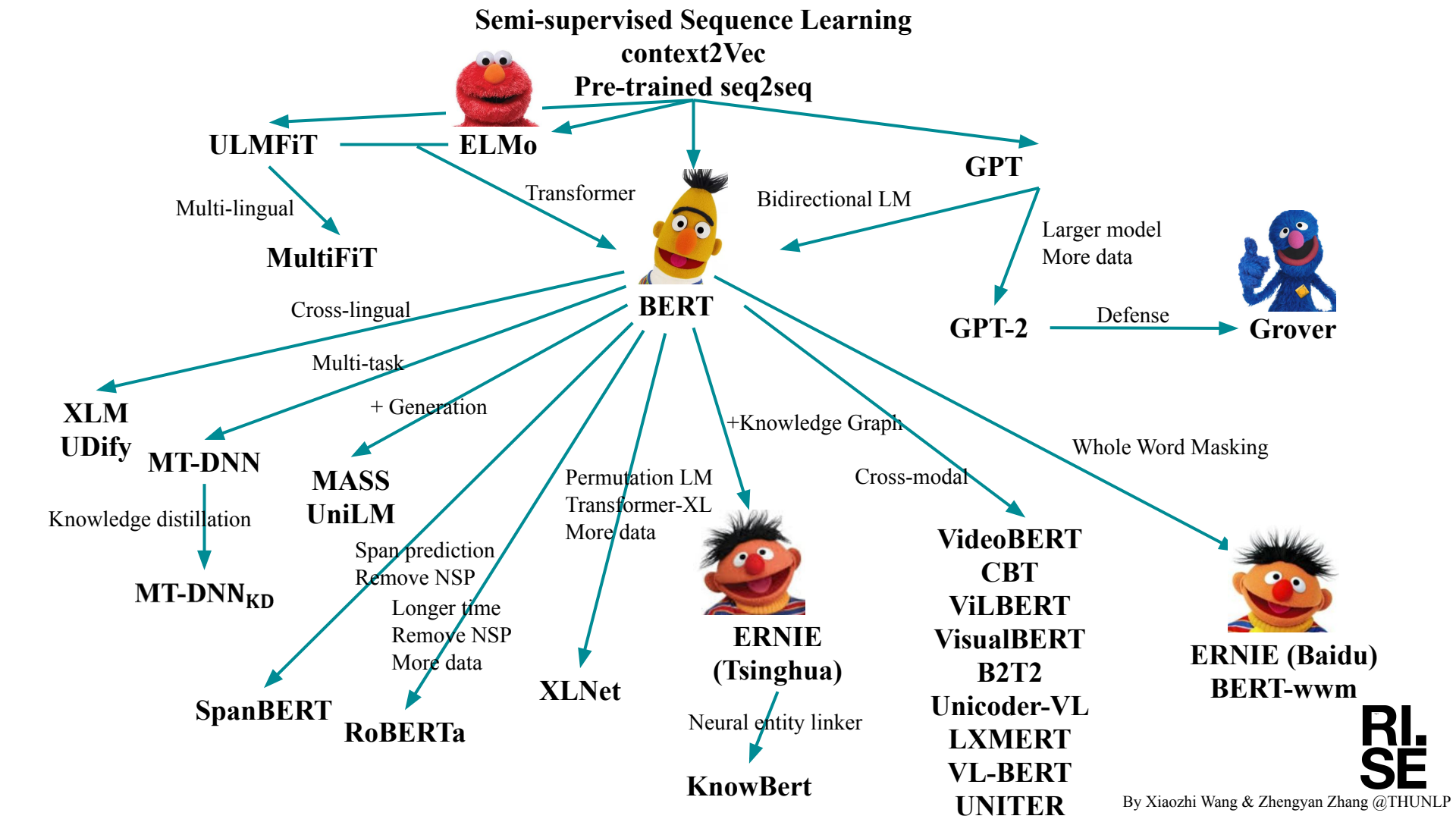
Om man vill applicera språkmodeller snabbt och smidigt är huggingface ett bra alternativ. https://huggingface.co/

Vilken språkmodell ska man välja?

Välj en modell någon annan tränat. Större och bättre språkmodeller kommer komma.

Det viktiga är att det finns finetuning data!

- Finns textmängder tillgängliga som kan användas?
- Finns det processer som skapar annoterad, eller kopplad text?


Semi-supervised Sequence Learning
context2Vec
Pre-trained seq2seq
ULMFiT
ELMo
GPT
Multi-lingual
MultiFiT
Transformer
Bidirectional LM
Larger model
More data
BERT
GPT-2
Defense
Grover
Cross-lingual
XLM
UDify
Multi-task
MT-DNN
Knowledge distillation
MT-DNN$_{KD}$
+ Generation
MASS
UniLM
Span prediction
Remove NSP
SpanBERT
Longer time
Remove NSP
More data
RoBERTa
Permutation LM
Transformer-XL
More data
XLNet
+Knowledge Graph
ERNIE
(Tsinghua)
Neural entity linker
KnowBert
Cross-modal
VideoBERT
CBT
ViLBERT
VisualBERT
B2T2
Unicoder-VL
LXMERT
VL-BERT
UNITER
Whole Word Masking
ERNIE (Baidu)
BERT-wwm
By Xiaozhi Wang & Zhengyan Zhang @THUNLP

Kurser och läromaterial

Allmänna pytorch tutorials
https://pytorch.org/tutorials/

En numera klassisk online-kurs i maskininlärning
https://www.coursera.org/learn/machine-learning

En kurs i NLP med huggingface biblioteken (mycket användbar)
https://huggingface.co/course/chapter1

Djupdykning i Transformers

https://arxiv.org/abs/1706.03762

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

https://github.com/huggingface/transformers

Verktyg

https://git-scm.com/

RI.
SE

https://www.docker.com/

RI.
SE

https://docs.conda.io/

RI.
SE

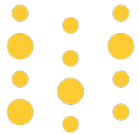
https://pytorch.org/

RI.
SE

HUGGING FACE

https://huggingface.co/

RI.
SE



Weights & Biases

https://wandb.ai/

RI.
SE

vi - terminal text editor