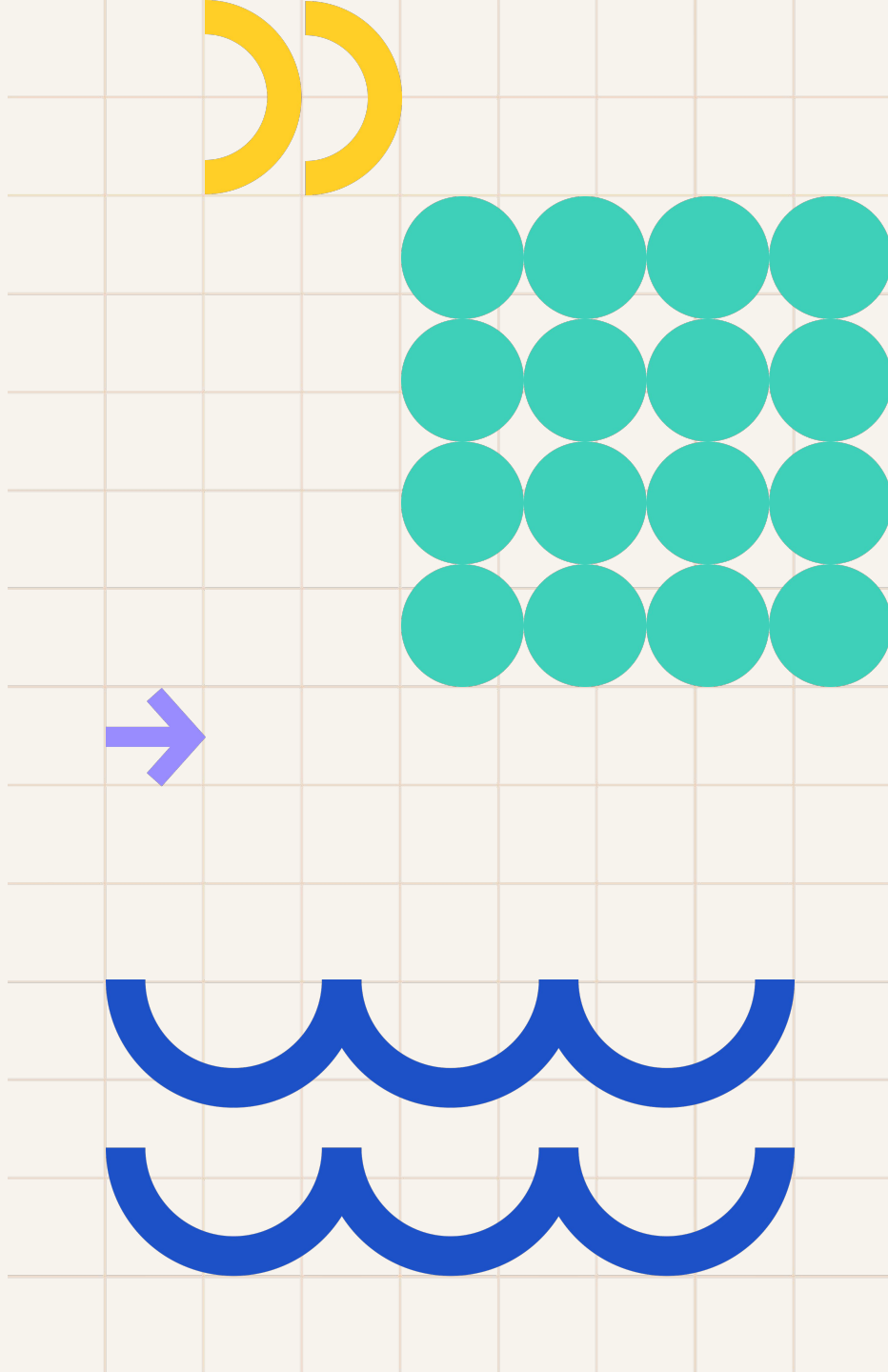
AIRI × Университет Сириус

Pretrained Language Models

Alsu Sagirova



Agenda

01 Transformer recap

02 Word Embeddings

03 Pretrained models

Telegram

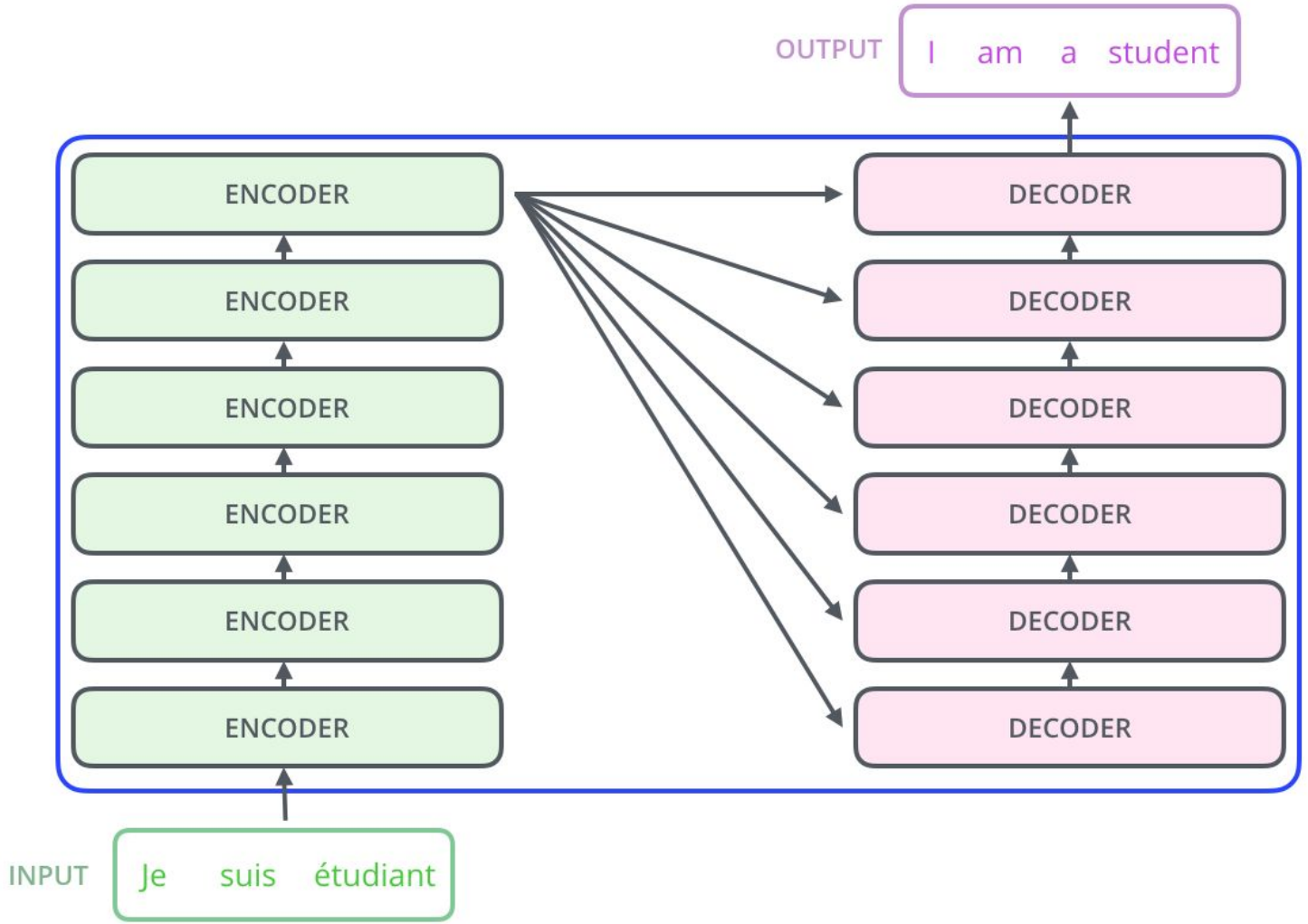
NLP course github

Feedback

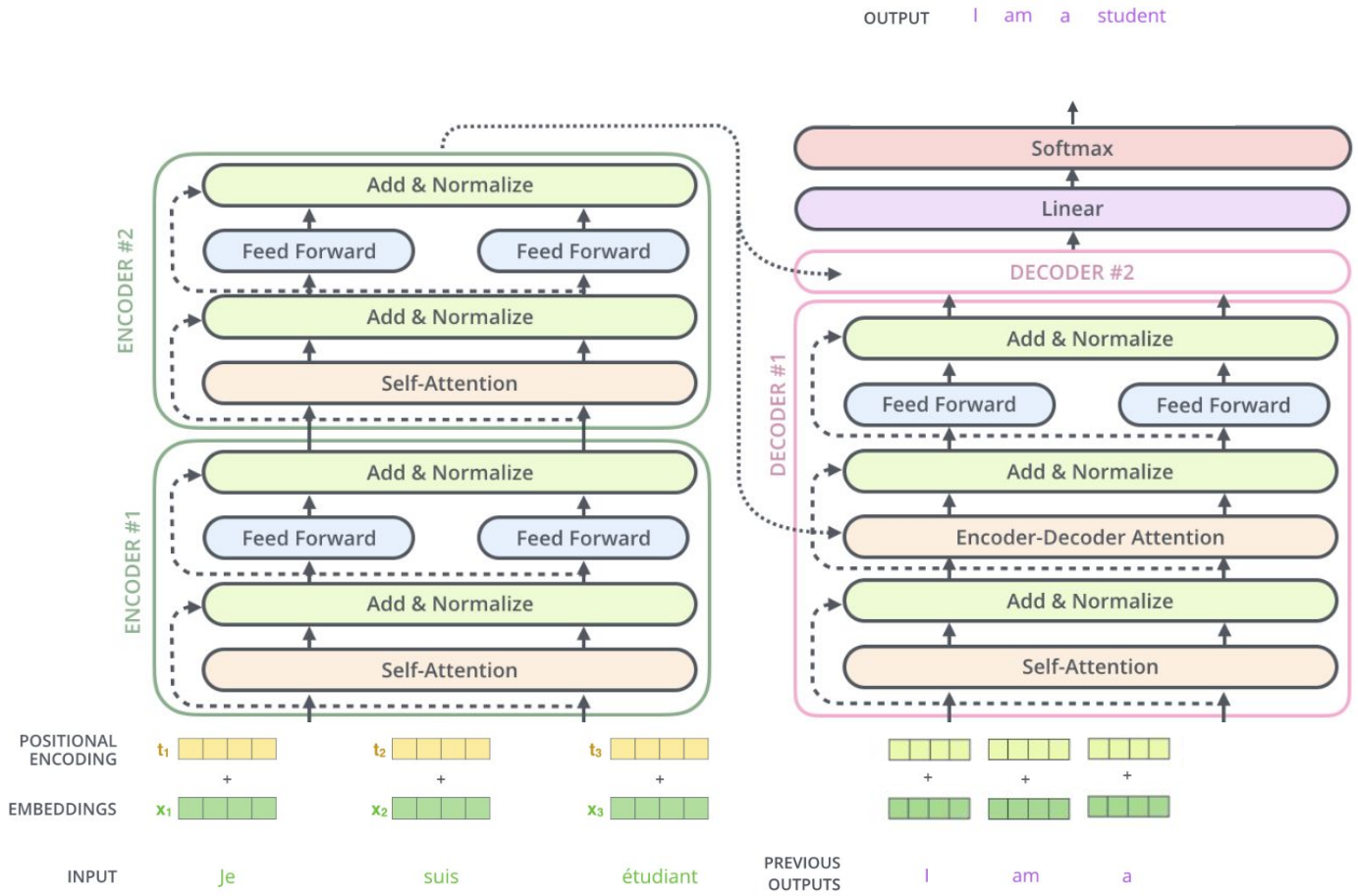
01

Transformer recap

Transformer (recap)



Transformer (recap)



AIRI × Университет Сириус

02

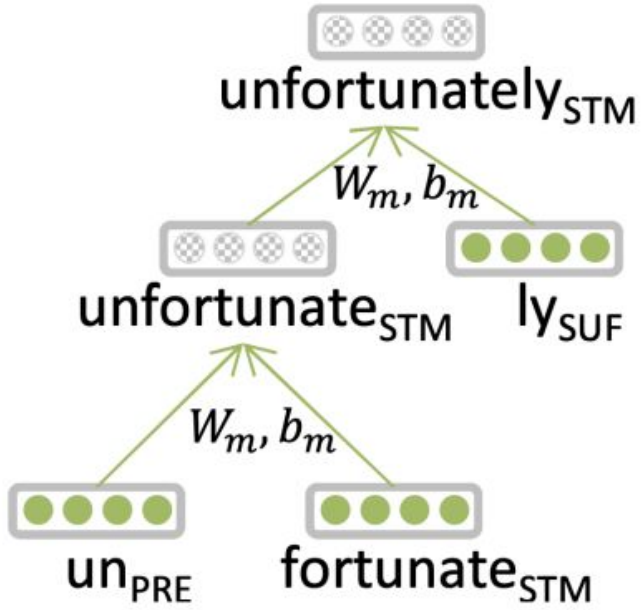
Word Embeddings

Word structure

Previously: language model operated with a large fixed vocabulary built from the training set → all *novel* words seen at test time would be mapped to a single UNK token

- Many languages exhibit complex **morphology**
- ***Subword modeling*** represents every word as a bag of character n-grams (subword tokens)
- At training and testing time, each word is split into a sequence of known subwords

hello → #he, hel, ell, llo, lo#



The byte-pair encoding algorithm (BPE)*

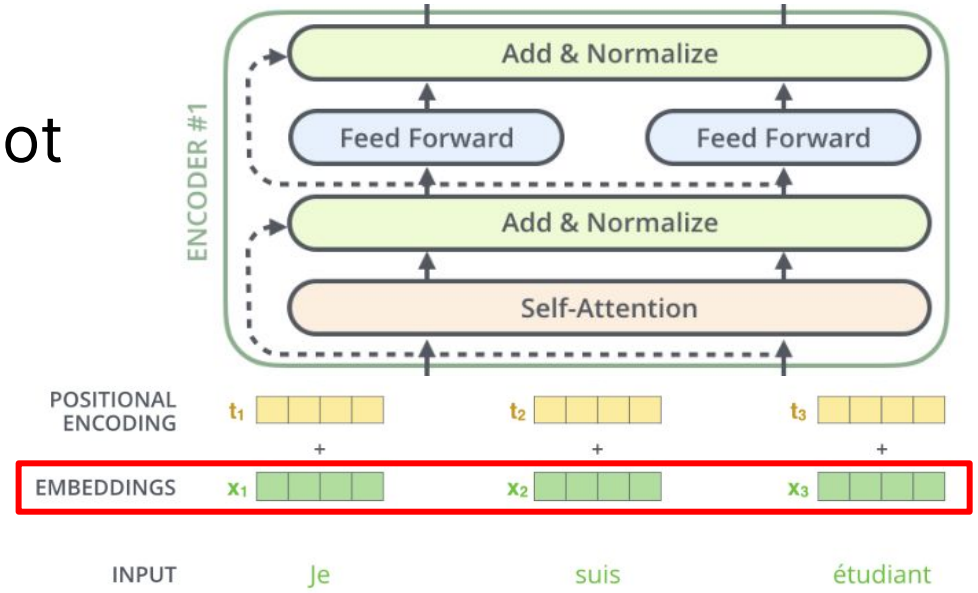
1. Start with a vocabulary containing only characters and an “end-of-word” symbol.
2. Using a corpus of text, find the most common pair of adjacent characters “r, e”; add subword “re” to the vocab.
3. Replace instances of the character pair with the new subword; repeat until desired vocab size.

Originally BPE was used in NLP for machine translation.
Now a similar method called WordPiece is used in pretrained models.

*Rico Sennrich, Barry Haddow, and Alexandra Birch. Neural Machine Translation of Rare Words with Subword Units, 2016

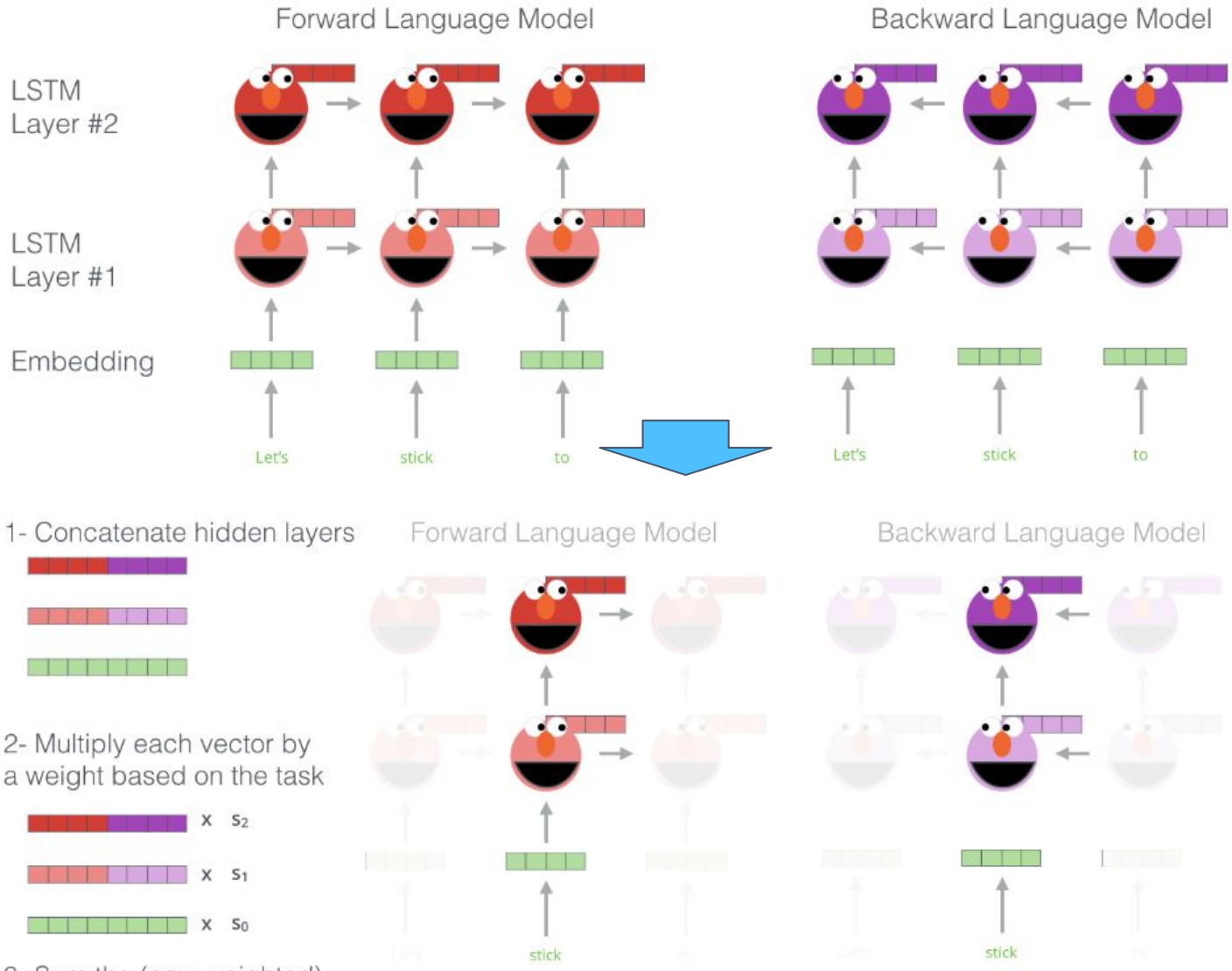
Word Embeddings

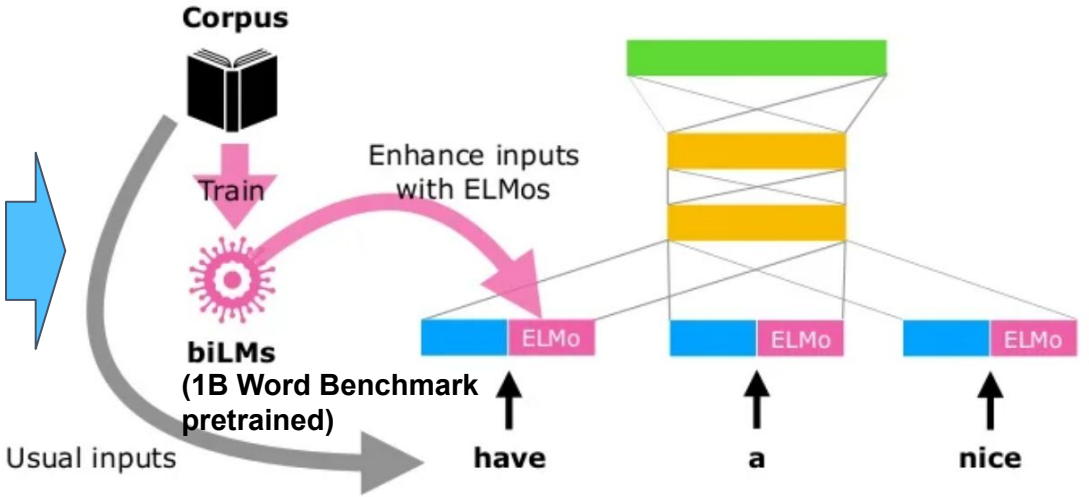
- trained from scratch as a part of your model
  - *pro:* task-specific domain data
  - *cons:* labeled training corpus might be not huge/diverse enough to learn relationships between words
- static pretrained (Word2Vec, GloVe)
  - *pro:* know semantics and syntax
  - *con:* embeddings are not task-specific and **not conditioned on context**



Word Embeddings → Contextualized Word-Embeddings

ELMo (Deep contextualized word representations)





ELMo (Deep contextualized word representations)

	Source	Nearest Neighbors
GloVe	play	playing, game, games, played, players, plays, player, Play, football, multiplayer
biLM	Chico Ruiz made a spectacular play on Alusik 's grounder {...}	Kieffer , the only junior in the group , was commended for his ability to hit in the clutch , as well as his all-round excellent play .
	Olivia De Havilland signed to do a Broadway play for Garson {...}	{...} they were actors who had been handed fat roles in a successful play , and had talent enough to fill the roles competently , with nice understatement .

Table 4: Nearest neighbors to "play" using GloVe and the context embeddings from a biLM.

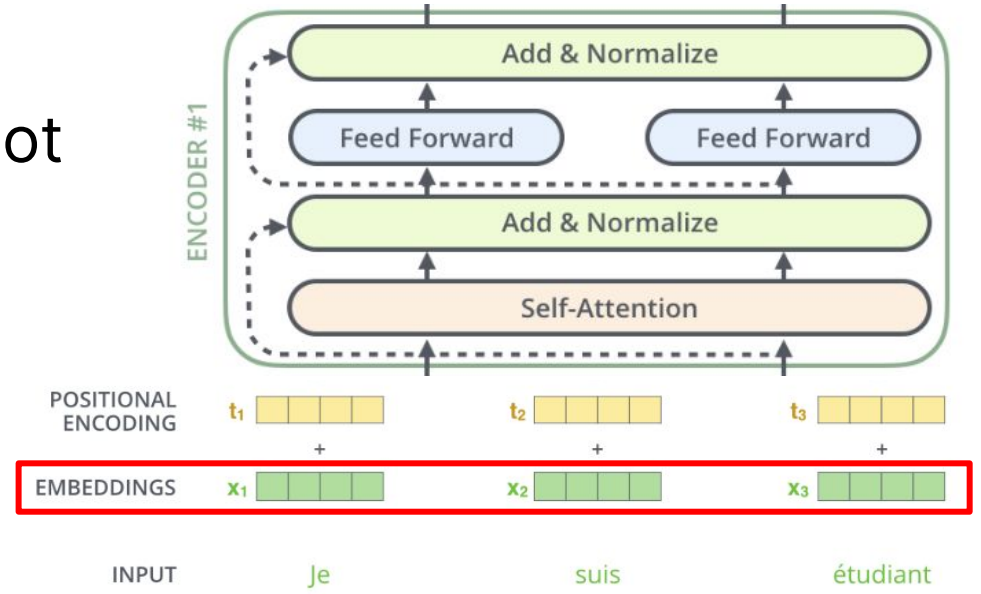
ELMo (Deep contextualized word representations)

Task	Previous SOTA		Our baseline	ELMo + baseline	Increase (absolute/ relative)
SQuAD	Liu et al. (2017)	84.4	81.1	85.8	4.7 / 24.9%
SNLI	Chen et al. (2017)	88.6	88.0	88.7 ± 0.17	0.7 / 5.8%
SRL	He et al. (2017)	81.7	81.4	84.6	3.2 / 17.2%
Coref	Lee et al. (2017)	67.2	67.2	70.4	3.2 / 9.8%
NER	Peters et al. (2017)	91.93 ± 0.19	90.15	92.22 ± 0.10	2.06 / 21%
SST-5	McCann et al. (2017)	53.7	51.4	54.7 ± 0.5	3.3 / 6.8%

Table 1: Test set comparison of ELMo enhanced neural models with state-of-the-art single model baselines across six benchmark NLP tasks. The performance metric varies across tasks – accuracy for SNLI and SST-5; F_1 for SQuAD, SRL and NER; average F_1 for Coref. Due to the small test sizes for NER and SST-5, we report the mean and standard deviation across five runs with different random seeds. The "increase" column lists both the absolute and relative improvements over our baseline.

Word Embeddings

- trained from scratch as a part of your model
  - *pro:* task-specific domain data
  - *cons:* labeled training corpus might be not huge/diverse enough to learn relationships between words
- static pretrained (Word2Vec, GloVe)
  - *pro:* know relationships between words as trained on large corpus
  - *con:* embeddings are not task-specific and not conditioned on context
- pretrained embeddings, **fine-tuned to adapt to the task**



"transferring" the knowledge of embeddings training data to the task-specific model

03

Pretrained models

Pretrained models

Through **the pretrained model**, we "transfer" the knowledge of its training data to our task-specific model.

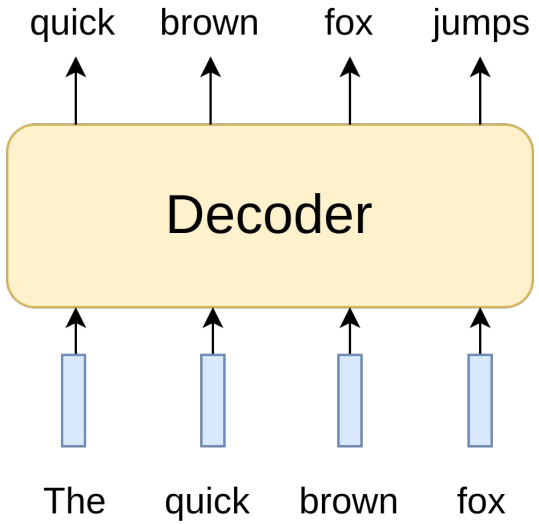
Advantages:

- strong representations of language
- strong parameter initializations for strong NLP models
- strong probability distributions over language that we can sample from

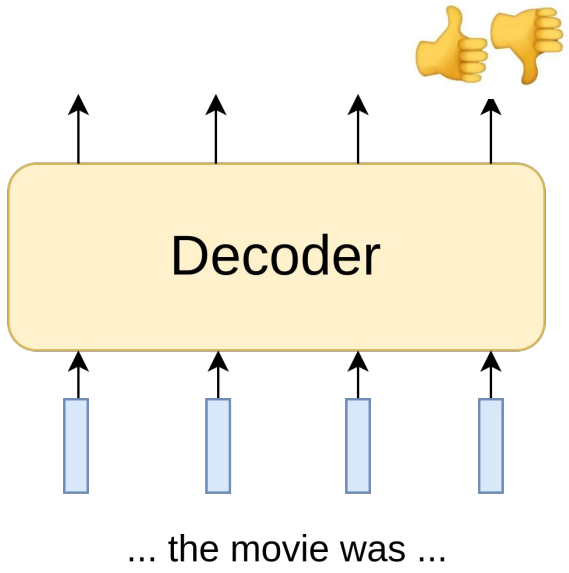
Language modeling for pretraining

In 2015, Andrew M. Dai and Quoc V. Le* from Google presented the idea of pretraining through language modeling followed by the task-specific fine-tuning.

- **Pretrain** a neural network to perform language modeling on a *large* amount of text
- Save the network parameters



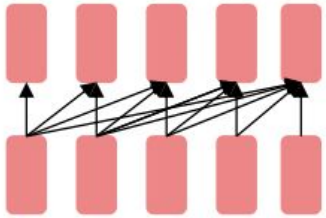
- **Fine-tune** on your task
- *Not many* labels, adapting to the task!



* Semi-supervised Sequence Learning, 2015

PLM architectures

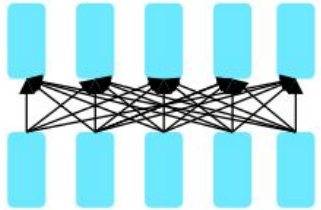
LMs used for sequence generation that do not condition on future



Decoders

(GPT-2, GPT-3, LaMDA)

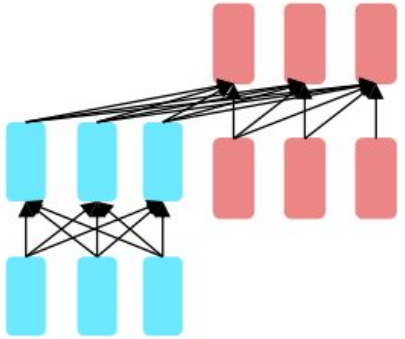
Bidirectional context architectures



Encoders

(BERT, RoBERTa etc.)

Combined architectures



Encoder-Decoders

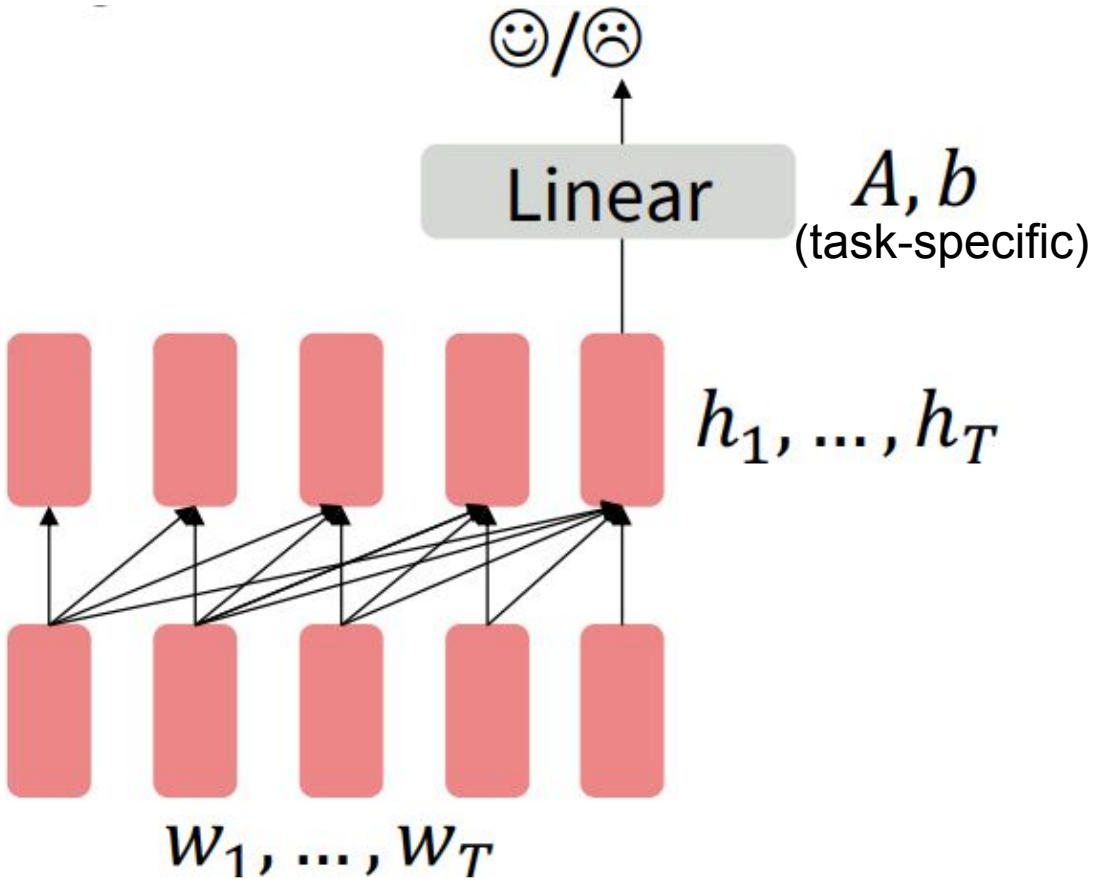
(Transformer, Meena, T5)

AIRI × Университет Сириус

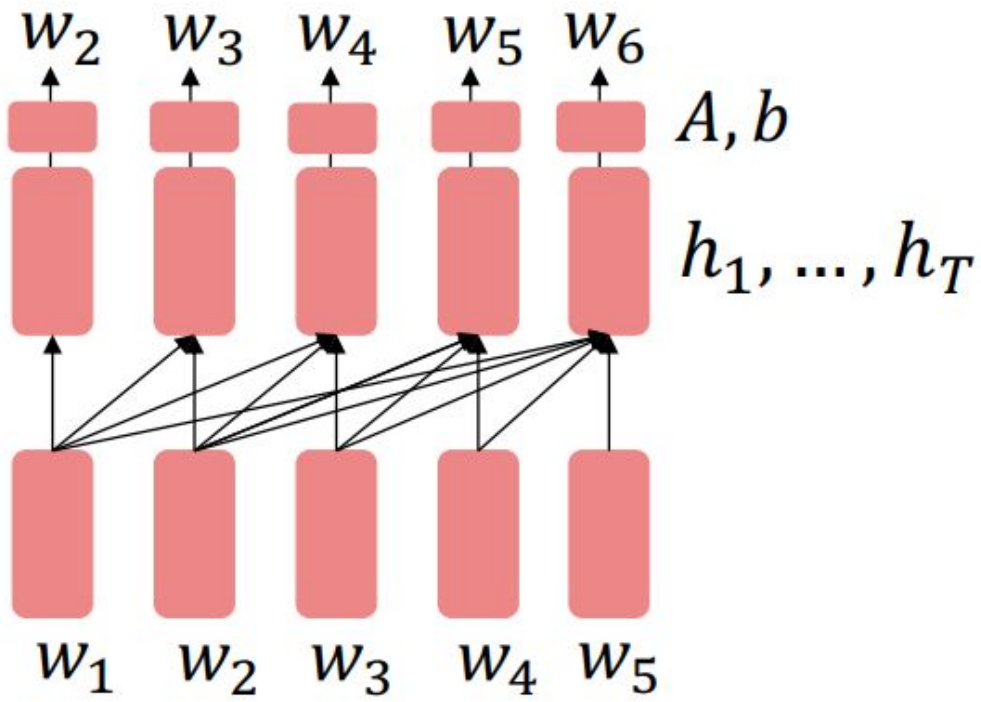
Decoders

Decoder-only models

Classification



Seq2seq



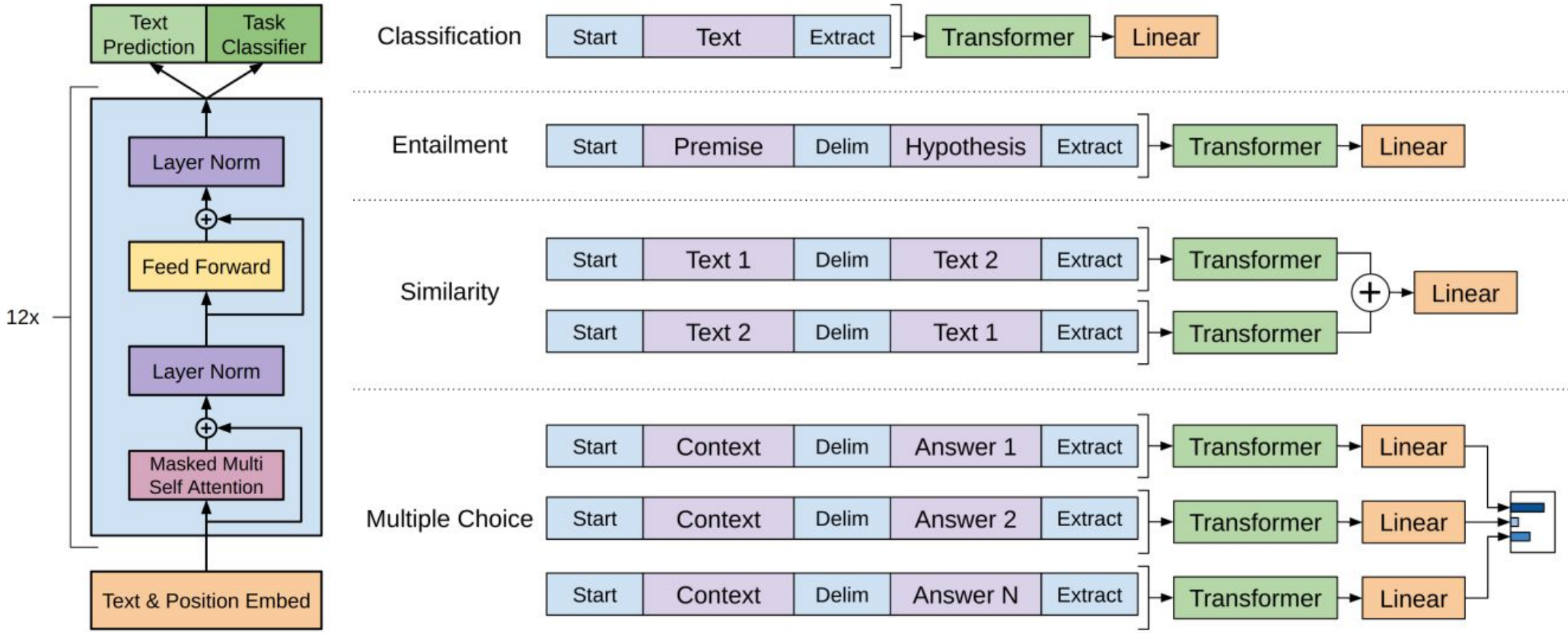
Decoder-only models

EXAMPLE: **Generative Pretrained Transformer (GPT)***:

- 12-layer Transformer decoder (without cross-attention)
- 768-dimensional hidden states
- 3072-dimensional feed-forward hidden layers
- BPE with 40,000 merges
- Trained on BooksCorpus
  (7000 unique books, long spans of contiguous text for learning long-distance dependencies)

* Radford et al., «Improving Language Understanding by Generative Pre-Training», 2018

GPT (OpenAI Transformer)



The [EXTRACT] token representation goes to the linear classifier layer.

GPT

Method	MNLI-m	MNLI-mm	SNLI	SciTail	QNLI	RTE
ESIM + ELMo [44] (5x)	-	-	89.3	-	-	-
CAFE [58] (5x)	80.2	79.0	89.3	-	-	-
Stochastic Answer Network [35] (3x)	80.6	80.1	-	-	-	-
CAFE [58]	78.7	77.9	88.5	83.3		
GenSen [64]	71.4	71.3	-	-	82.3	59.2
Multi-task BiLSTM + Attn [64]	72.2	72.1	-	-	82.1	**61.7**
Finetuned Transformer LM (ours)	**82.1**	**81.4**	**89.9**	**88.3**	**88.1**	56.0

Table 2: Experimental results on natural language inference tasks, comparing our model with current state-of-the-art methods. 5x indicates an ensemble of 5 models. All datasets use accuracy as the evaluation metric.

GPT

Method	Story Cloze	RACE-m	RACE-h	RACE
val-LS-skip [55]	76.5	-	-	-
Hidden Coherence Model [7]	77.6	-	-	-
Dynamic Fusion Net [67] (9x)	-	55.6	49.4	51.2
BiAttention MRU [59] (9x)	-	60.2	50.3	53.3
Finetuned Transformer LM (ours)	**86.5**	**62.9**	**57.4**	**59.0**

Table 3: Results on question answering and commonsense reasoning, comparing our model with current state-of-the-art methods.. 9x means an ensemble of 9 models.

GPT

Method	Classification		Semantic Similarity			GLUE
	CoLA (mc)	SST2 (acc)	MRPC (F1)	STSB (pc)	QQP (F1)	
Sparse byte mLSTM [16]	-	**93.2**	-	-	-	-
TF-KLD [23]	-	-	**86.0**	-	-	-
ECNU (mixed ensemble) [60]	-	-	-	81.0	-	-
Single-task BiLSTM + ELMo + Attn [64]	35.0	90.2	80.2	55.5	66.1	64.8
Multi-task BiLSTM + ELMo + Attn [64]	18.9	91.6	83.5	72.8	63.3	68.9
Finetuned Transformer LM (ours)	**45.4**	91.3	82.3	**82.0**	**70.3**	**72.8**

Table 4: Semantic similarity and classification results, comparing our model with current state-of-the-art methods. All task evaluations in this table were done using the GLUE benchmark. (*mc*= Mathews correlation, *acc*=Accuracy, *pc*=Pearson correlation)

GPT

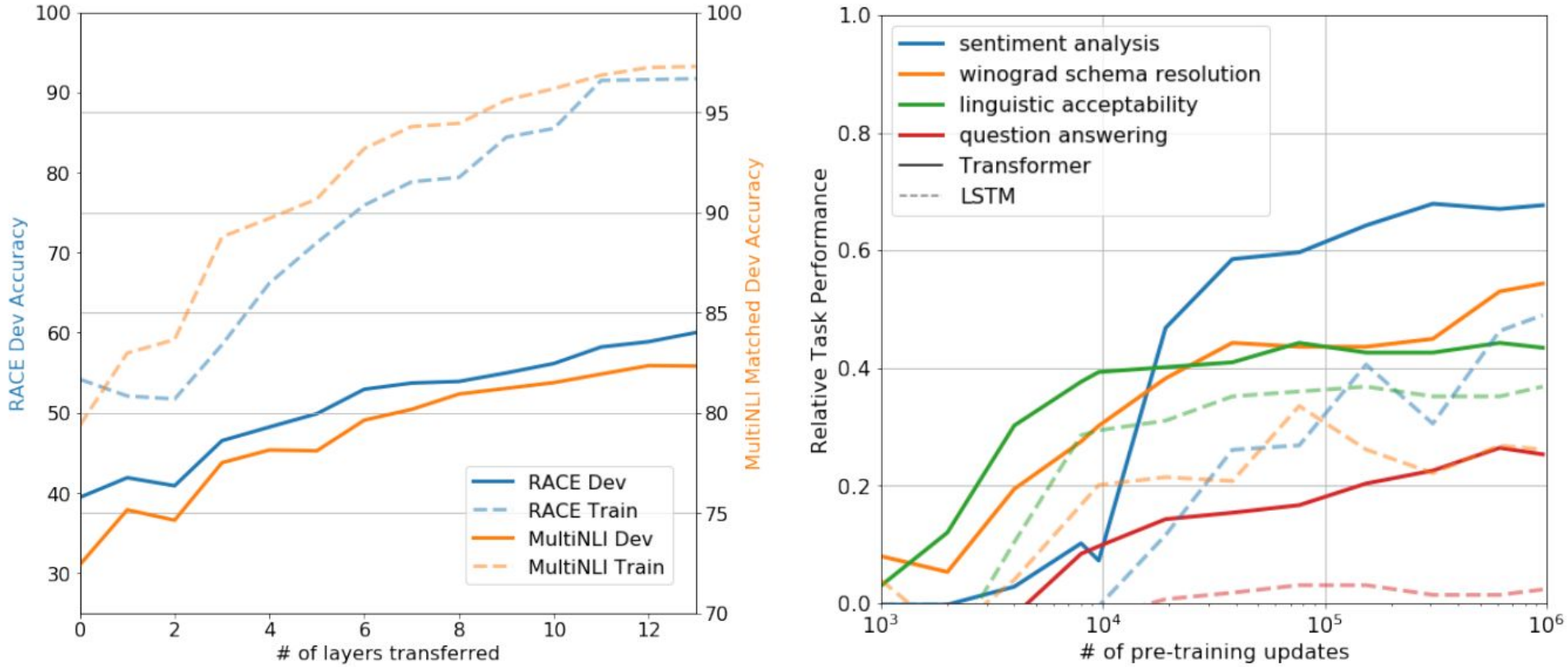


Figure 2: (**left**) Effect of transferring increasing number of layers from the pre-trained language model on RACE and MultiNLI. (**right**) Plot showing the evolution of zero-shot performance on different tasks as a function of LM pre-training updates. Performance per task is normalized between a random guess baseline and the current state-of-the-art with a single model.

GPT-2 (Language Models are Unsupervised Multitask Learners)

- LM fine-tuning
- 12-48 Transformer decoder layers
- WebText dataset (texts from 45M web pages)

Language Models are Unsupervised Multitask Learners

	LAMBADA (PPL)	LAMBADA (ACC)	CBT-CN (ACC)	CBT-NE (ACC)	WikiText2 (PPL)	PTB (PPL)	enwik8 (BPB)	text8 (BPC)	WikiText103 (PPL)	1BW (PPL)
SOTA	99.8	59.23	85.7	82.3	39.14	46.54	0.99	1.08	18.3	**21.8**
117M	**35.13**	45.99	**87.65**	**83.4**	**29.41**	65.85	1.16	1.17	37.50	75.20
345M	**15.60**	55.48	**92.35**	**87.1**	**22.76**	47.33	1.01	**1.06**	26.37	55.72
762M	**10.87**	**60.12**	**93.45**	**88.0**	**19.93**	**40.31**	**0.97**	**1.02**	22.05	44.575
1542M	**8.63**	**63.24**	**93.30**	**89.05**	**18.34**	**35.76**	**0.93**	**0.98**	**17.48**	42.16

Table 3. Zero-shot results on many datasets. No training or fine-tuning was performed for any of these results. PTB and WikiText-2 results are from (Gong et al., 2018). CBT results are from (Bajgar et al., 2016). LAMBADA accuracy result is from (Hoang et al., 2018) and LAMBADA perplexity result is from (Grave et al., 2016). Other results are from (Dai et al., 2019).

GPT-2 (Language Models are Unsupervised Multitask Learners)

Context (human-written): In a shocking finding, scientist discovered a herd of unicorns living in a remote, previously unexplored valley, in the Andes Mountains. Even more surprising to the researchers was the fact that the unicorns spoke perfect English.

GPT-2: The scientist named the population, after their distinctive horn, Ovid's Unicorn. These four-horned, silver-white unicorns were previously unknown to science.

Now, after almost two centuries, the mystery of what sparked this odd phenomenon is finally solved.

Dr. Jorge Pérez, an evolutionary biologist from the University of La Paz, and several companions, were exploring the Andes Mountains when they found a small valley, with no other animals or humans. Pérez noticed that the valley had what appeared to be a natural fountain, surrounded by two peaks of rock and silver snow.

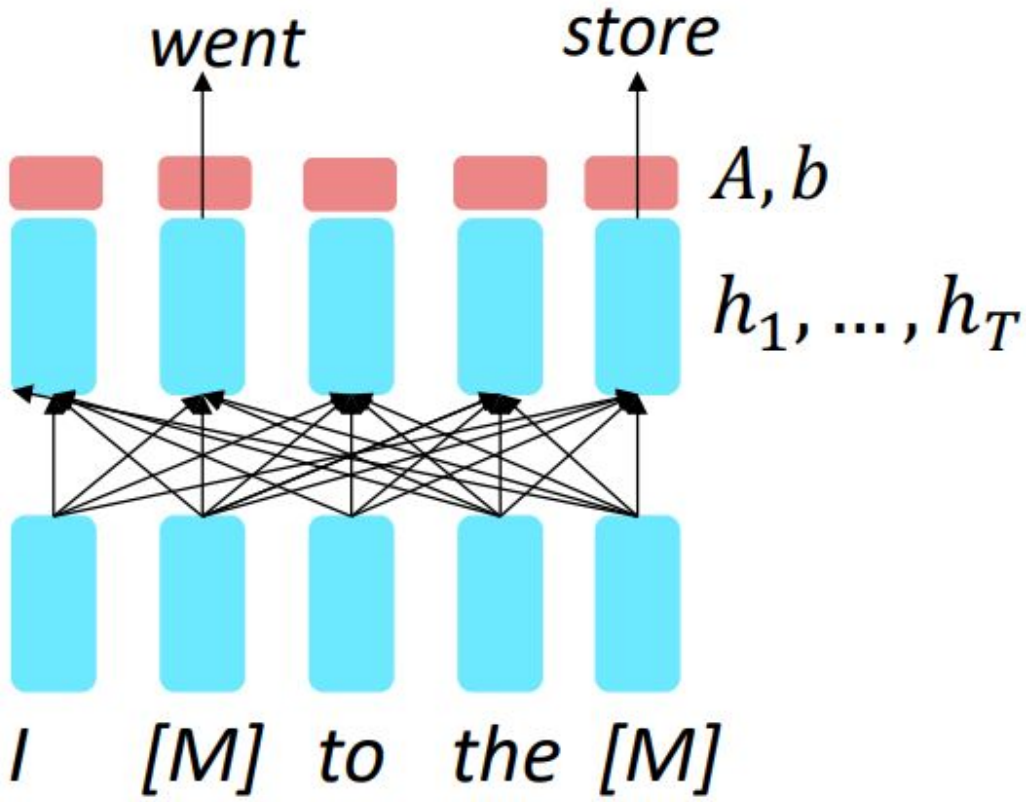
Pérez and the others then ventured further into the valley. "By the time we reached the top of one peak, the water looked blue, with some crystals on top," said Pérez.

Pérez and his friends were astonished to see the unicorn herd. These creatures could be seen from the air without having to move too much to see them – they were so close they could touch their horns.

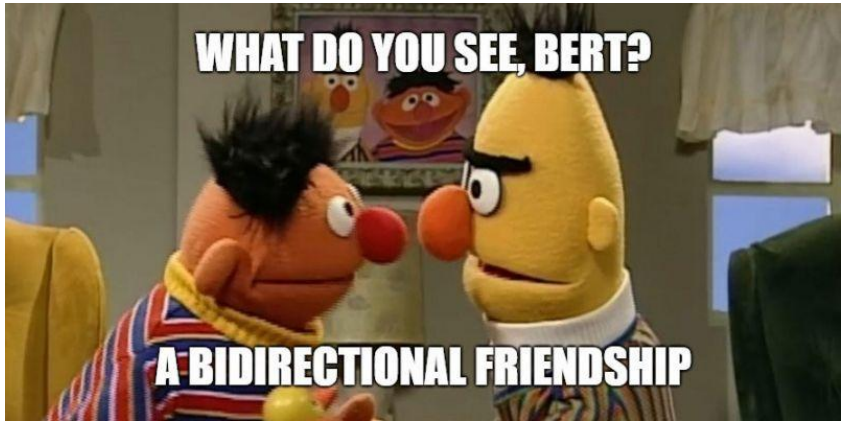
Encoders

Pretrained encoder-only models

- Bidirectional access
- LM pretraining impossible!
- Solution:
  Masked Language Modeling
- Loss is calculated for masked tokens only



EXAMPLE: BERT (Bidirectional Encoder Representations from Transformers)



AIRI × Университет Сириус

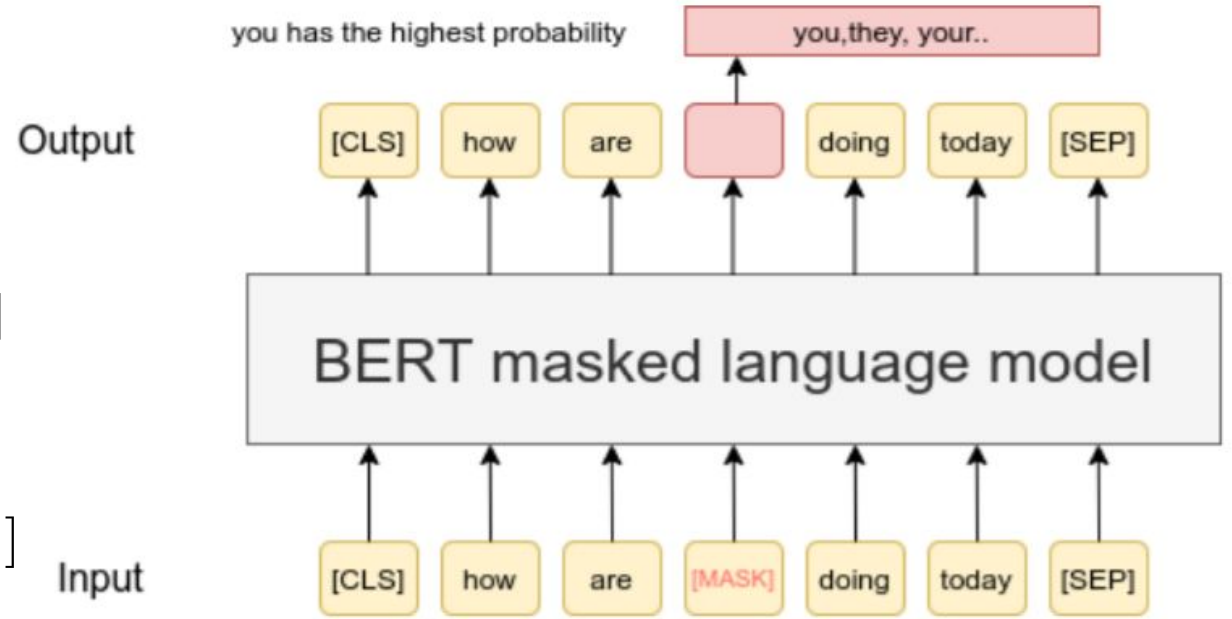
BERT: details

- Released models
  - BERT-base: 12 layers, 768-dim hidden states, 12 attention heads, 110 million params
  - BERT-large: 24 layers, 1024-dim hidden states, 16 attention heads, 340 million params
- Datasets: BooksCorpus (800M words) and English Wikipedia (2,500M words)
- Pretraining done with 64 TPU chips for a total of 4 days
- Finetuning on a single GPU

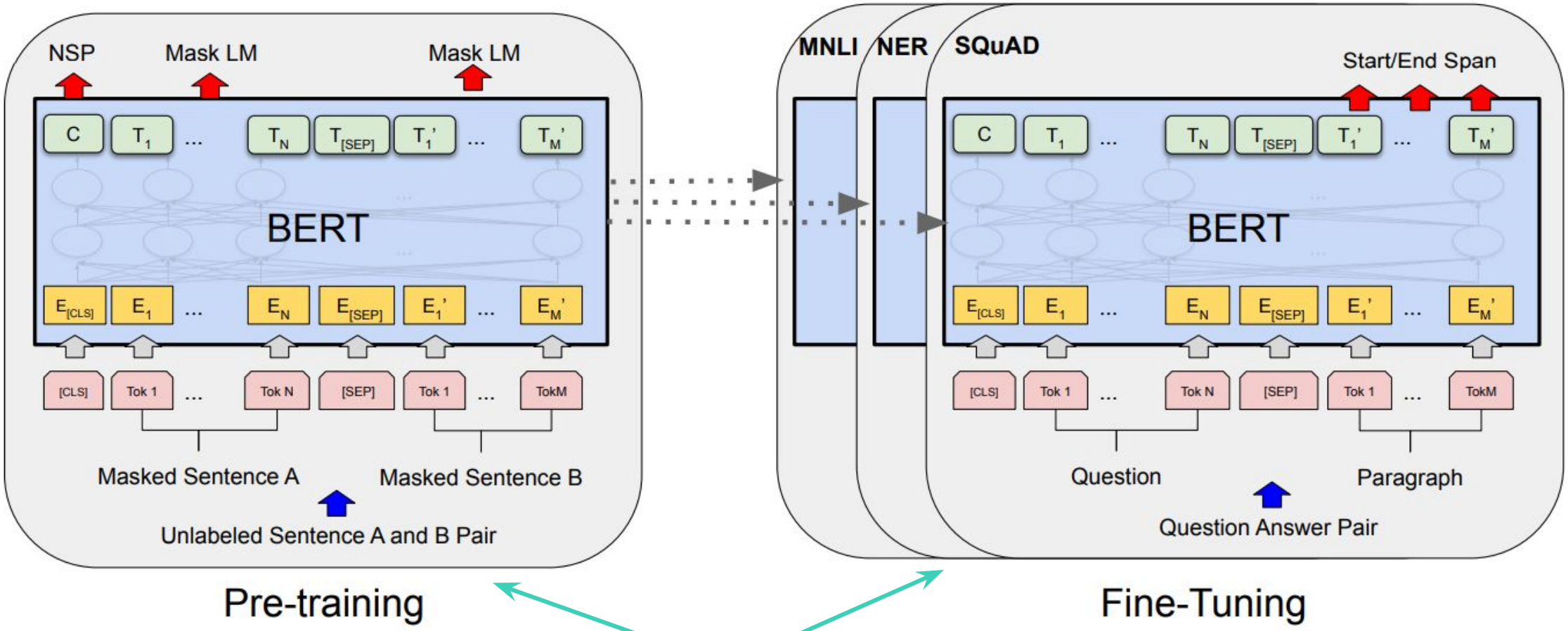
MLM pretraining for BERT

How BERT was pretrained:

- A random 15% of input tokens are masked
- [MASK] token does not appear in fine-tuning → input tokens are replaced with
  - 80%: replace with [MASK]
    `my dog is hairy -> my dog is [MASK]`
  - 10%: replace with random token
    `my dog is hairy -> my dog is apple`
  - 10%: leave token unchanged
    `my dog is hairy -> my dog is hairy`



BERT



- Unified architecture for pretraining & fine-tuning
- The same pretrained model weights for different downstream tasks
- During fine-tuning, all parameters are fine-tuned

AIRI × Университет Сириус

BERT

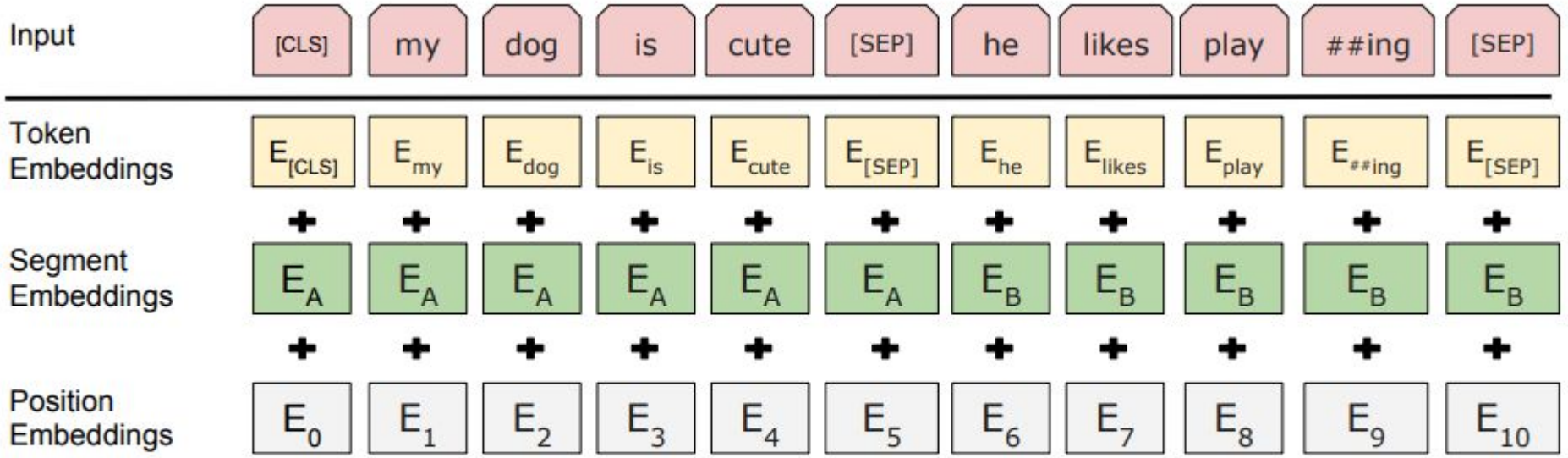


Figure 2: BERT input representation. The input embeddings are the sum of the token embeddings, the segmentation embeddings and the position embeddings.

NB *In addition to MLM pretraining task, authors test Next Sentence Prediction pretraining (check if the second sentence in input is actually following the first sentence).*

BERT

System	MNLI-(m/mm) 392k	QQP 363k	QNLI 108k	SST-2 67k	CoLA 8.5k	STS-B 5.7k	MRPC 3.5k	RTE 2.5k	**Average** -
Pre-OpenAI SOTA	80.6/80.1	66.1	82.3	93.2	35.0	81.0	86.0	61.7	74.0
BiLSTM+ELMo+Attn	76.4/76.1	64.8	79.8	90.4	36.0	73.3	84.9	56.8	71.0
OpenAI GPT	82.1/81.4	70.3	87.4	91.3	45.4	80.0	82.3	56.0	75.1
$\text{BERT}_{\text{BASE}}$	84.6/83.4	71.2	90.5	93.5	52.1	85.8	88.9	66.4	79.6
$\text{BERT}_{\text{LARGE}}$	**86.7/85.9**	**72.1**	**92.7**	**94.9**	**60.5**	**86.5**	**89.3**	**70.1**	**82.1**

Table 1: GLUE Test results, scored by the evaluation server (https://gluebenchmark.com/leaderboard). The number below each task denotes the number of training examples. The "Average" column is slightly different than the official GLUE score, since we exclude the problematic WNLI set.[8] BERT and OpenAI GPT are single-model, single task. F1 scores are reported for QQP and MRPC, Spearman correlations are reported for STS-B, and accuracy scores are reported for the other tasks. We exclude entries that use BERT as one of their components.

Major BERT improvements

- RoBERTa:
  - more train data
  - bigger batches
  - longer inputs

  } *improved pretraining without architecture change*

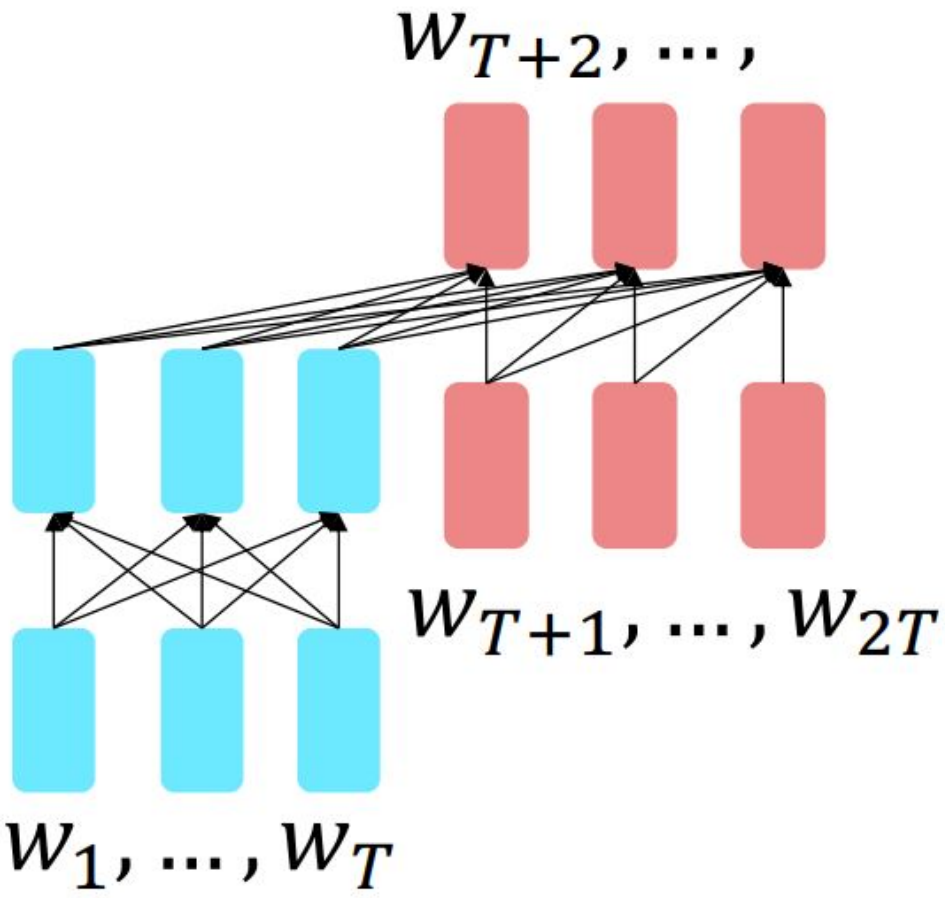
  - remove next sentence prediction!
- SpanBERT
  - pre-training method for better span representation and prediction
  - masking contiguous spans of words makes a harder, more useful pretraining task

Encoder-decoders

Encoder-decoders pretraining

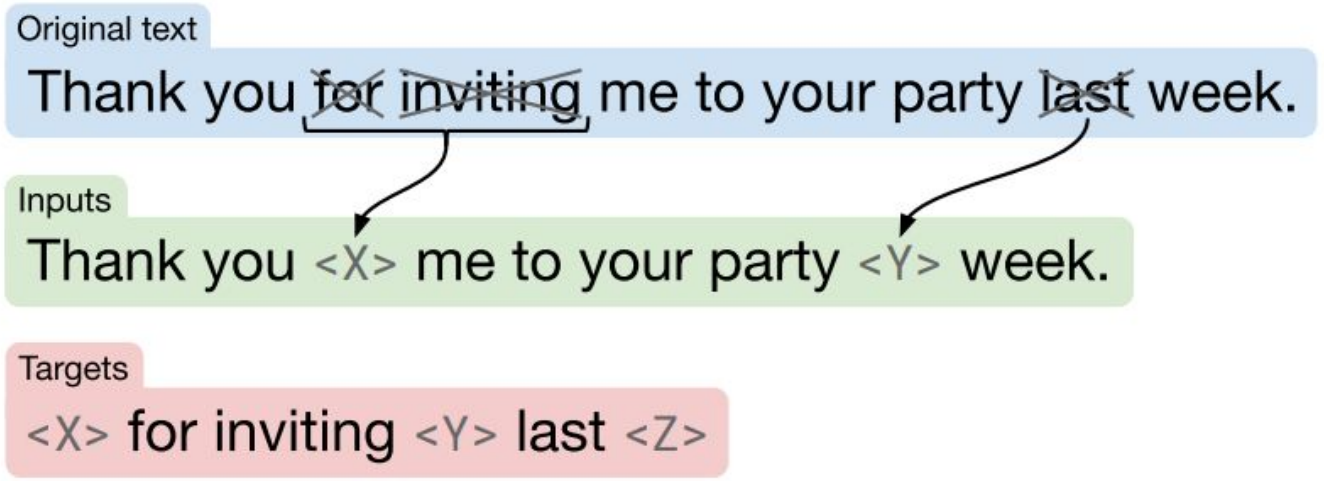
- Decoder: language modeling
- Encoder: language modeling where a prefix of every input is provided to the encoder and is not predicted.

EXAMPLE:
T5 (Text-to-Text Transfer Transformer)

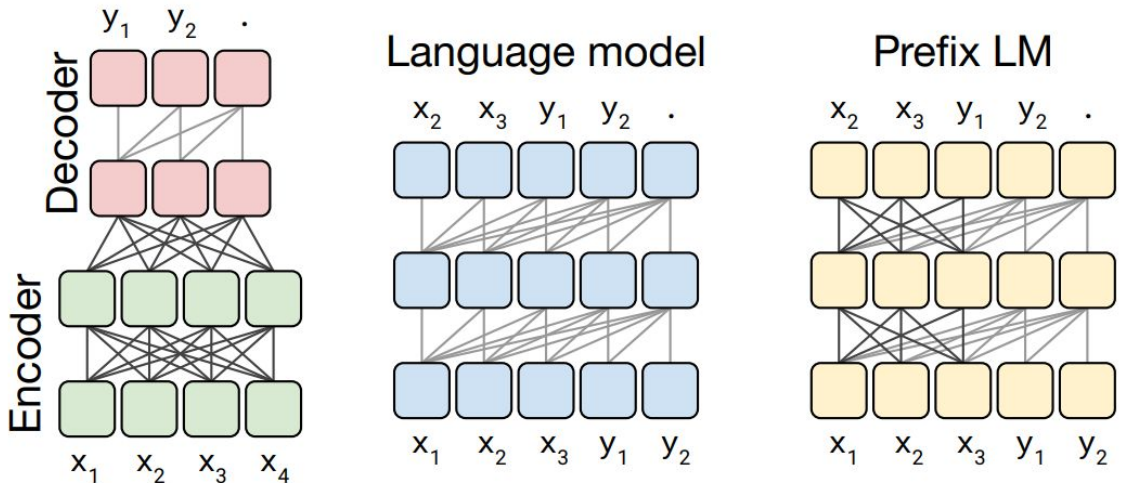


T5 pretraining objective

- Corrupted words are chosen randomly
- Each consecutive span of corrupted tokens is replaced by a sentinel token (<X> and <Y>)unique over the example
- The output sequence: dropped-out spans, delimited by the sentinel tokens used to replace them in the input plus a final sentinel token <Z>.



T5 pretraining objectives exploration



Architecture	Objective	Params	Cost	GLUE	CNNDM	SQuAD	SGLUE	EnDe	EnFr	EnRo
★ Encoder-decoder	Denoising	$2P$	M	**83.28**	**19.24**	**80.88**	**71.36**	**26.98**	**39.82**	**27.65**
Enc-dec, shared	Denoising	P	M	82.81	18.78	**80.63**	**70.73**	26.72	39.03	**27.46**
Enc-dec, 6 layers	Denoising	P	$M/2$	80.88	18.97	77.59	68.42	26.38	38.40	26.95
Language model	Denoising	P	M	74.70	17.93	61.14	55.02	25.09	35.28	25.86
Prefix LM	Denoising	P	M	81.82	18.61	78.94	68.11	26.43	37.98	27.39
Encoder-decoder	LM	$2P$	M	79.56	18.59	76.02	64.29	26.27	39.17	26.86
Enc-dec, shared	LM	P	M	79.60	18.13	76.35	63.50	26.62	39.17	27.05
Enc-dec, 6 layers	LM	P	$M/2$	78.67	18.26	75.32	64.06	26.13	38.42	26.89
Language model	LM	P	M	73.78	17.54	53.81	56.51	25.23	34.31	25.38
Prefix LM	LM	P	M	79.68	17.84	76.87	64.86	26.28	37.51	26.76

Artificial Intelligence Research Institute

airi.net

- airi_research_institute
- AIRI Institute
- AIRI Institute
- AIRI_inst
- artificial-intelligence-research-institute