

Pre-training + Fine-tuning Paradigm, I

LING 574 Deep Learning for NLP

Shane Steinert-Threlkeld

Note on Transformer Architecture

Do Transformer Modifications Transfer Across Implementations and Applications?

Sharan Narang* Hyung Won Chung Yi Tay William Fedus
Thibault Fevry† Michael Matena † Karishma Malkan† Noah Fiedel
Noam Shazeer Zhenzhong Lan† Yanqi Zhou Wei Li
Nan Ding Jake Marcus Adam Roberts Colin Raffel

Google Research

Abstract

The research community has proposed copious modifications to the Transformer architecture since it was introduced over three years ago, relatively few of which have seen widespread adoption. In this paper, we comprehensively evaluate many of these modifications in a shared experimental setting that covers most of the common uses of the Transformer in natural language processing. Surprisingly, we find that most modifications do not meaningfully improve performance. Furthermore, most of the Transformer

will yield equal-or-better performance on any task that the pipeline is applicable to. For example, residual connections in convolutional networks (He et al., 2016) are designed to ideally improve performance on any task where these models are applicable (image classification, semantic segmentation, etc.). In practice, when proposing a new improvement, it is impossible to test it on every applicable downstream task, so researchers must select a few representative tasks to evaluate it on. However, the proposals that are ultimately adopted by the research community and practitioners tend to be those that reliably improve performance across a wide variety of tasks “in

[link](#)

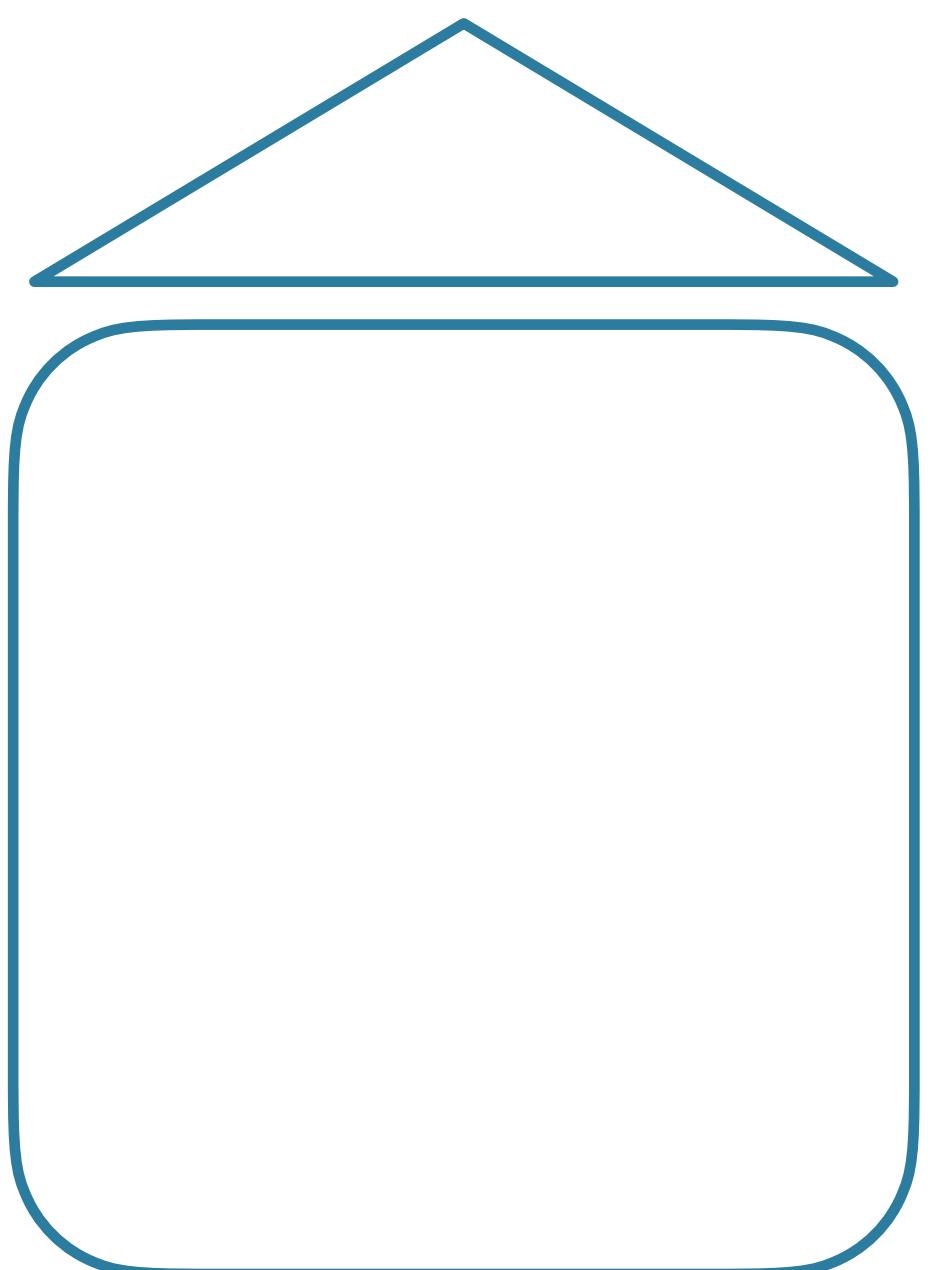
Today's Plan

- Transfer learning in general
- Language model pre-training: initial steps
- Transformer-based pre-training
 - *Encoder only*
- Next time:
 - Decoder only
 - Encoder-Decoder
 - [Some] limitations [more later in course]

Transfer Learning

Standard Learning

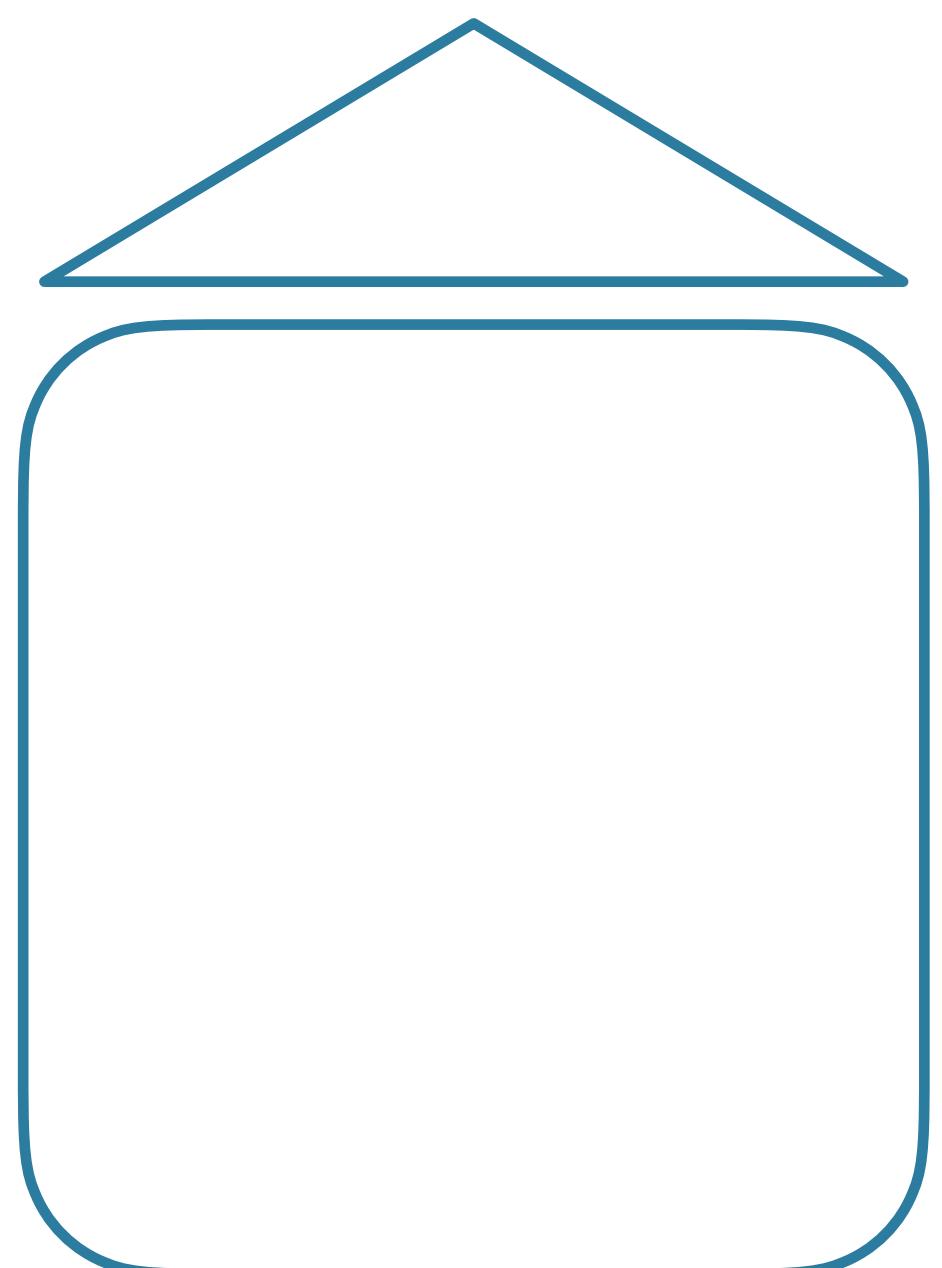
Task I outputs



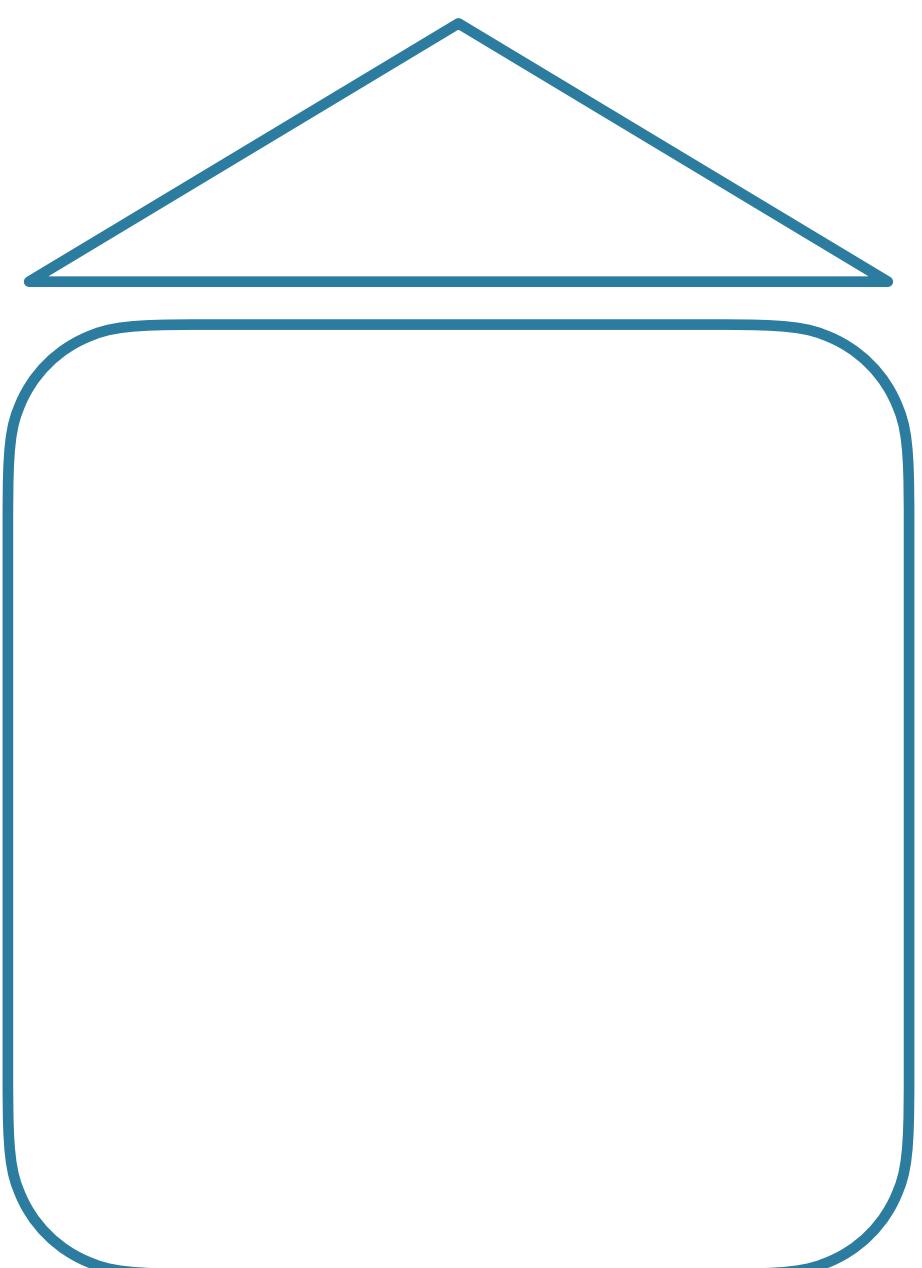
Task I inputs

Standard Learning

Task 1 outputs



Task 2 outputs

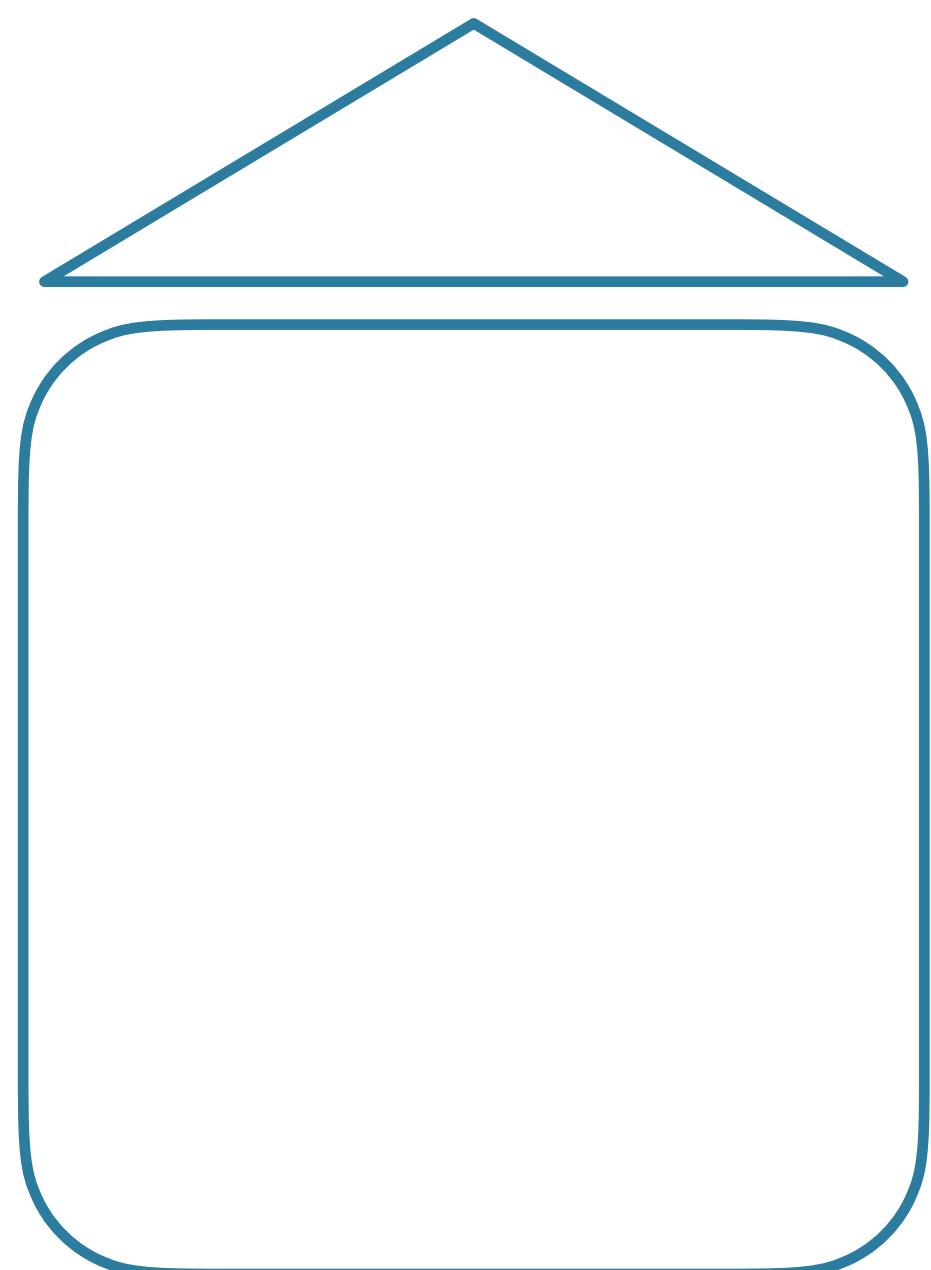


Task 1 inputs

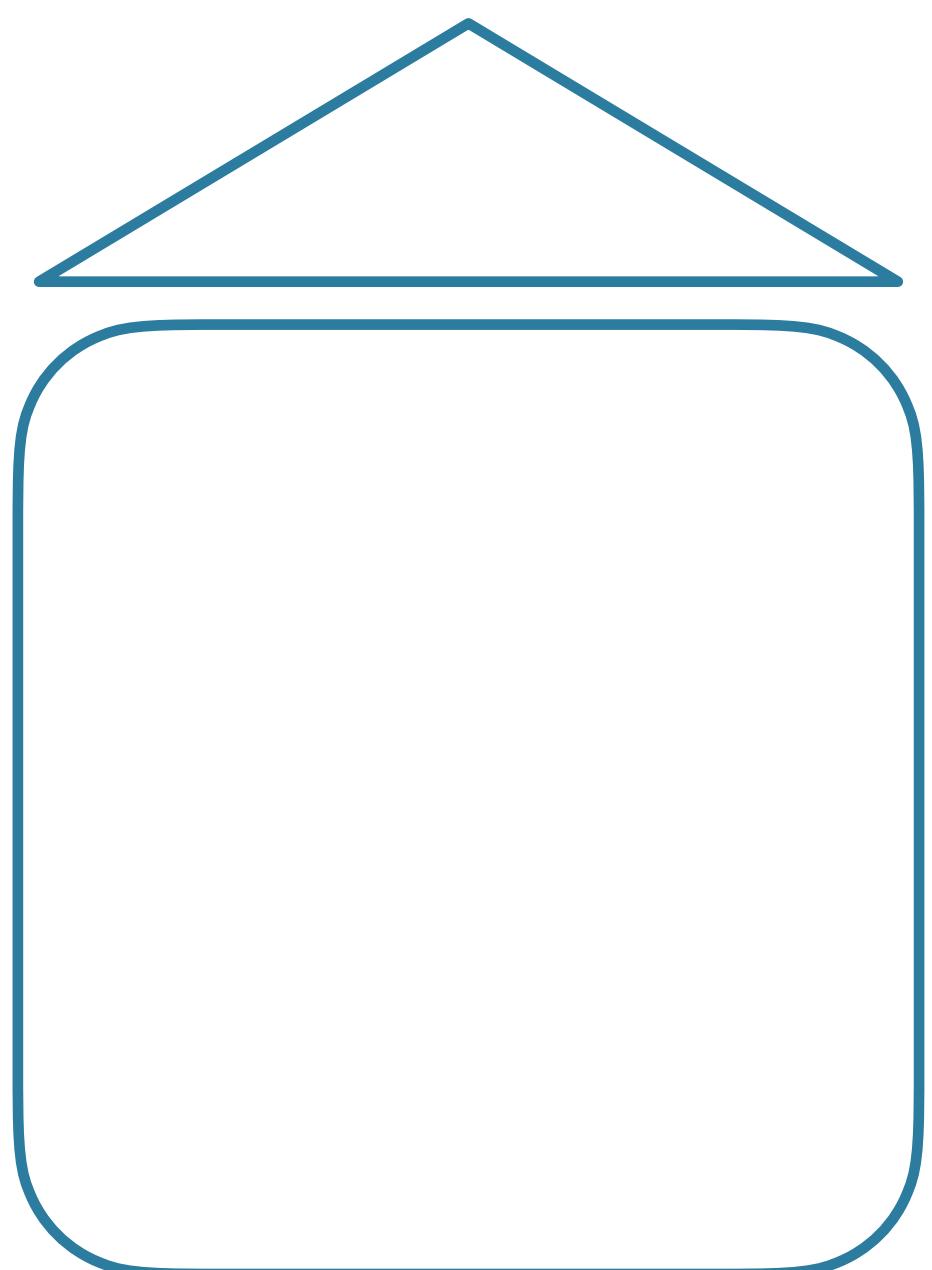
Task 2 inputs

Standard Learning

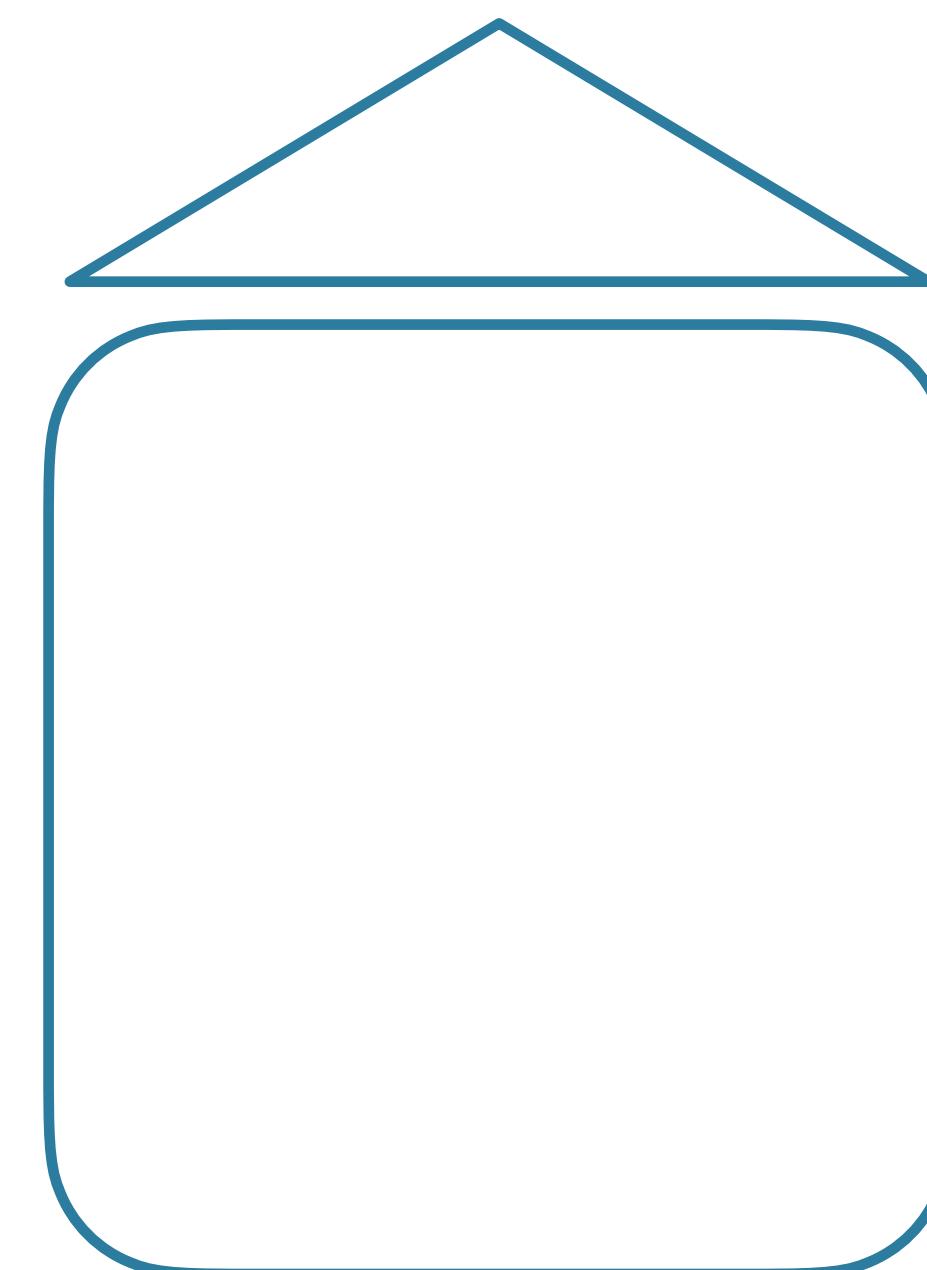
Task 1 outputs



Task 2 outputs



Task 3 outputs



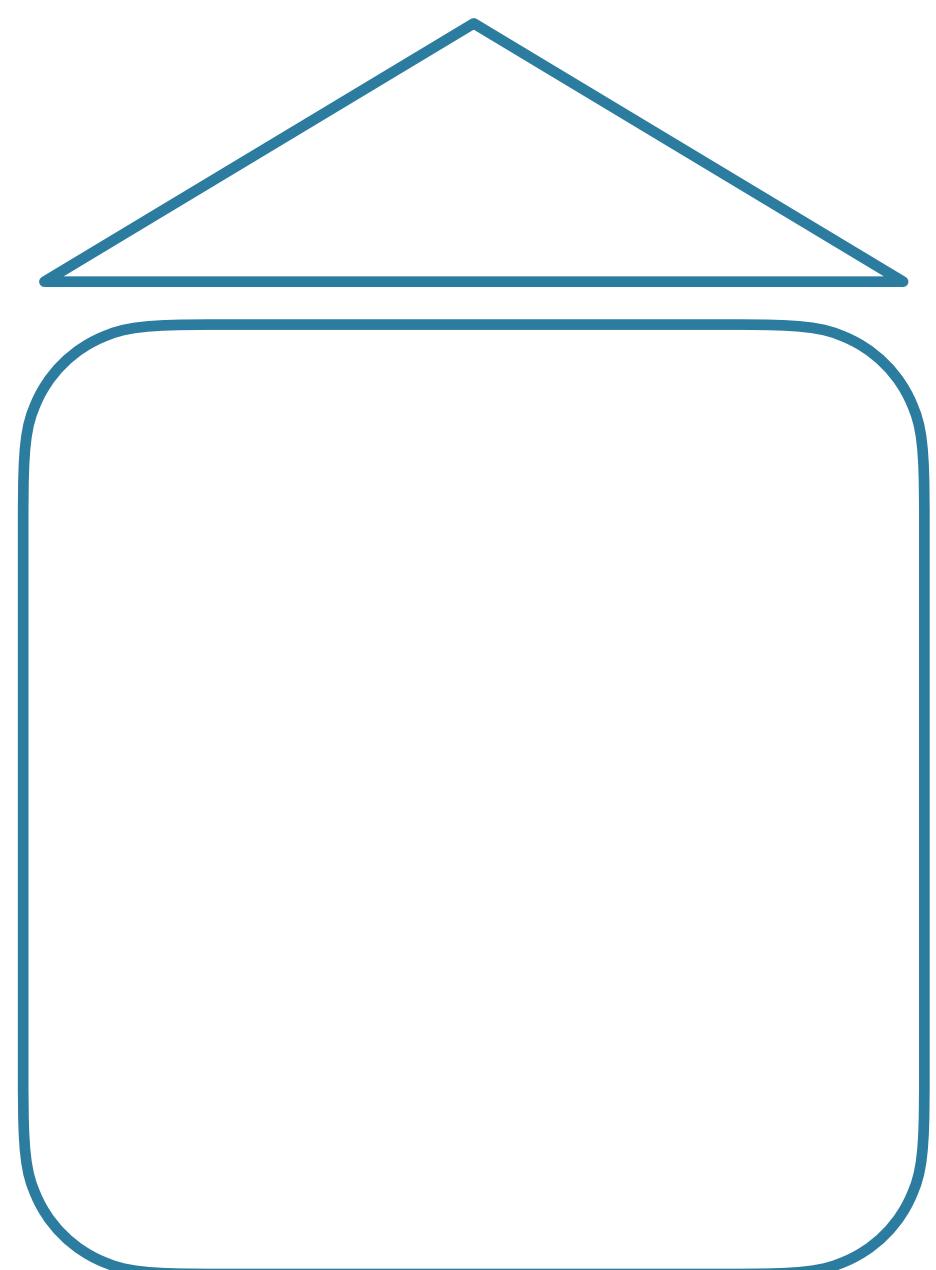
Task 1 inputs

Task 2 inputs

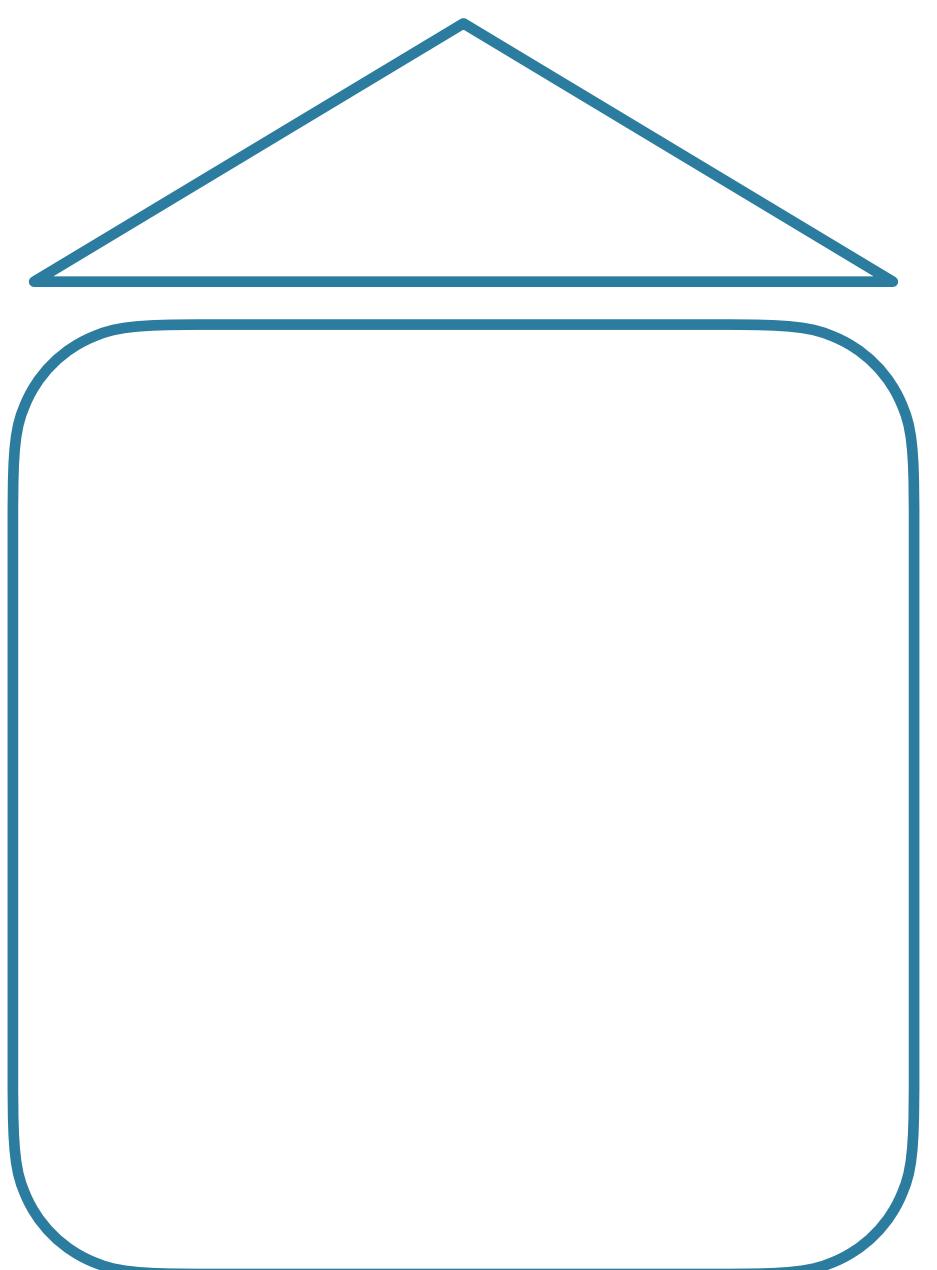
Task 3 inputs

Standard Learning

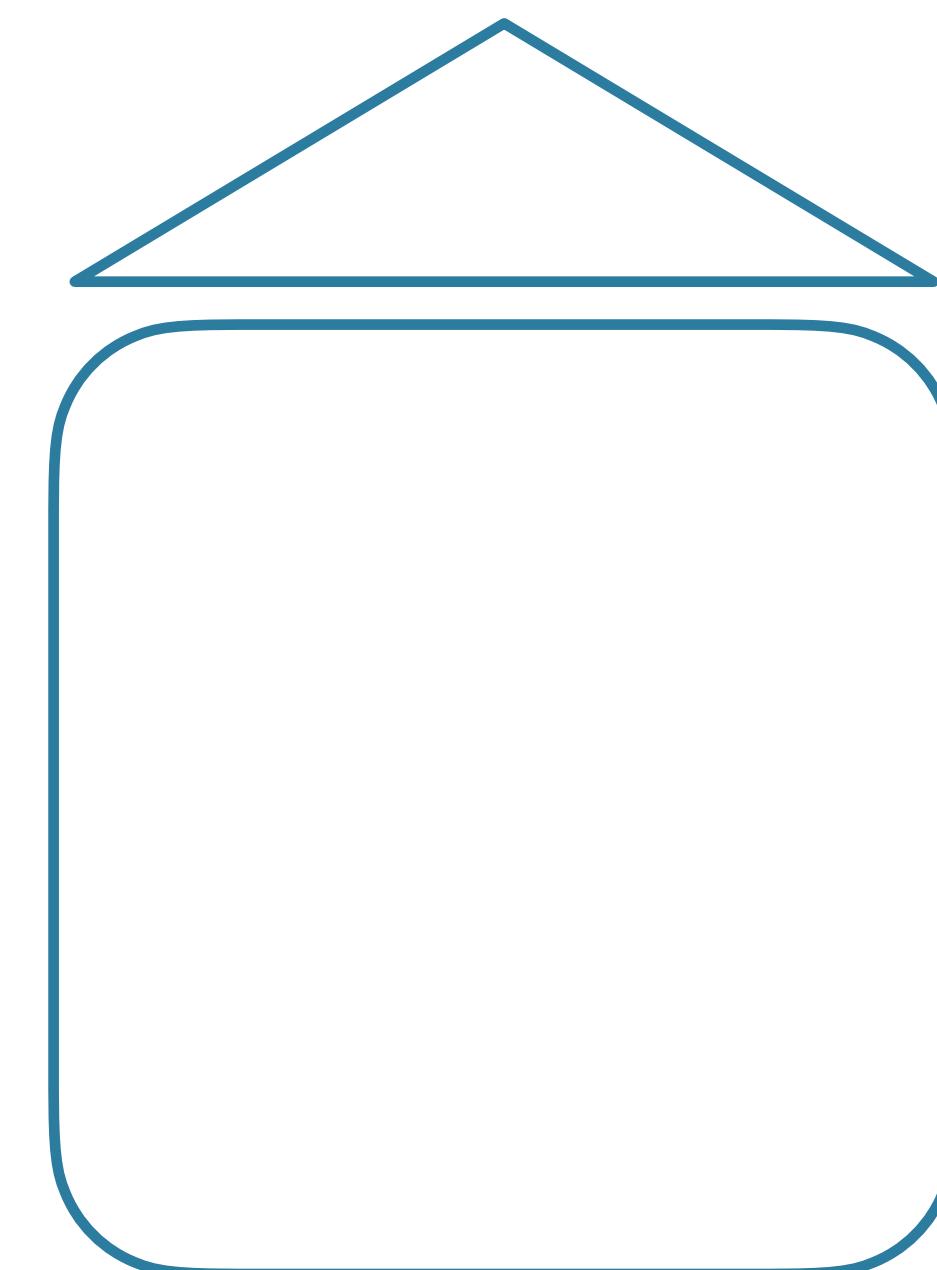
Task 1 outputs



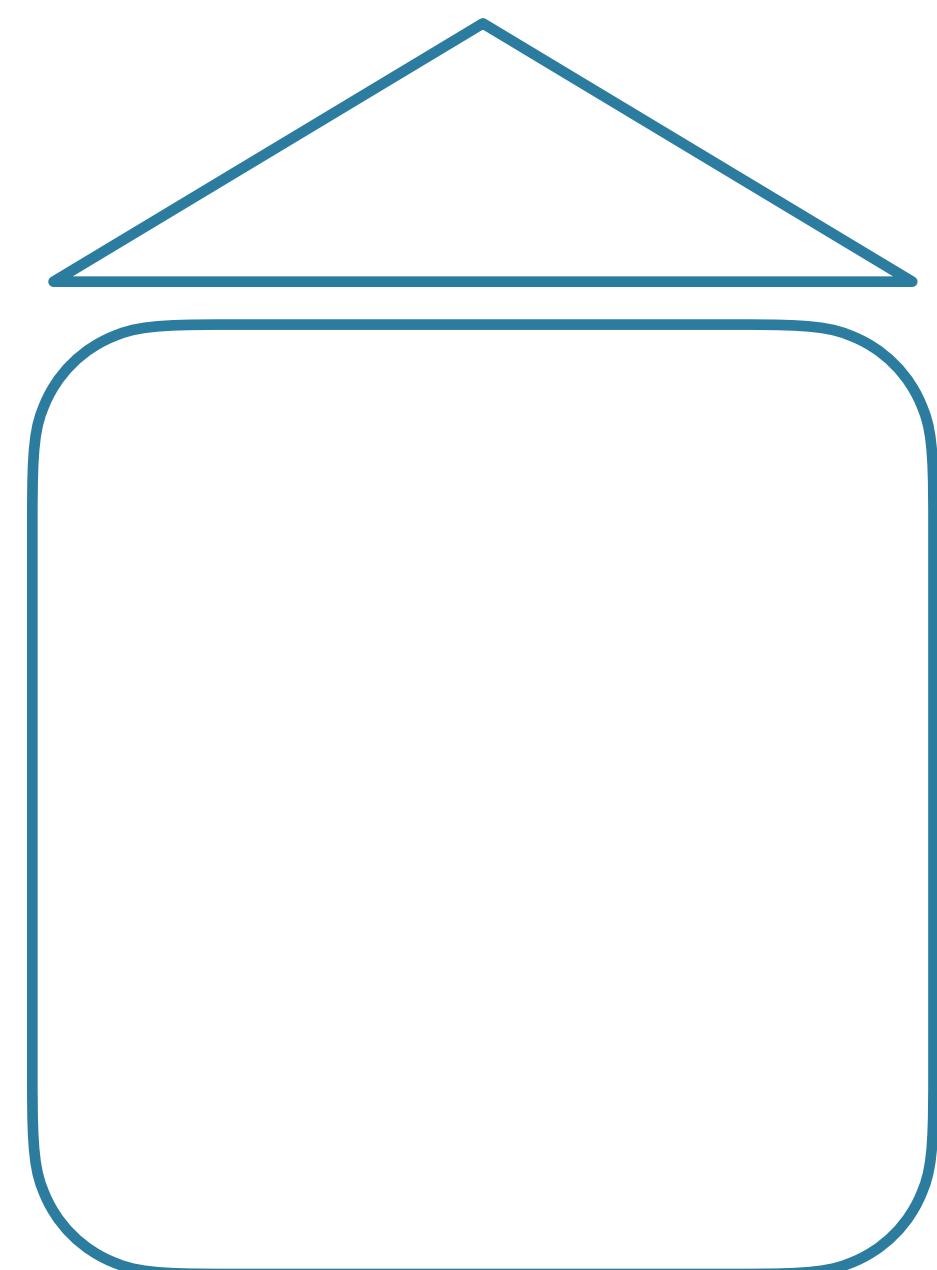
Task 2 outputs



Task 3 outputs



Task 4 outputs



Task 1 inputs

Task 2 inputs

Task 3 inputs

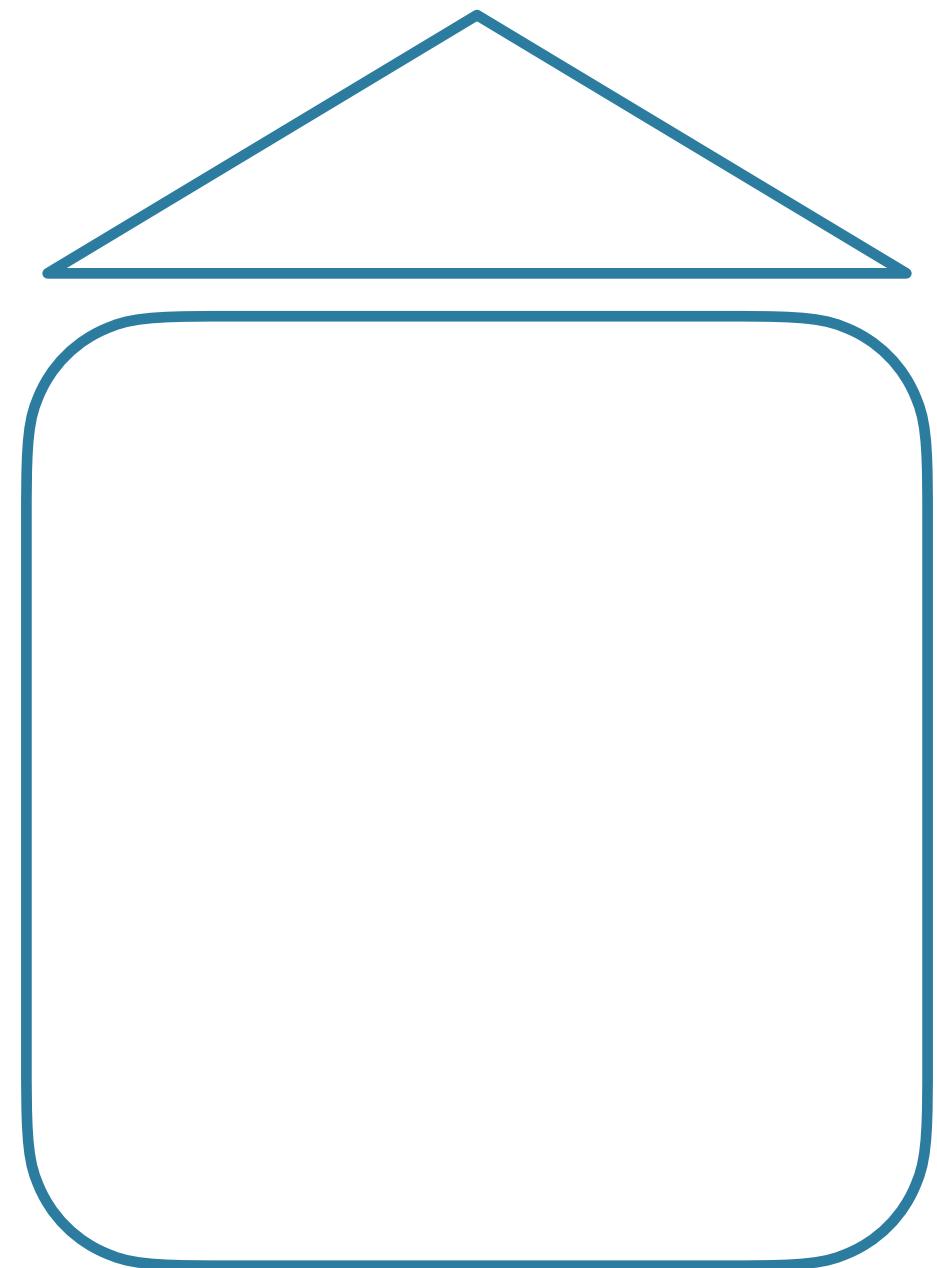
Task 4 inputs

Standard Learning

- New task = new model
- Expensive!
 - Training time
 - Storage space
 - Data availability
 - Can be impossible in low-data regimes

Transfer Learning

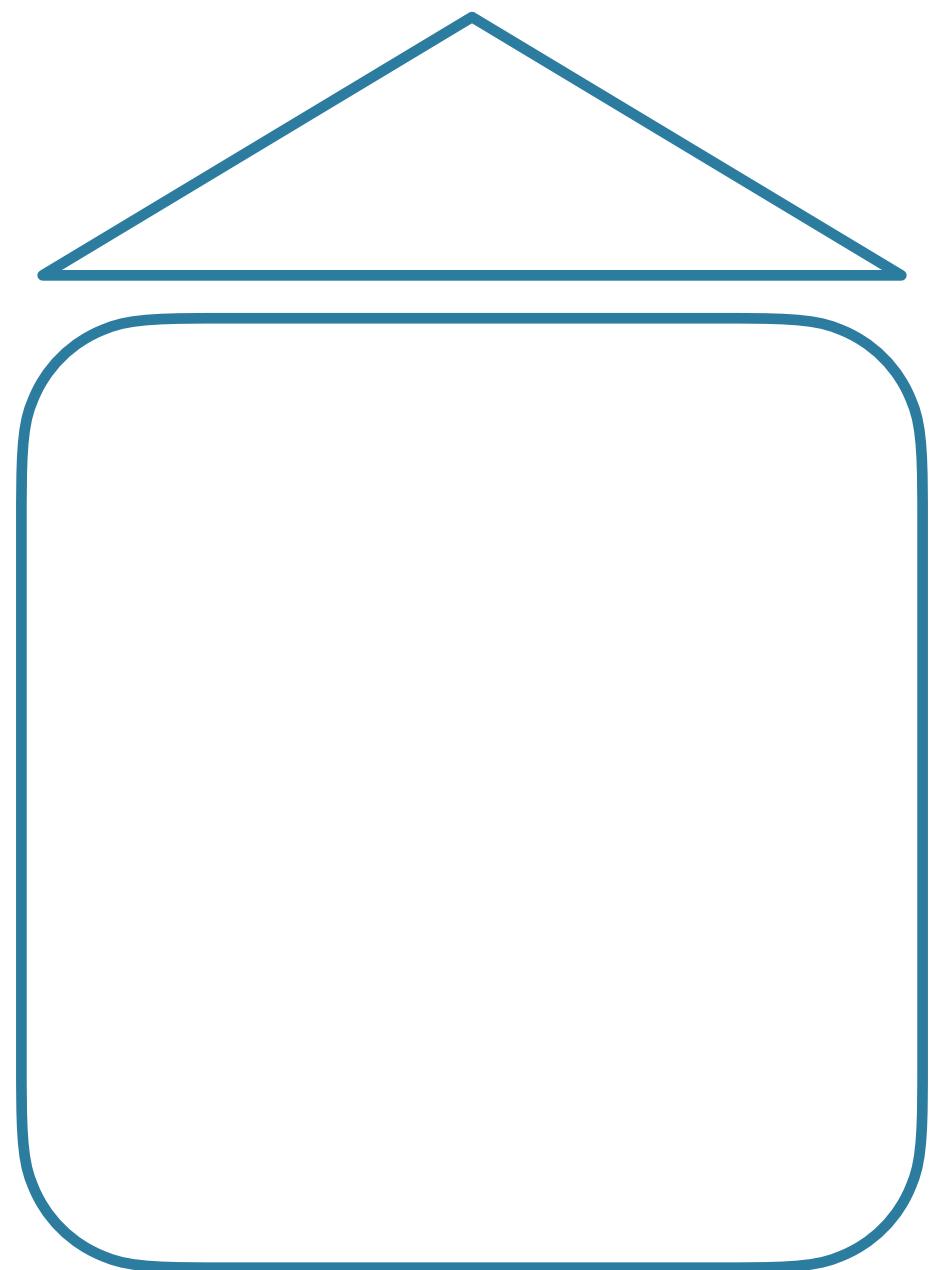
“pre-training” task outputs



“pre-training” task inputs

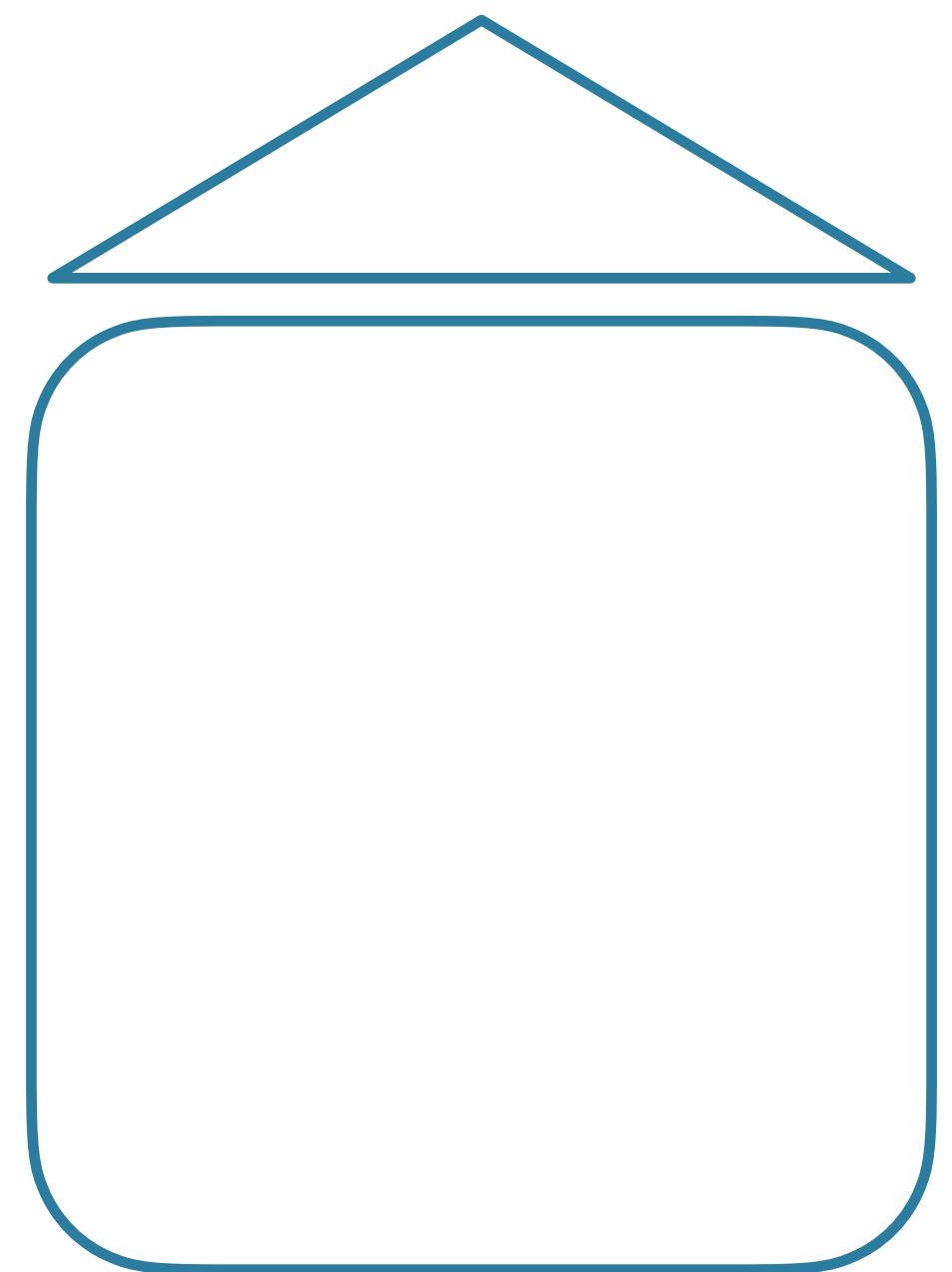
Transfer Learning

“pre-training” task outputs



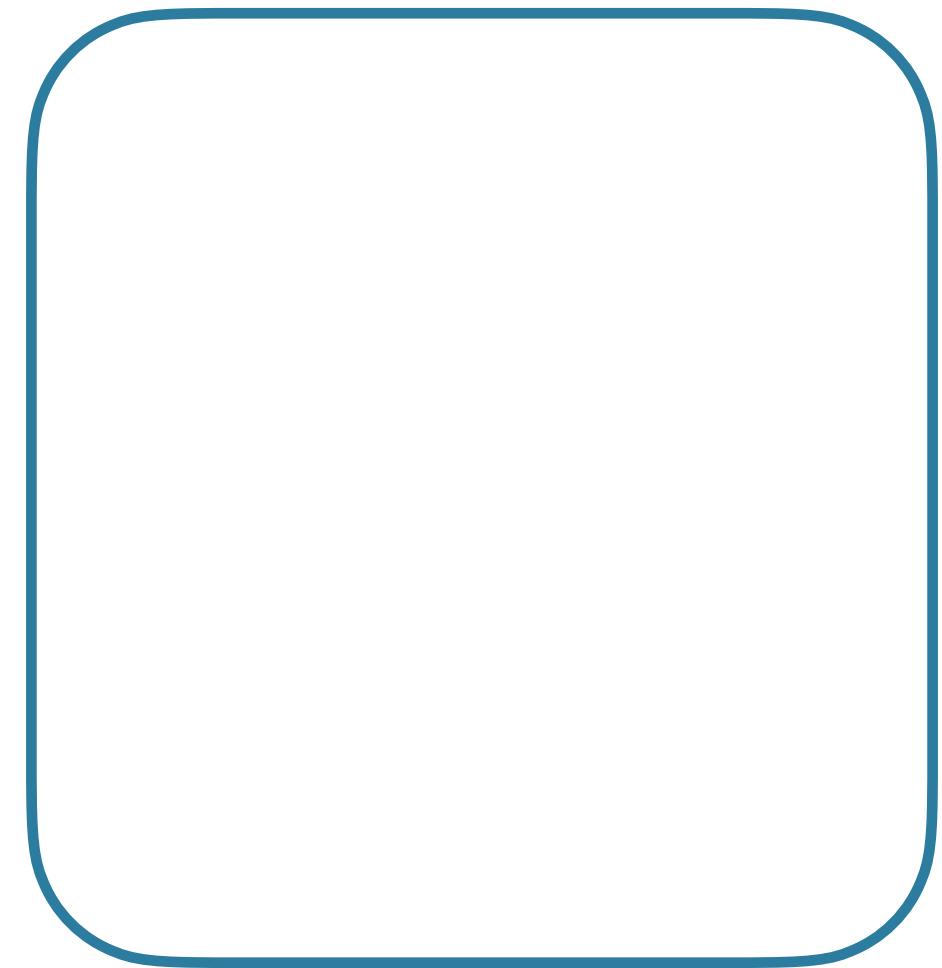
Transfer Learning

“pre-training” task outputs



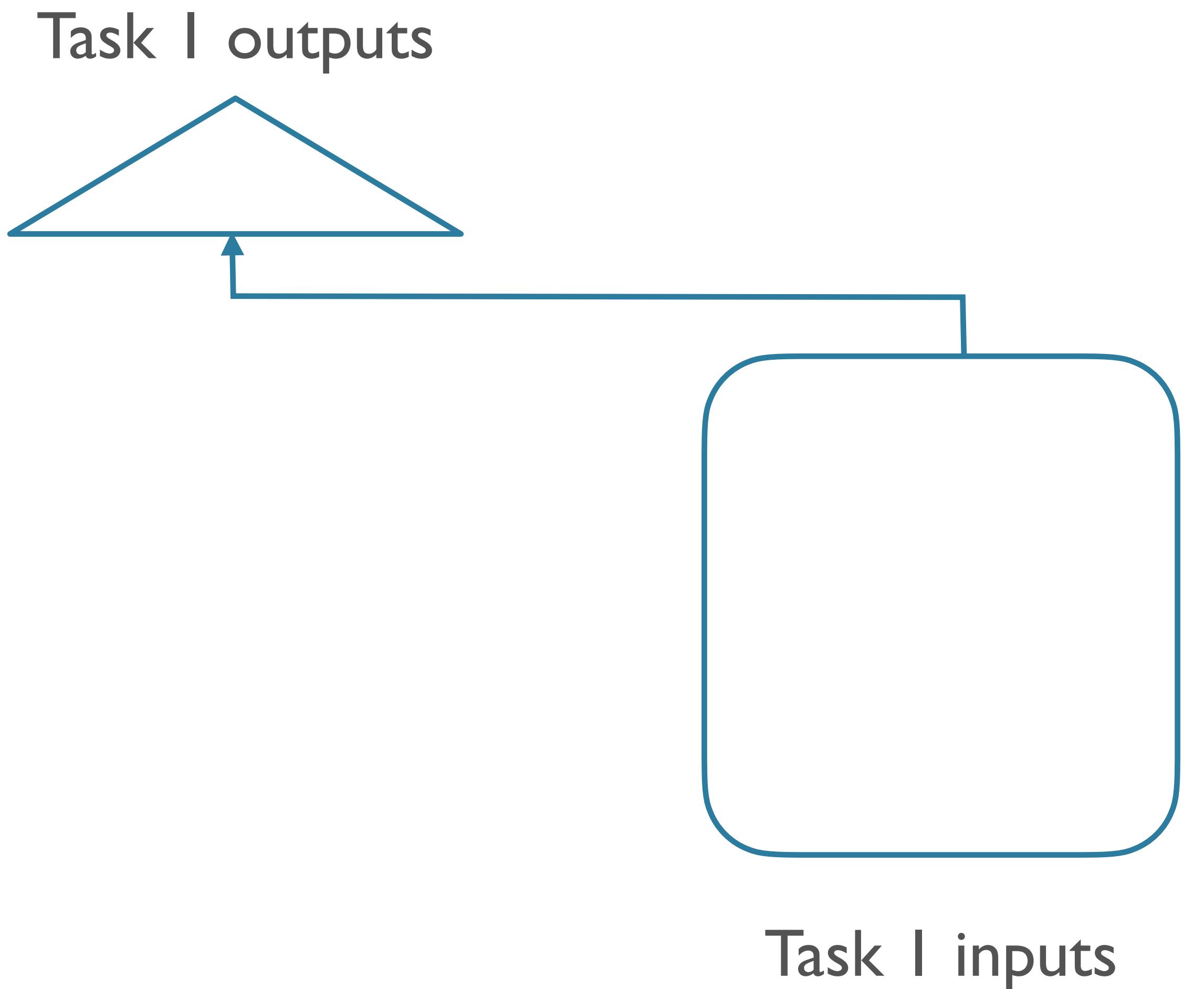
Task I inputs

Transfer Learning

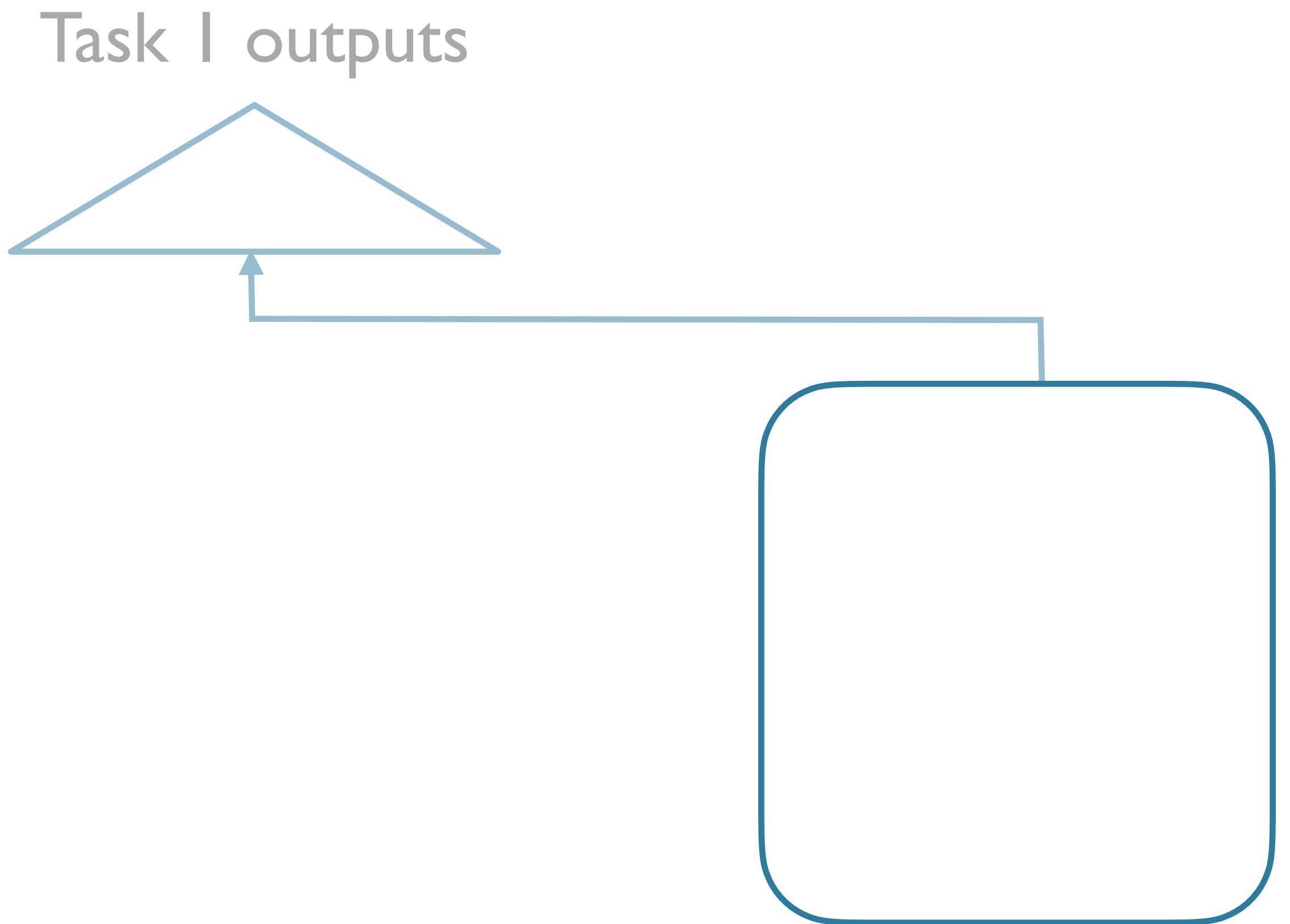


Task I inputs

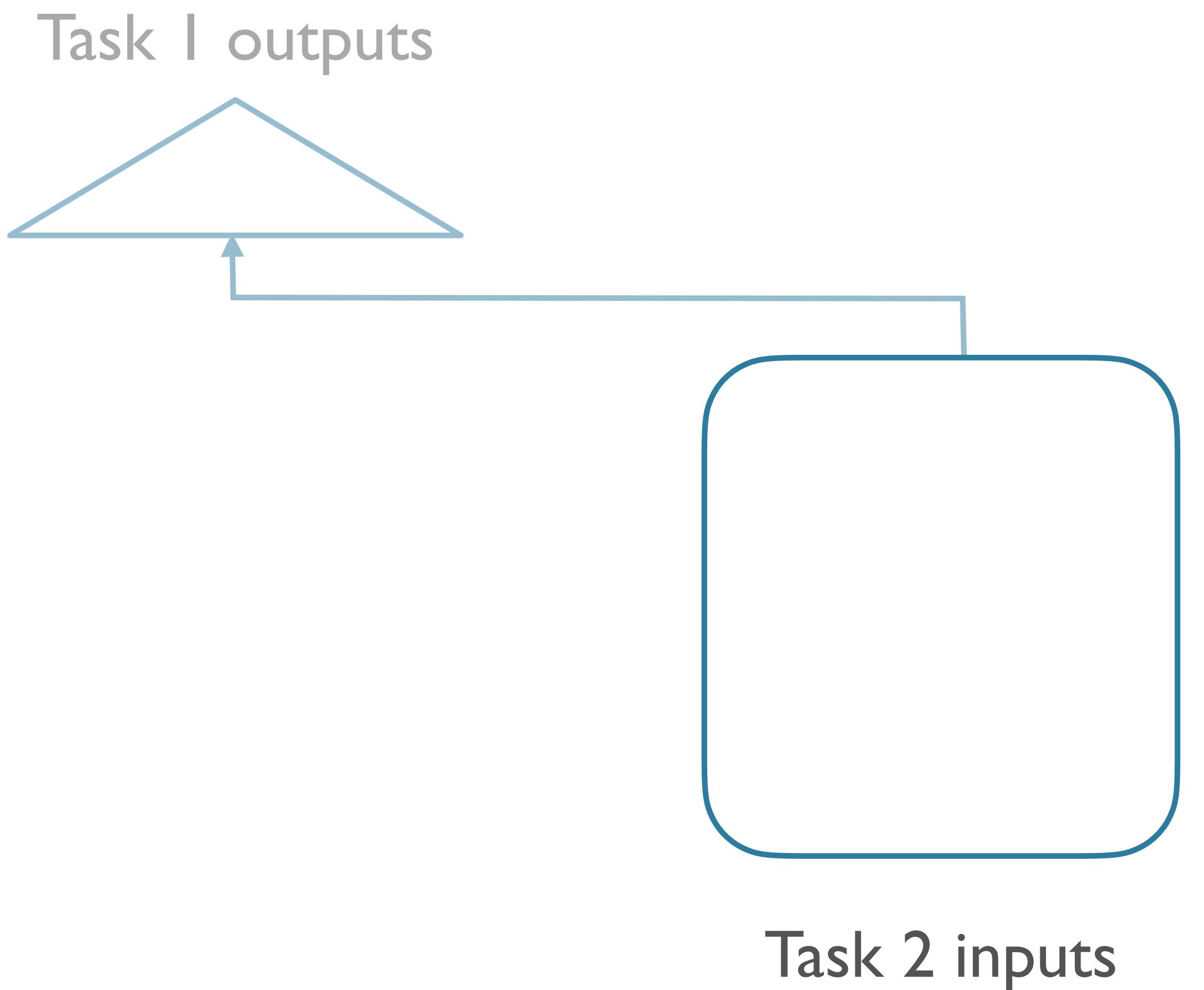
Transfer Learning



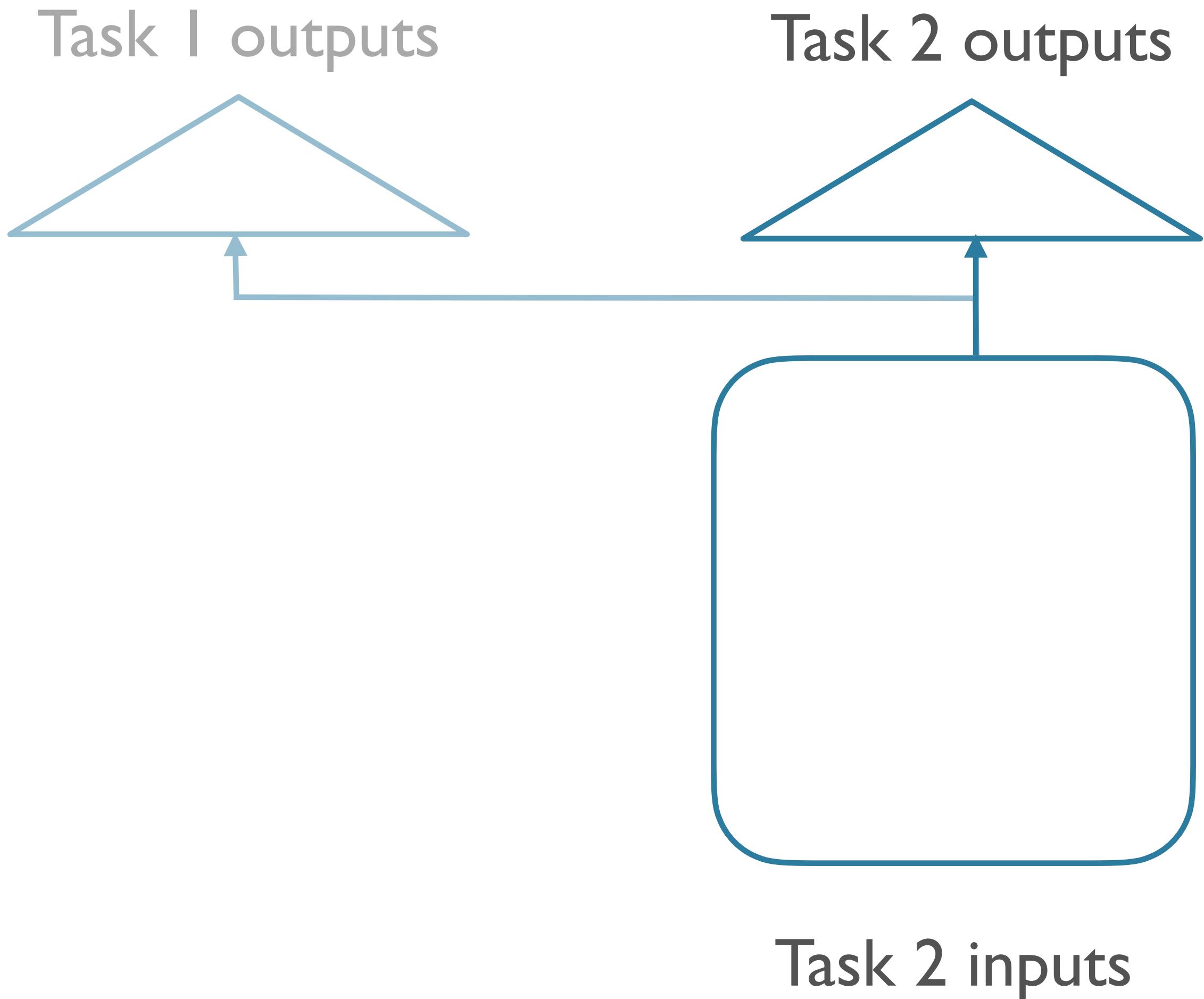
Transfer Learning



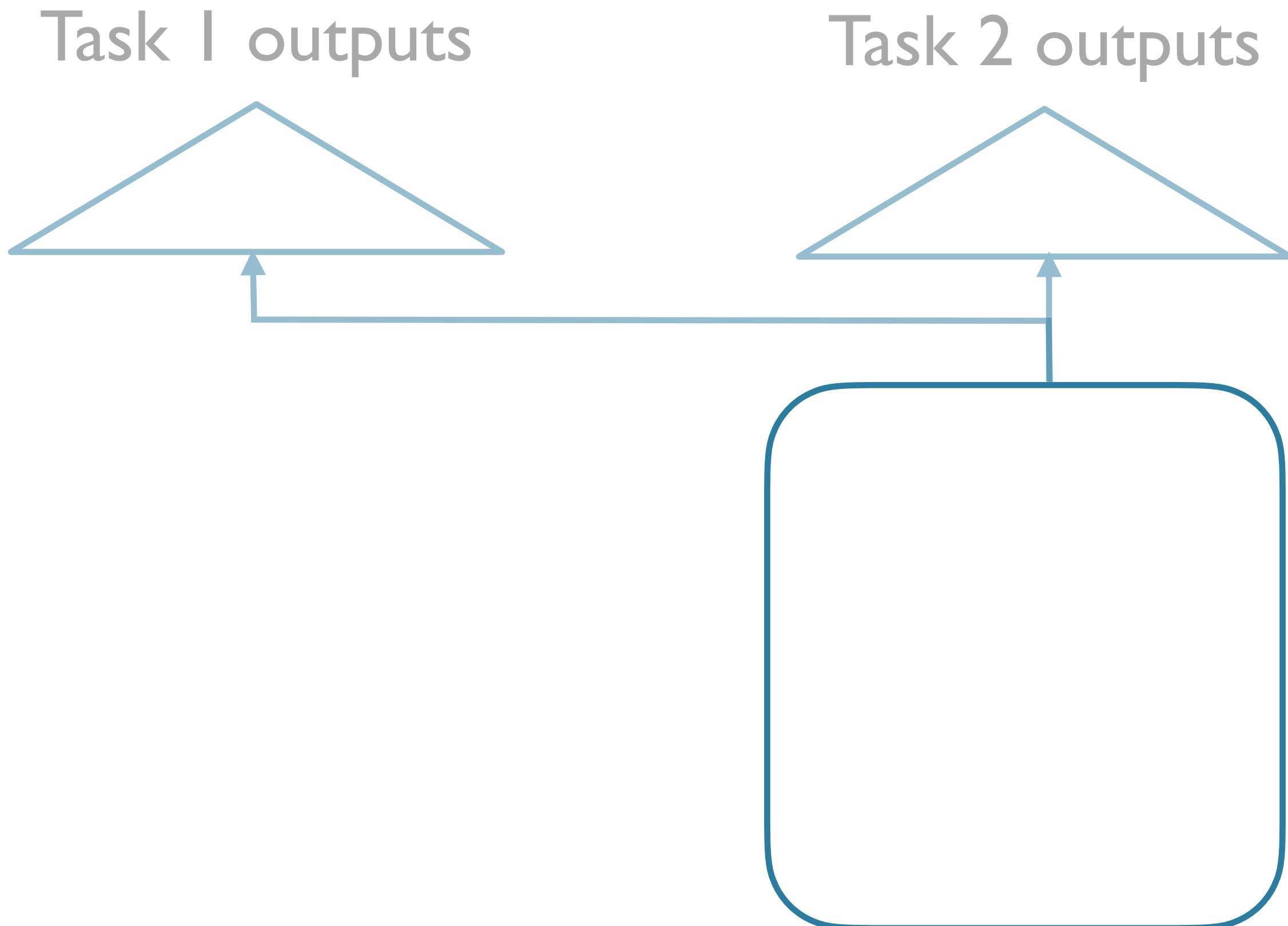
Transfer Learning



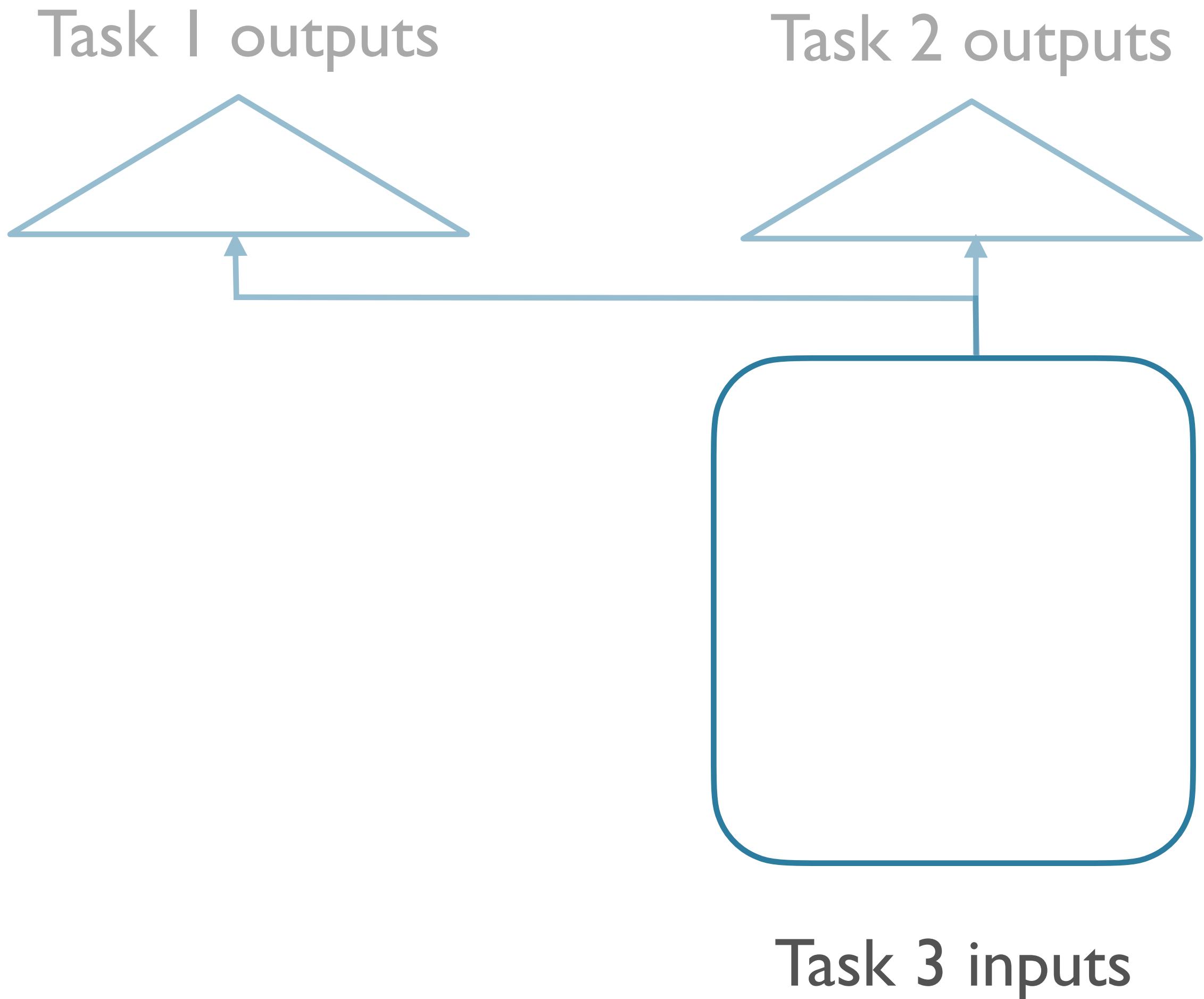
Transfer Learning



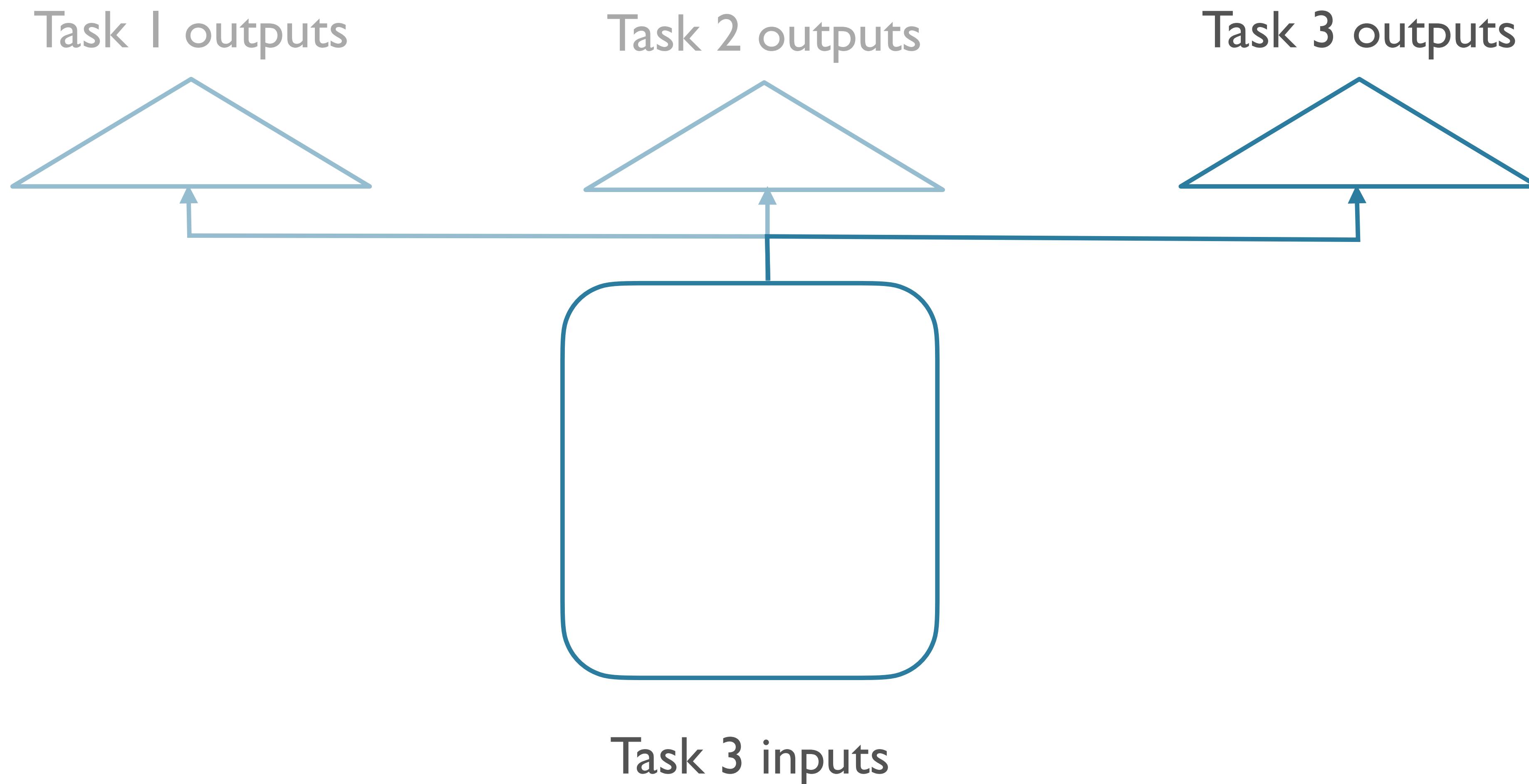
Transfer Learning



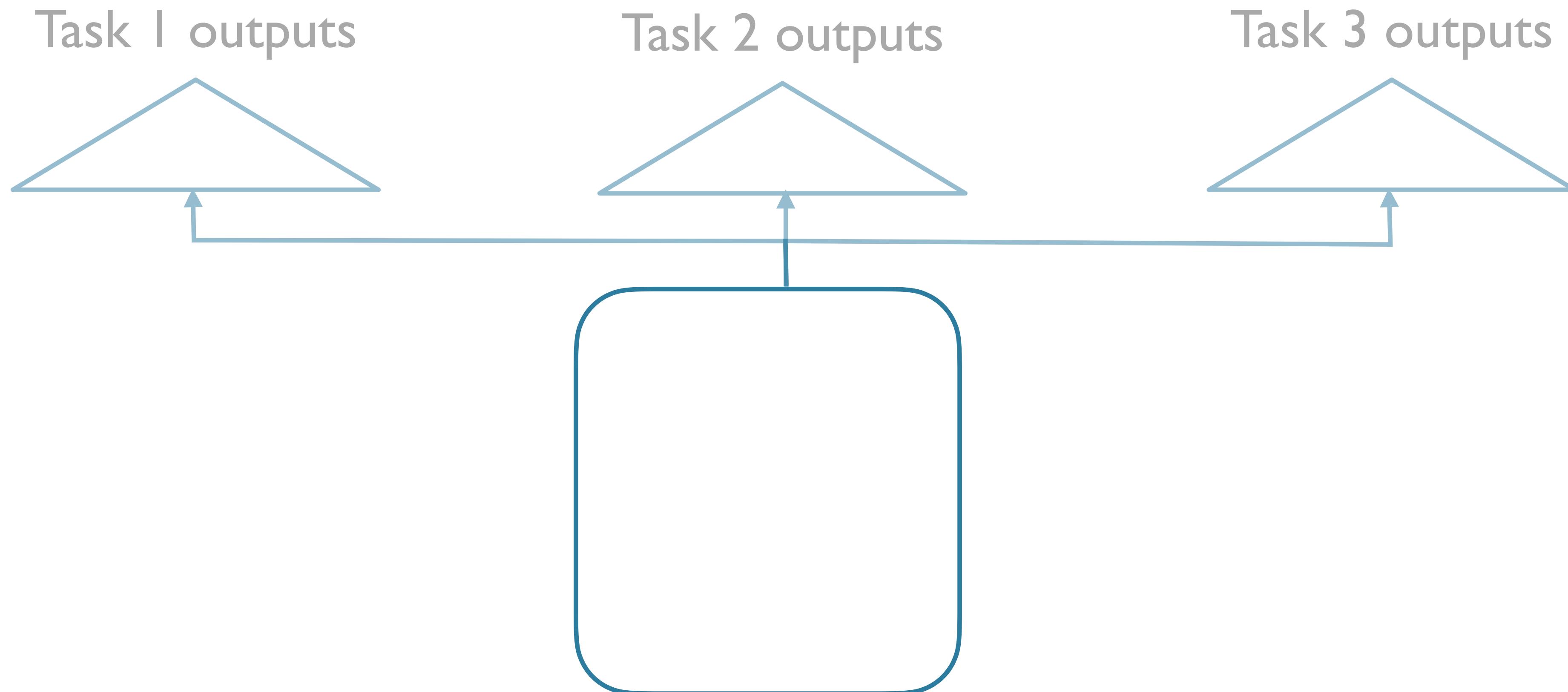
Transfer Learning



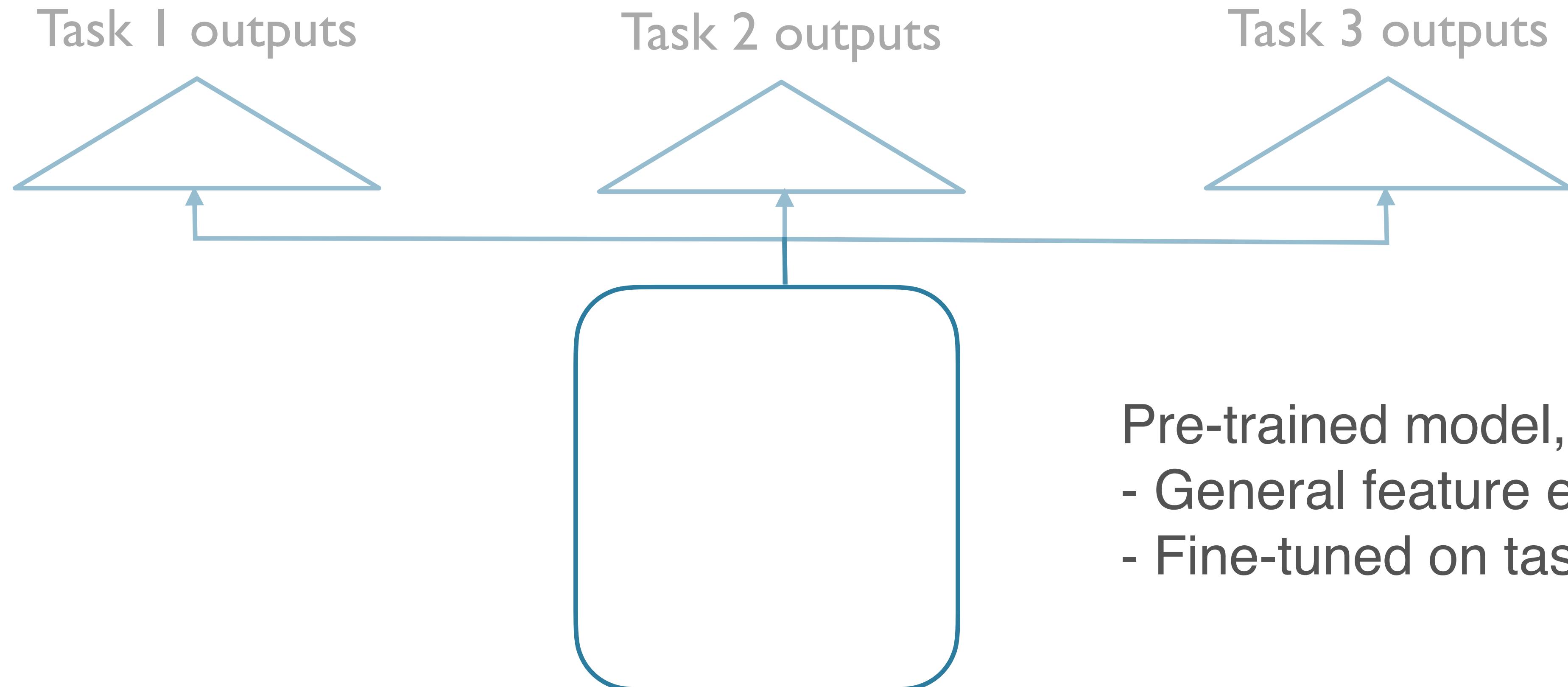
Transfer Learning



Transfer Learning



Transfer Learning



Pre-training + Fine-tuning

- Step 1: *pre-train* a model on a “general” task
 - Questions: which task for pre-training? More in a minute.
 - Goal: produce general-purpose representations of the input (“representation learning”), that will be useful when “transferred” to a more specific task.
- Step 2: *fine-tune* that model on the main task
 - Replace the “head” of the model with some task-specific layers
 - Run supervised training with the resulting model

Transfer Learning in Computer Vision

CNN Features off-the-shelf: an Astounding Baseline for Recognition

Ali Sharif Razavian Hossein Azizpour Josephine Sullivan Stefan Carlsson

CVAP, KTH (Royal Institute of Technology)
Stockholm, Sweden

{razavian,azizpour,sullivan,stefanc}@csc.kth.se

“We use features extracted from the OverFeat network as a generic image representation to tackle the diverse range of recognition tasks of object image classification, scene recognition, fine grained recognition, attribute detection and image retrieval applied to a diverse set of datasets. We selected these tasks and datasets as they gradually move further away from the original task and data the OverFeat network was trained to solve [cf. ImageNet]. Astonishingly, we report consistent superior results compared to the highly tuned state-of-the-art systems in all the visual classification tasks on various datasets”

Current Benchmarks

Rank	Name	Model	URL	Score	BoolQ	CB	COPA	MultiRC	ReCoRD	RTE	WiC	WSC	AX-g	AX-b
1	JDExplore d-team	Vega v2		91.3	90.5	98.6/99.2	99.4	88.2/62.4	94.4/93.9	96.0	77.4	98.6	100.0/50.0	-0.4
+	2 Liam Fedus	ST-MoE-32B		91.2	92.4	96.9/98.0	99.2	89.6/65.8	95.1/94.4	93.5	77.7	96.6	96.1/94.1	72.3
3	Microsoft Alexander v-team	Turing NLR v5		90.9	92.0	95.9/97.6	98.2	88.4/63.0	96.4/95.9	94.1	77.1	97.3	93.3/95.5	67.8
4	ERNIE Team - Baidu	ERNIE 3.0		90.6	91.0	98.6/99.2	97.4	88.6/63.2	94.7/94.2	92.6	77.4	97.3	92.7/94.7	68.6
5	Yi Tay	PaLM 540B		90.4	91.9	94.4/96.0	99.0	88.7/63.6	94.2/93.3	94.1	77.4	95.9	95.5/90.4	72.9
+	6 Zirui Wang	T5 + UDG, Single Model (Google Brain)		90.4	91.4	95.8/97.6	98.0	88.3/63.0	94.2/93.5	93.0	77.9	96.6	92.7/91.9	69.1
+	7 DeBERTa Team - Microsoft	DeBERTa / TuringNLRv4		90.3	90.4	95.7/97.6	98.4	88.2/63.7	94.5/94.1	93.2	77.5	95.9	93.3/93.8	66.7
8	SuperGLUE Human Baselines	SuperGLUE Human Baselines		89.8	89.0	95.8/98.9	100.0	81.8/51.9	91.7/91.3	93.6	80.0	100.0	99.3/99.7	76.6
+	9 T5 Team - Google	T5		89.3	91.2	93.9/96.8	94.8	88.1/63.3	94.1/93.4	92.5	76.9	93.8	92.7/91.9	65.6
10	SPoT Team - Google	Frozen T5 1.1 + SPoT		89.2	91.1	95.8/97.6	95.6	87.9/61.9	93.3/92.4	92.9	75.8	93.8	83.1/82.6	66.9
+	11 Huawei Noah's Ark Lab	NEZHA-Plus		86.7	87.8	94.4/96.0	93.6	84.6/55.1	90.1/89.6	89.1	74.6	93.2	87.1/74.4	58.0
+	12 Alibaba PAI&ICBU	PAI Albert		86.1	88.1	92.4/96.4	91.8	84.6/54.7	89.0/88.3	88.8	74.1	93.2	98.3/99.2	75.6
+	13 Infosys : DAWN : AI Research	RoBERTa-iCETS		86.0	88.5	93.2/95.2	91.2	86.4/58.2	89.9/89.3	89.9	72.9	89.0	88.8/81.5	61.8
+	14 Tencent Jarvis Lab	RoBERTa (ensemble)		85.9	88.2	92.5/95.6	90.8	84.4/53.4	91.5/91.0	87.9	74.1	91.8	89.3/75.6	57.6
15	Zhuiyi Technology	RoBERTa-mtl-adv		85.7	87.1	92.4/95.6	91.2	85.1/54.3	91.7/91.3	88.1	72.1	91.8	91.0/78.1	58.5
16	Facebook AI	RoBERTa		84.6	87.1	90.5/95.2	90.6	84.4/52.5	90.6/90.0	88.2	69.9	89.0	91.0/78.1	57.9

Language Model Pre-training

Where to transfer *from*?

Where to transfer *from*?

- Goal: find a linguistic task that will build general-purpose / *transferable* representations

Where to transfer *from*?

- Goal: find a linguistic task that will build general-purpose / *transferable* representations
- Possibilities:

Where to transfer *from*?

- Goal: find a linguistic task that will build general-purpose / *transferable* representations
- Possibilities:
 - Constituency or dependency parsing

Where to transfer *from*?

- Goal: find a linguistic task that will build general-purpose / *transferable* representations
- Possibilities:
 - Constituency or dependency parsing
 - Semantic parsing

Where to transfer *from*?

- Goal: find a linguistic task that will build general-purpose / *transferable* representations
- Possibilities:
 - Constituency or dependency parsing
 - Semantic parsing
 - Machine translation

Where to transfer *from*?

- Goal: find a linguistic task that will build general-purpose / *transferable* representations
- Possibilities:
 - Constituency or dependency parsing
 - Semantic parsing
 - Machine translation
 - QA

Where to transfer *from*?

- Goal: find a linguistic task that will build general-purpose / *transferable* representations
- Possibilities:
 - Constituency or dependency parsing
 - Semantic parsing
 - Machine translation
 - QA
 - ...

Where to transfer *from*?

- Goal: find a linguistic task that will build general-purpose / *transferable* representations
- Possibilities:
 - Constituency or dependency parsing
 - Semantic parsing
 - Machine translation
 - QA
 - ...
- Scalability issue: all require expensive annotation

Language Modeling

Language Modeling

- A good language model should produce good *general-purpose* and *transferable* representations

Language Modeling

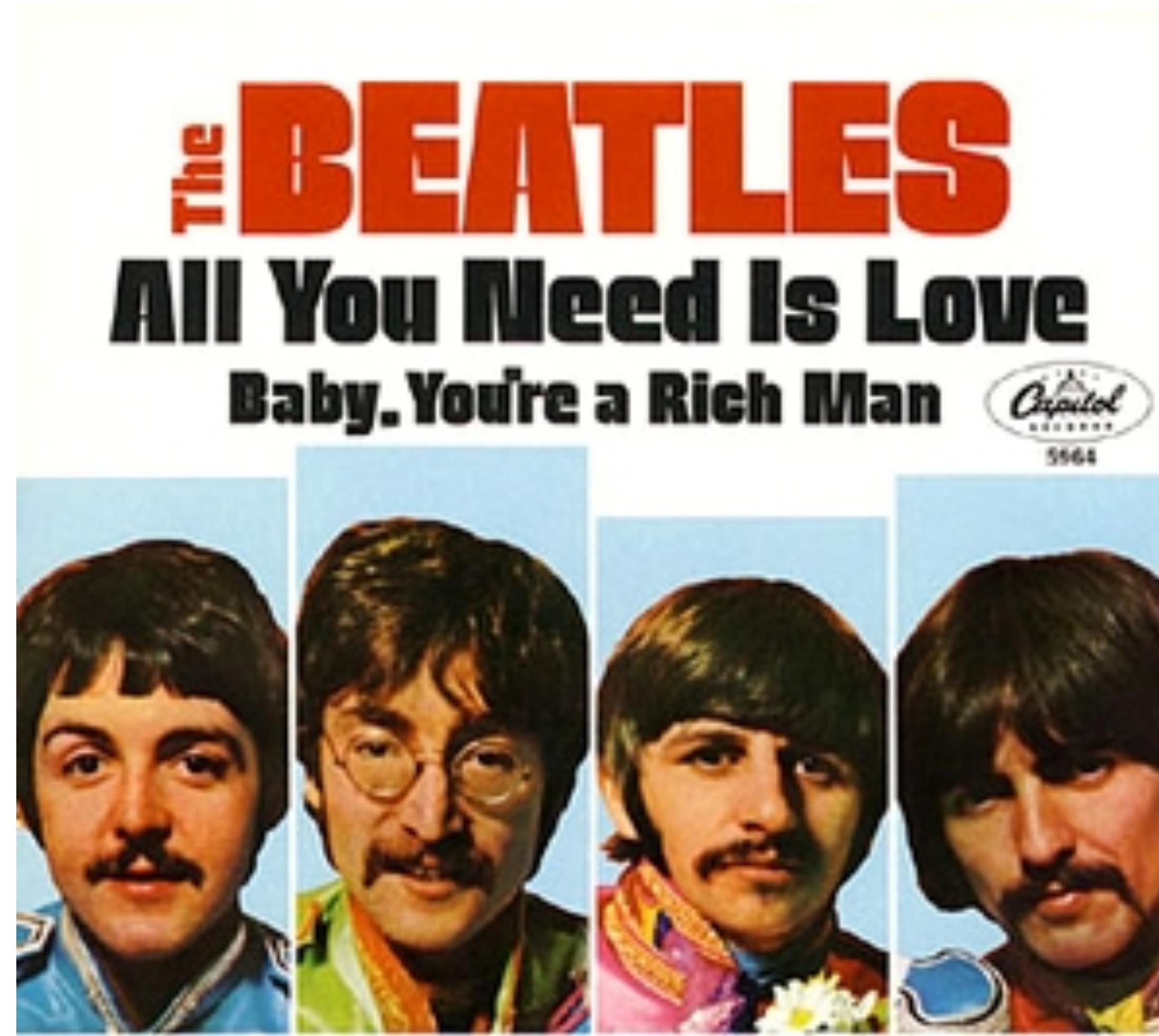
- A good language model should produce good *general-purpose* and *transferable* representations
- Linguistic knowledge:
 - The bicycles, even though old, were in good shape because _____ ...
 - The bicycle, even though old, was in good shape because _____ ...

Language Modeling

- A good language model should produce good *general-purpose* and *transferable* representations
- Linguistic knowledge:
 - The bicycles, even though old, were in good shape because _____ ...
 - The bicycle, even though old, was in good shape because _____ ...
- World knowledge:
 - The University of Washington was founded in _____
 - Seattle had a huge population boom as a launching point for expeditions to _____

Data for LM is cheap

Data for LM is cheap



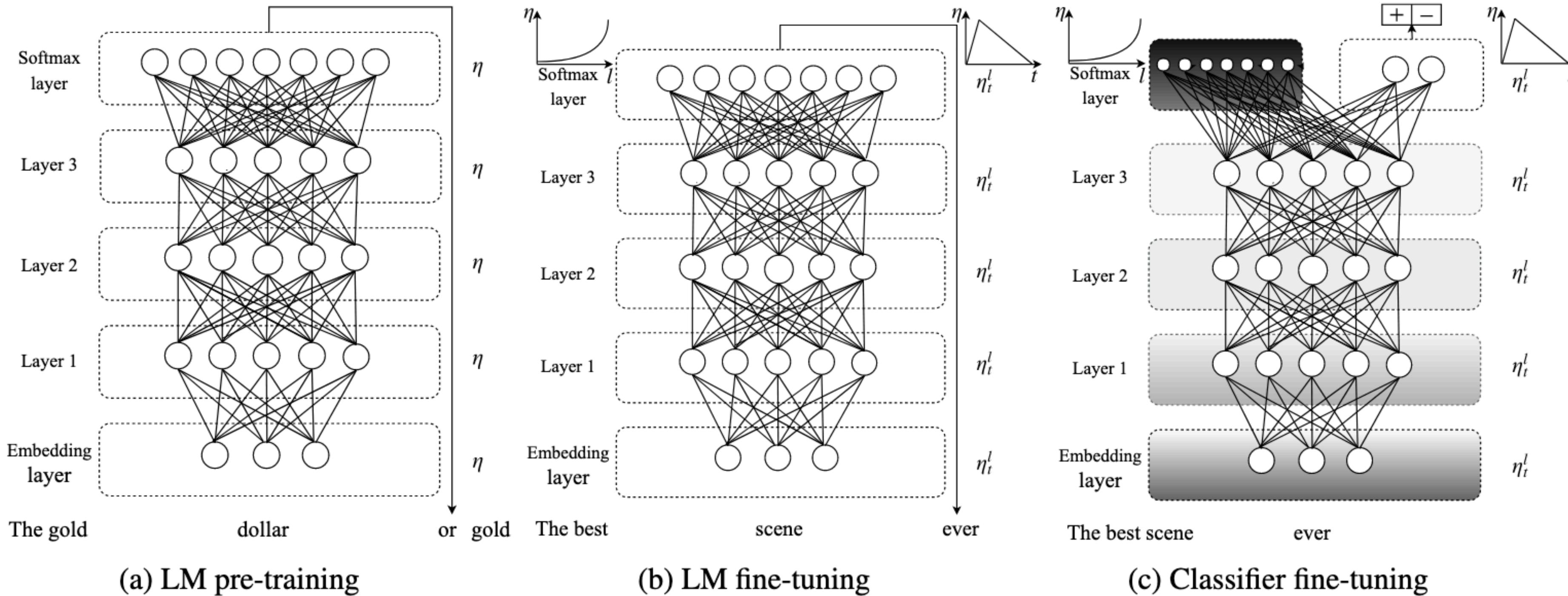
Data for LM is cheap



Language Model Pre-training

- A currently powerful paradigm for training models for NLP tasks:
 - *Pre-train* a large language model on a large amount of raw text
 - *Fine-tune* a small model on top of the LM for the task you care about
 - [or use the LM as a general feature extractor]
 - [or prompt it for “in-context learning”; later]

ULMFiT

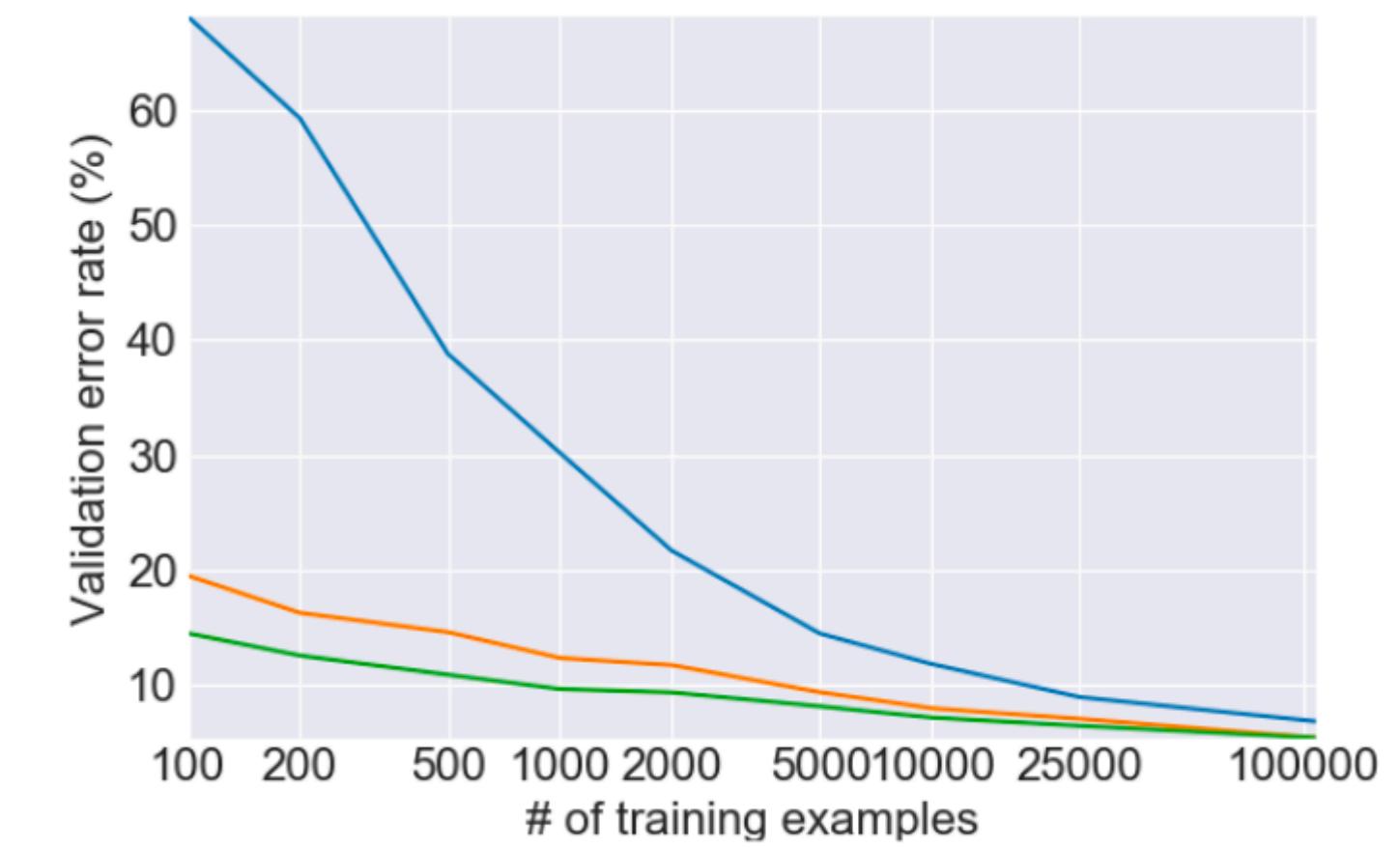
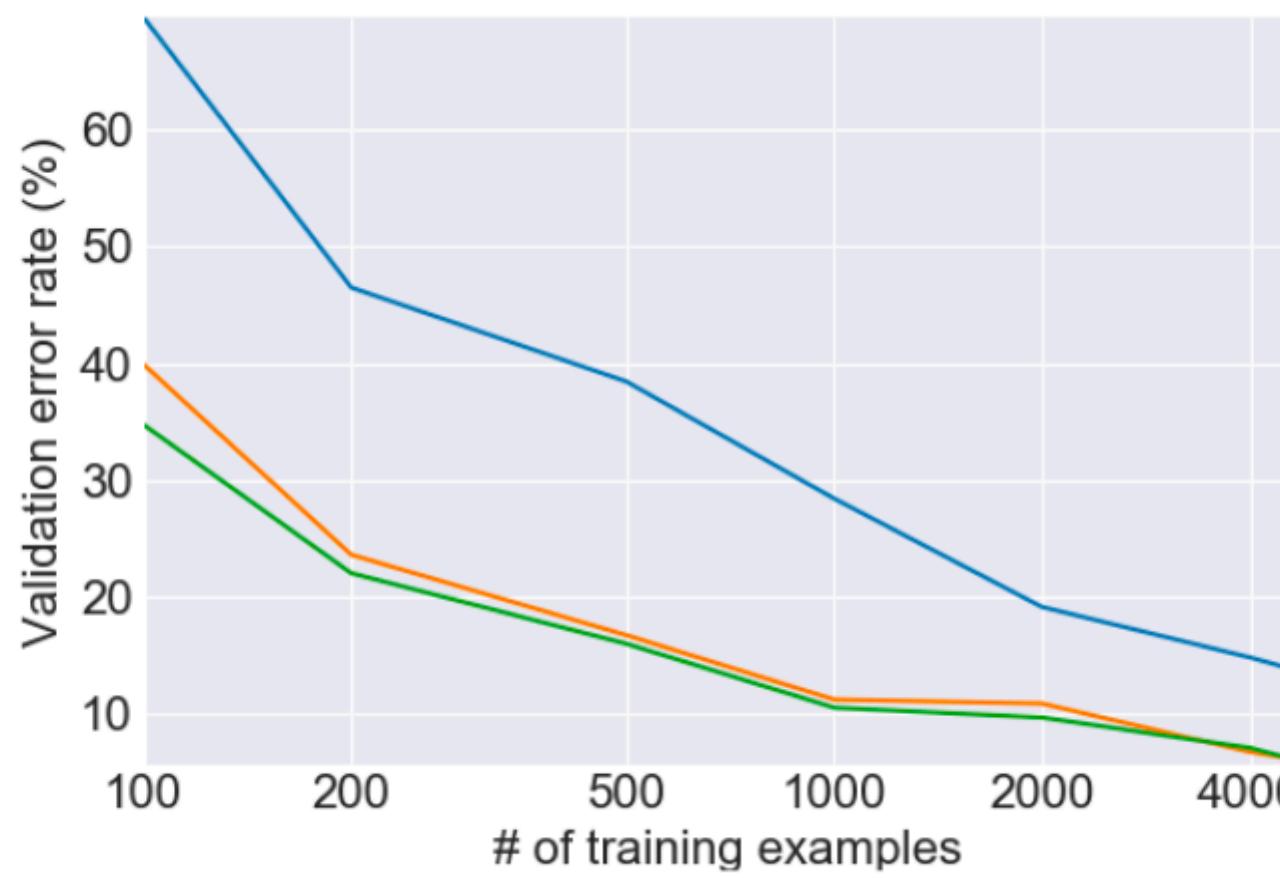
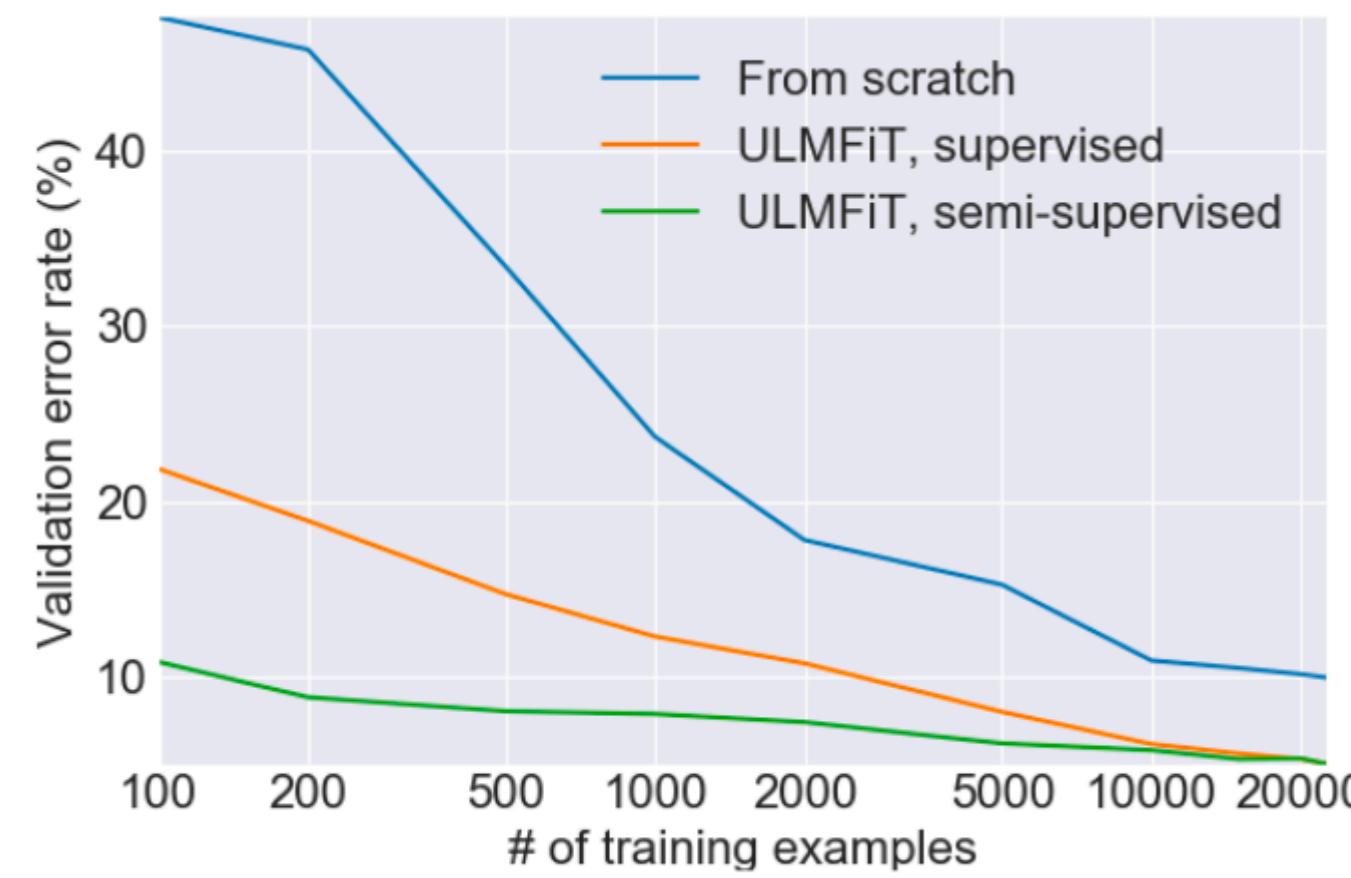


Universal Language Model Fine-tuning for Text Classification (ACL '18)

ULMFiT

Model	Test	Model	Test
IMDb	CoVe (McCann et al., 2017)	8.2	CoVe (McCann et al., 2017) 4.2
	oh-LSTM (Johnson and Zhang, 2016)	5.9	TBCNN (Mou et al., 2015) 4.0
	Virtual (Miyato et al., 2016)	5.9	LSTM-CNN (Zhou et al., 2016) 3.9
	ULMFiT (ours)	4.6	ULMFiT (ours) 3.6

ULMFiT



Deep Contextualized Word Representations

Peters et. al (2018)

Deep Contextualized Word Representations

Peters et. al (2018)

- NAACL 2018 Best Paper Award

Deep Contextualized Word Representations

Peters et. al (2018)

- NAACL 2018 Best Paper Award
- Embeddings from Language Models (ELMo)
 - [aka the OG NLP Muppet]



ELMo

Deep contextualized word representations

Matthew E. Peters[†], Mark Neumann[†], Mohit Iyyer[†], Matt Gardner[†],
`{matthewp, markn, mohiti, mattg}@allenai.org`

Christopher Clark^{*}, Kenton Lee^{*}, Luke Zettlemoyer^{†*}
`{csquared, kentonl, lsz}@cs.washington.edu`

[†]Allen Institute for Artificial Intelligence

^{*}Paul G. Allen School of Computer Science & Engineering, University of Washington

Abstract

We introduce a new type of *deep contextualized* word representation that models both (1) complex characteristics of word use (e.g., syntax and semantics), and (2) how these uses vary across linguistic contexts (i.e., to model polysemy). Our word vectors are learned functions of the internal states of a deep bidirectional language model (biLM), which is pre-trained on a large text corpus. We show that these representations can be easily added to existing models and significantly improve the state of the art across six challenging NLP problems, including question answering, textual entailment and sentiment analysis. We also present an analysis showing that exposing the deep internals of the pre-trained network is crucial, allowing downstream models to mix different types of semi-supervision signals.

guage model (LM) objective on a large text corpus. For this reason, we call them ELMo (Embeddings from Language Models) representations. Unlike previous approaches for learning contextualized word vectors (Peters et al., 2017; McCann et al., 2017), ELMo representations are deep, in the sense that they are a function of all of the internal layers of the biLM. More specifically, we learn a linear combination of the vectors stacked above each input word for each end task, which markedly improves performance over just using the top LSTM layer.

Combining the internal states in this manner allows for very rich word representations. Using intrinsic evaluations, we show that the higher-level LSTM states capture context-dependent aspects of word meaning (e.g., they can be used without modification to perform well on supervised

ELMo

Deep contextualized word representations

Matthew E. Peters[†], Mark Neumann[†], Mohit Iyyer[†], Matt Gardner[†],
`{matthewp, markn, mohiti, mattg}@allenai.org`

Christopher Clark^{*}, Kenton Lee^{*}, Luke Zettlemoyer^{†*}
`{csquared, kentonl, lsz}@cs.washington.edu`

[†]Allen Institute for Artificial Intelligence

^{*}Paul G. Allen School of Computer Science & Engineering, University of Washington

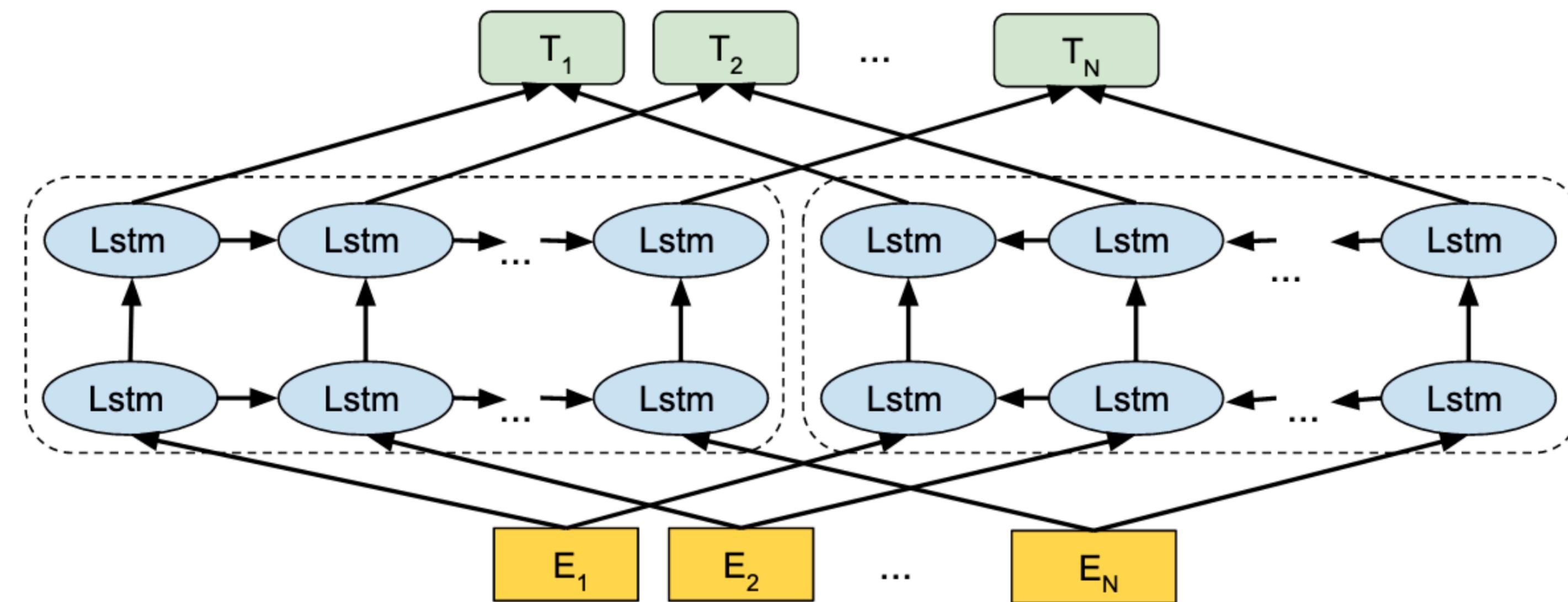
Abstract

We introduce a new type of *deep contextualized* word representation that models both (1) complex characteristics of word use (e.g., syntax and semantics), and (2) how these uses vary across linguistic contexts (i.e., to model polysemy). Our word vectors are learned functions of the internal states of a deep bidirectional language model (biLM), which is pre-trained on a large text corpus. We show that these representations can be easily added to existing models and significantly improve the state of the art across six challenging NLP problems, including question answering, textual entailment and sentiment analysis. We also present an analysis showing that exposing the deep internals of the pre-trained network is crucial, allowing downstream models to mix different types of semi-supervision signals.

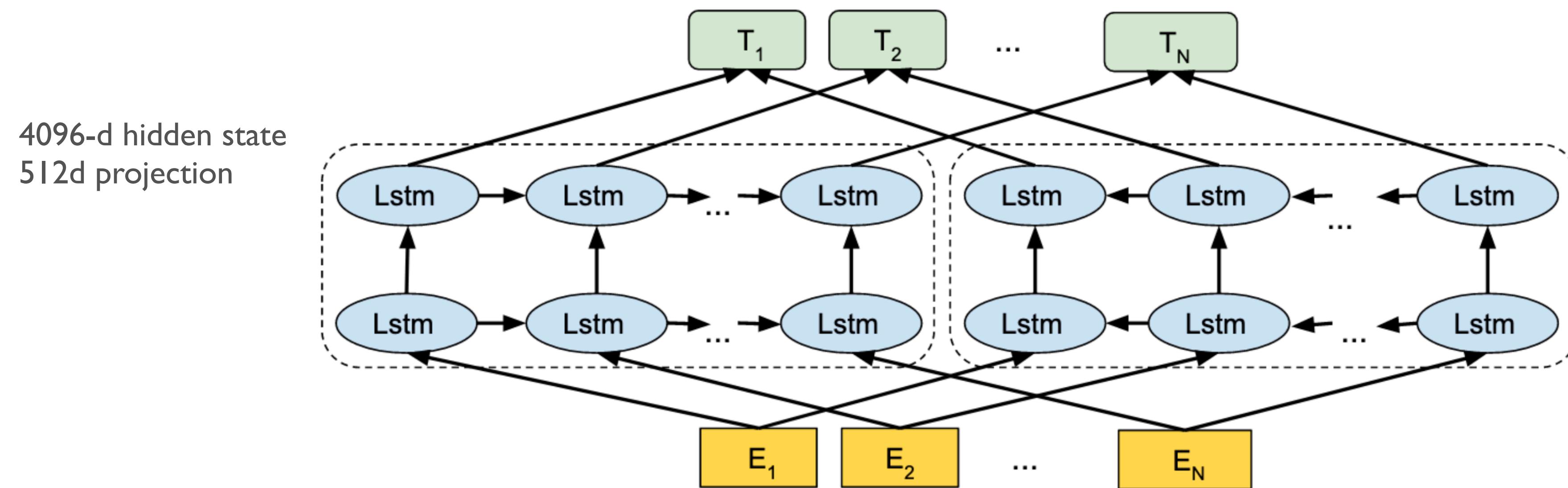
guage model (LM) objective on a large text corpus. For this reason, we call them ELMo (Embeddings from Language Models) representations. Unlike previous approaches for learning contextualized word vectors (Peters et al., 2017; McCann et al., 2017), ELMo representations are deep, in the sense that they are a function of all of the internal layers of the biLM. More specifically, we learn a linear combination of the vectors stacked above each input word for each end task, which markedly improves performance over just using the top LSTM layer.

Combining the internal states in this manner allows for very rich word representations. Using intrinsic evaluations, we show that the higher-level LSTM states capture context-dependent aspects of word meaning (e.g., they can be used without modification to perform well on supervised

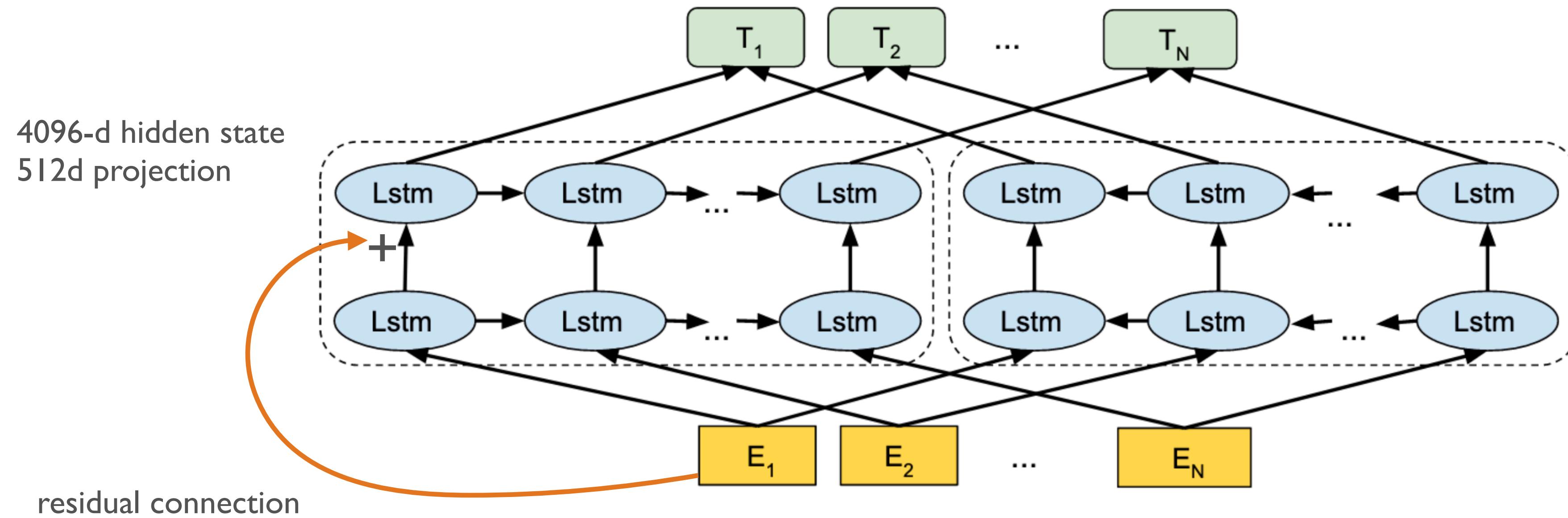
ELMo Model



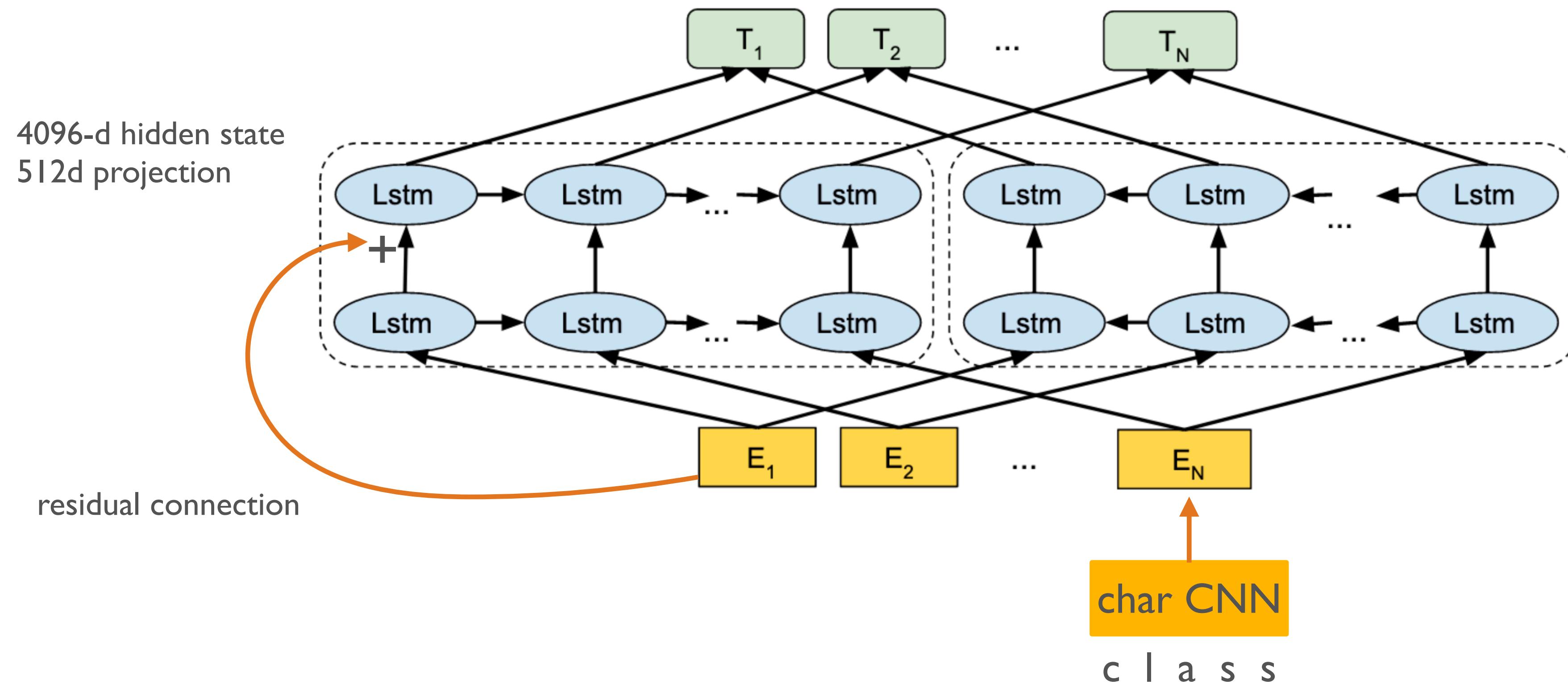
ELMo Model



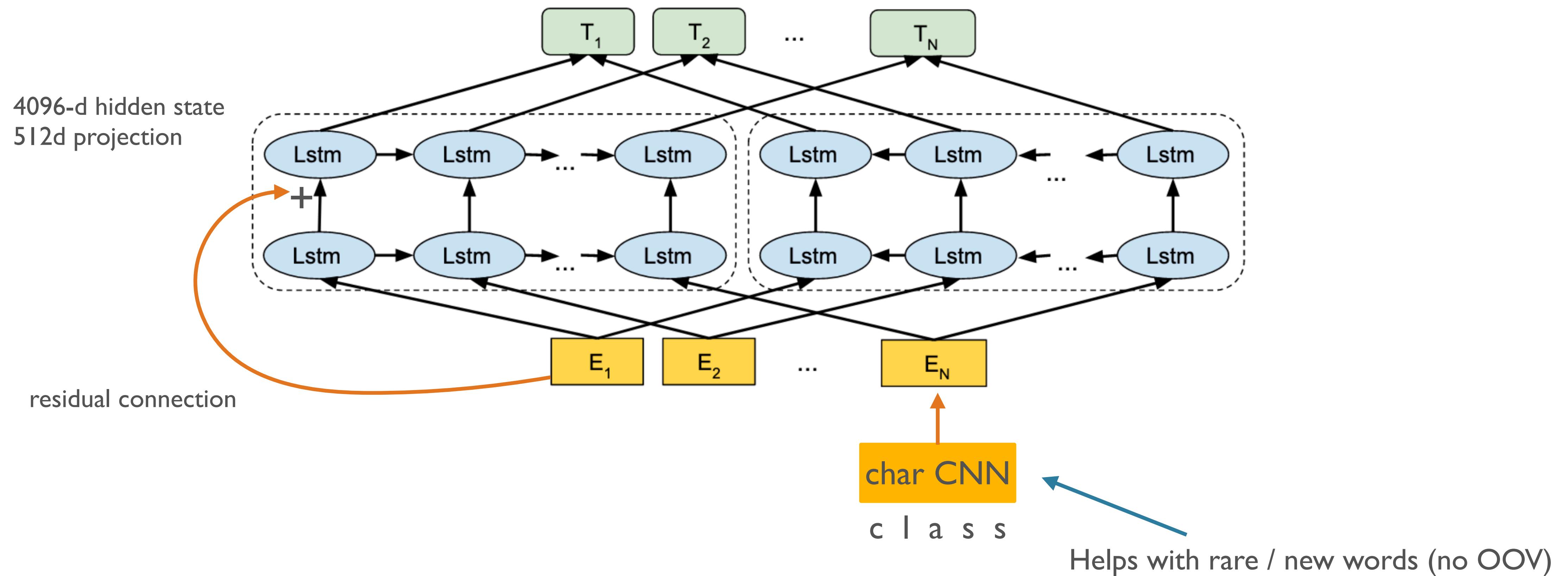
ELMo Model



ELMo Model



ELMo Model



ELMo Training

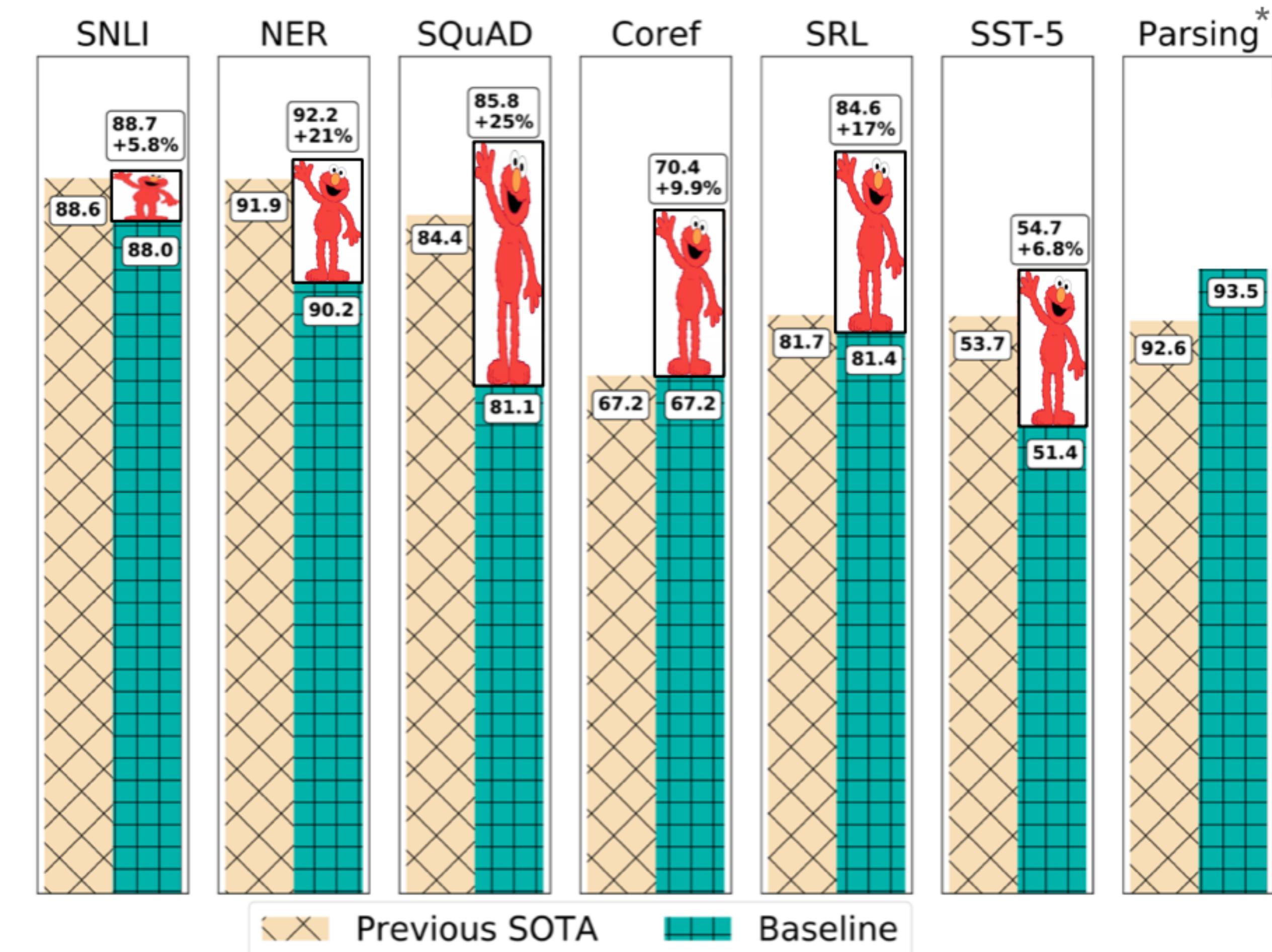
- 10 epochs on 1B Word Benchmark
- NB: not SOTA perplexity even at time of publishing
 - See “Exploring the Limits of Language Modeling” paper
- Regularization:
 - Dropout
 - L2 norm

Deep Contextualized Word Representations

Peters et. al (2018)

- Used in place of other embeddings on multiple tasks:

SQuAD = [Stanford Question Answering Dataset](#)
SNLI = [Stanford Natural Language Inference Corpus](#)
SST-5 = [Stanford Sentiment Treebank](#)



*Kitaev and Klein, ACL 2018 (see also Joshi et al., ACL 2018)

Global vs. Contextual Word Vectors

- Global vectors: one vector per word-type
 - E.g. word2vec, GloVe
 - No difference between e.g. “play” as a verb, noun, or its different senses
- Contextual vectors: one vector per word-occurrence
 - “We saw a really great **play** last week.”
 - “Do you want to **play** basketball tomorrow?”
 - Each *occurrence* gets its own vector representation.

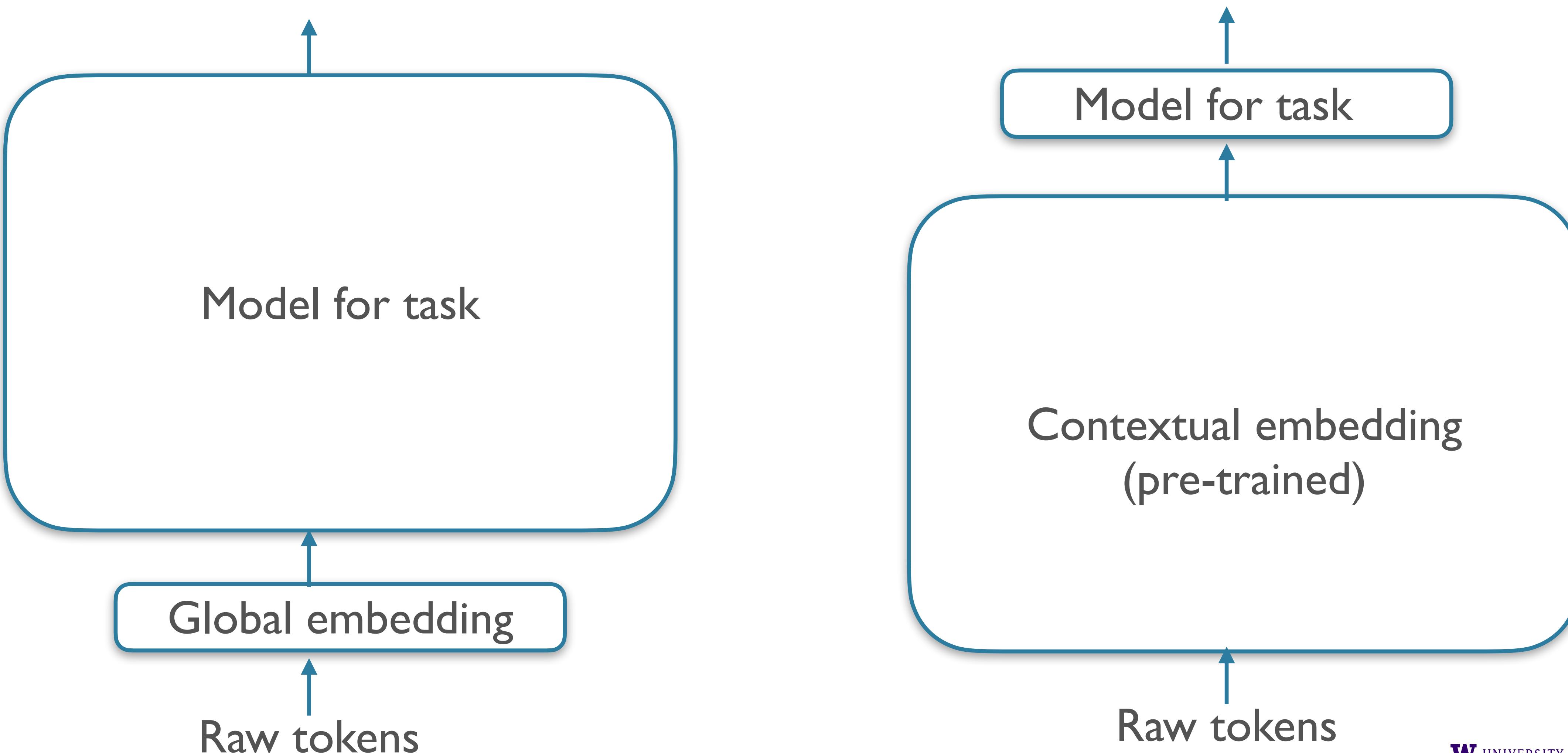
Deep Contextualized Word Representations

Peters et. al (2018)

- Comparison to GloVe:

	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... Olivia De Havilland signed to do a Broadway play for Garson...	Kieffer, the only junior in the group, was commended for his ability to hit in the clutch, as well as his all-round excellent playthey were actors who had been handed fat roles in a successful play , and had talent enough to fill the roles competently, with nice understatement.

Shallow vs Deep Pre-training



Pre-trained Transformers

Paralellizability + Scale

- ULMFiT + ELMo:
 - Demonstrate the value of LM pre-training + transfer learning
 - Noted that there are “virtually unlimited” quantities of data for LM
 - Used bi-LSTMs for the LM
- Concurrently: Transformer paper introduced
- Triggered an explosion in the pretraining approach
- Lack of recurrence —> paralellizability —> scaling up both model size and dataset size

Pre-trained Transformers: Encoder-only

BERT: Bidirectional Encoder Representations from Transformers

[Devlin et al NAACL 2019](#)



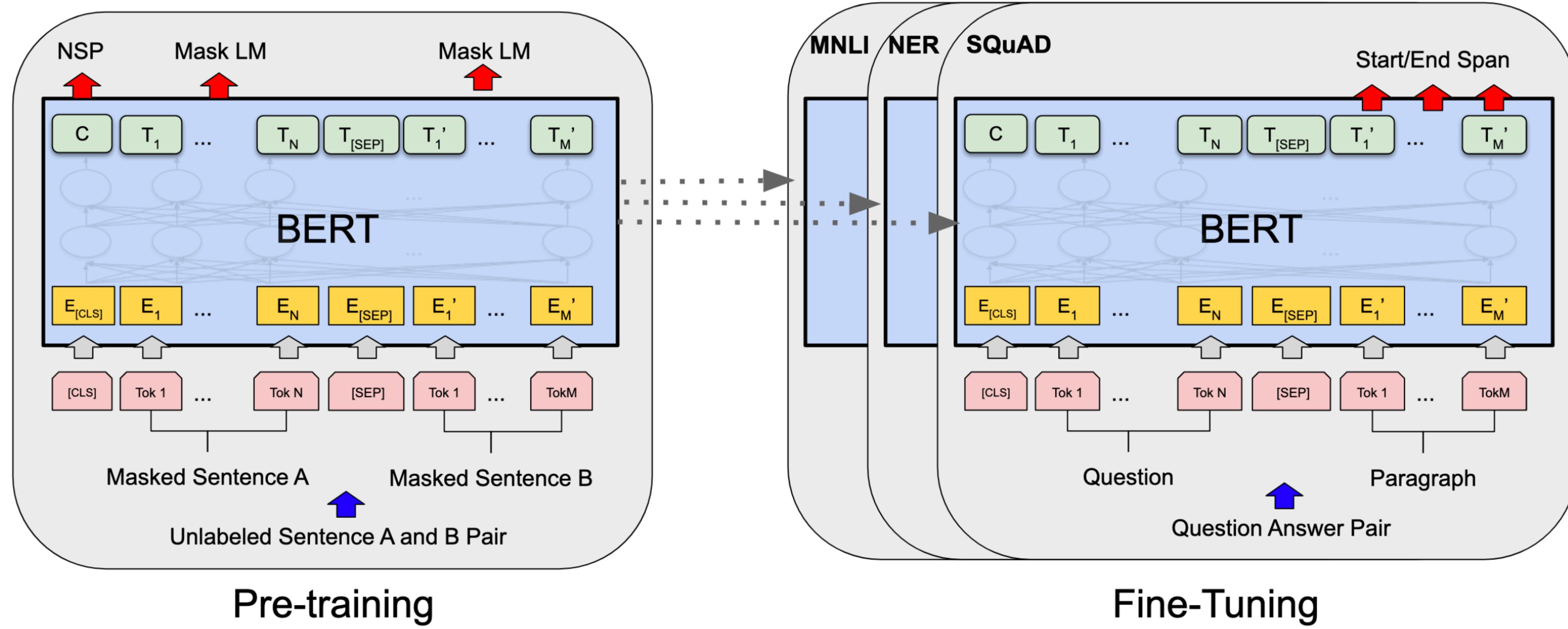
Overview

- Encoder Representations from Transformers: ✓
- Bidirectional:?
 - BiLSTM (ELMo): left-to-right and right-to-left
 - Self-attention: every token can see every other
 - NB: *adirectional* probably a better term
- How do you treat the encoder as an LM (as computing $P(w_t | w_{t-1}, w_{t-2}, \dots, w_1)$)?
 - Don't: modify the task

Masked Language Modeling

- Language modeling: next word prediction
- *Masked Language Modeling* (a.k.a. cloze task): fill-in-the-blank
 - Nancy Pelosi sent the articles of _____ to the Senate.
 - Seattle _____ some snow, so UW was delayed due to _____ roads.
- I.e. $P(w_t | w_{t+k}, w_{t+(k-1)}, \dots, w_{t+1}, w_{t-1}, \dots, w_{t-(m+1)}, w_{t-m})$
 - (very similar to CBOW: continuous bag of words from word2vec)
- Auxiliary training task: next sentence prediction.
 - Given sentences A and B, binary classification: did B follow A in the corpus or not?

Schematically



Some details

Some details

- BASE model:
 - 12 Transformer Blocks
 - Hidden vector size: 768
 - Attention heads / layer: 12
 - Total parameters: 110M

Some details

- BASE model:
 - 12 Transformer Blocks
 - Hidden vector size: 768
 - Attention heads / layer: 12
 - Total parameters: 110M
- LARGE model:
 - 24 Transformer Blocks
 - Hidden vector size: 1024
 - Attention heads / layer: 16
 - Total parameters: 340M

Some details

- BASE model:
 - 12 Transformer Blocks
 - Hidden vector size: 768
 - Attention heads / layer: 12
 - Total parameters: 110M
- LARGE model:
 - 24 Transformer Blocks
 - Hidden vector size: 1024
 - Attention heads / layer: 16
 - Total parameters: 340M

this is the first work to demonstrate convincingly that scaling to extreme model sizes also leads to large improvements on very small scale tasks, provided that the model has been sufficiently pre-trained. Peters et al. (2018b) presented

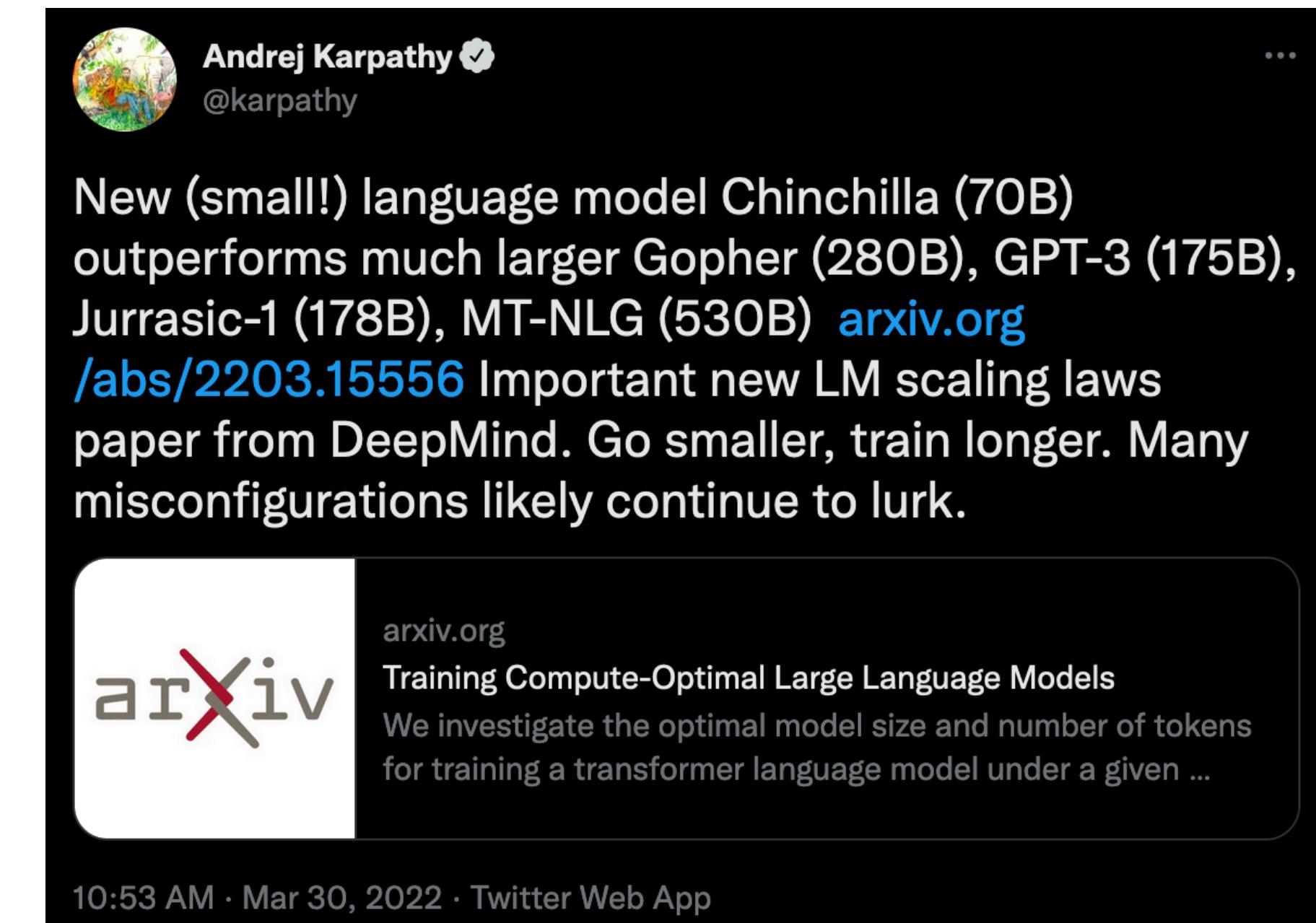
Some details

- BASE model:
 - 12 Transformer Blocks
 - Hidden vector size: 768
 - Attention heads / layer: 12
 - Total parameters: 110M
- LARGE model:
 - 24 Transformer Blocks
 - Hidden vector size: 1024
 - Attention heads / layer: 16
 - Total parameters: 340M

this is the first work to demonstrate convincingly that scaling to extreme model sizes also leads to large improvements on very small scale tasks, provided that the model has been sufficiently pre-trained. Peters et al. (2018b) presented

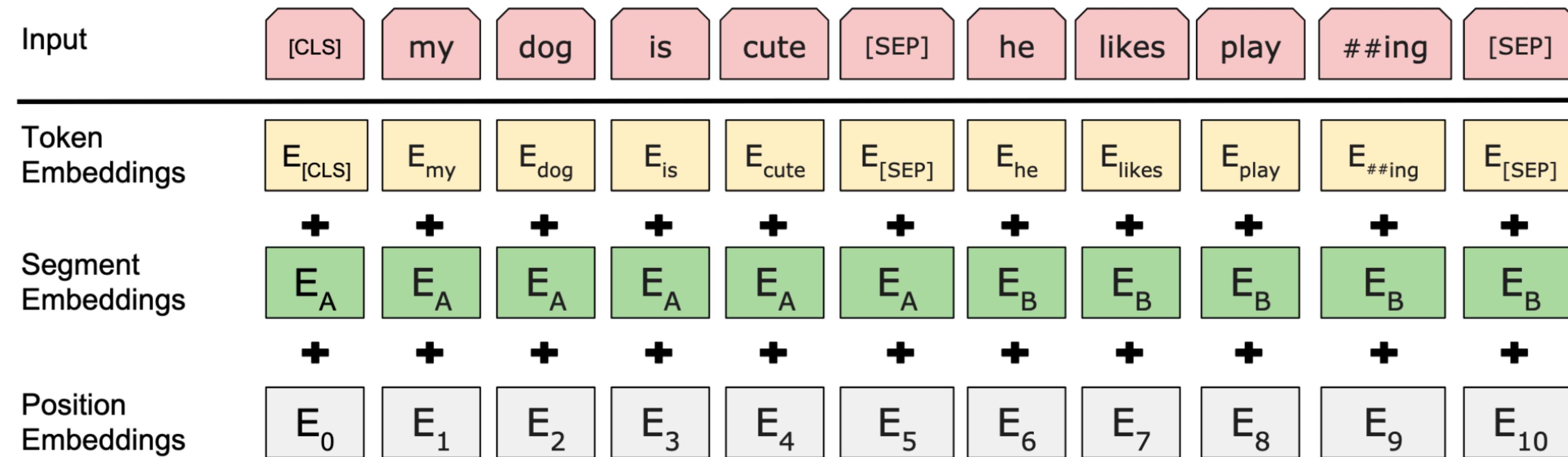
Some details

- BASE model:
 - 12 Transformer Blocks
 - Hidden vector size: 768
 - Attention heads / layer: 12
 - Total parameters: 110M
- LARGE model:
 - 24 Transformer Blocks
 - Hidden vector size: 1024
 - Attention heads / layer: 16
 - Total parameters: 340M

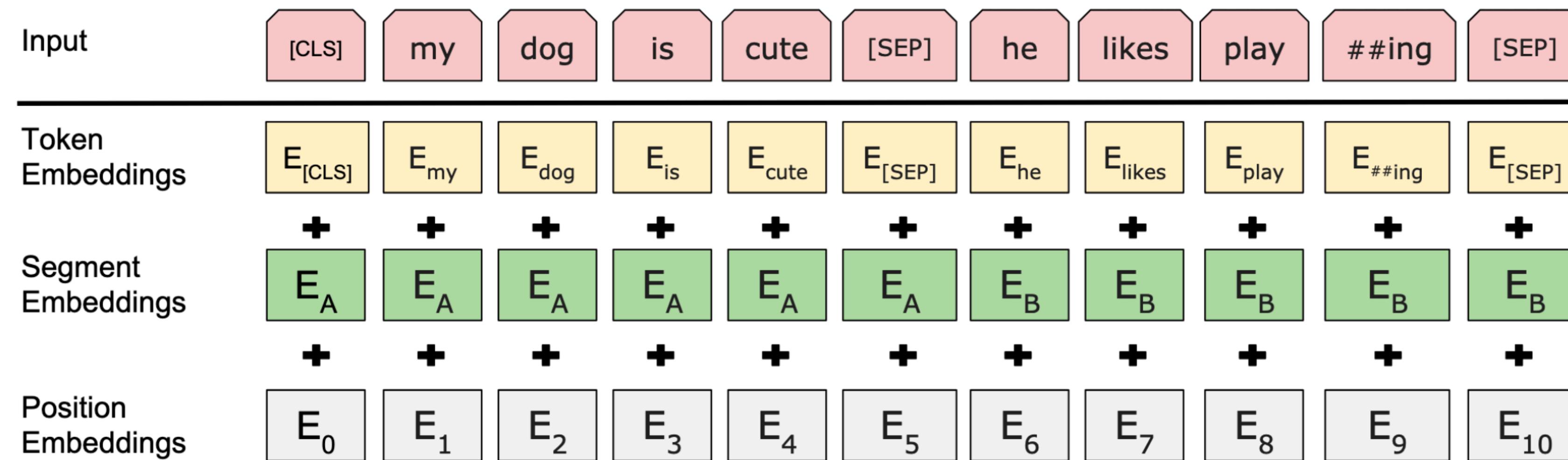


leads to large improvements on very small scale tasks, provided that the model has been sufficiently pre-trained. Peters et al. (2018b) presented

Input Representation

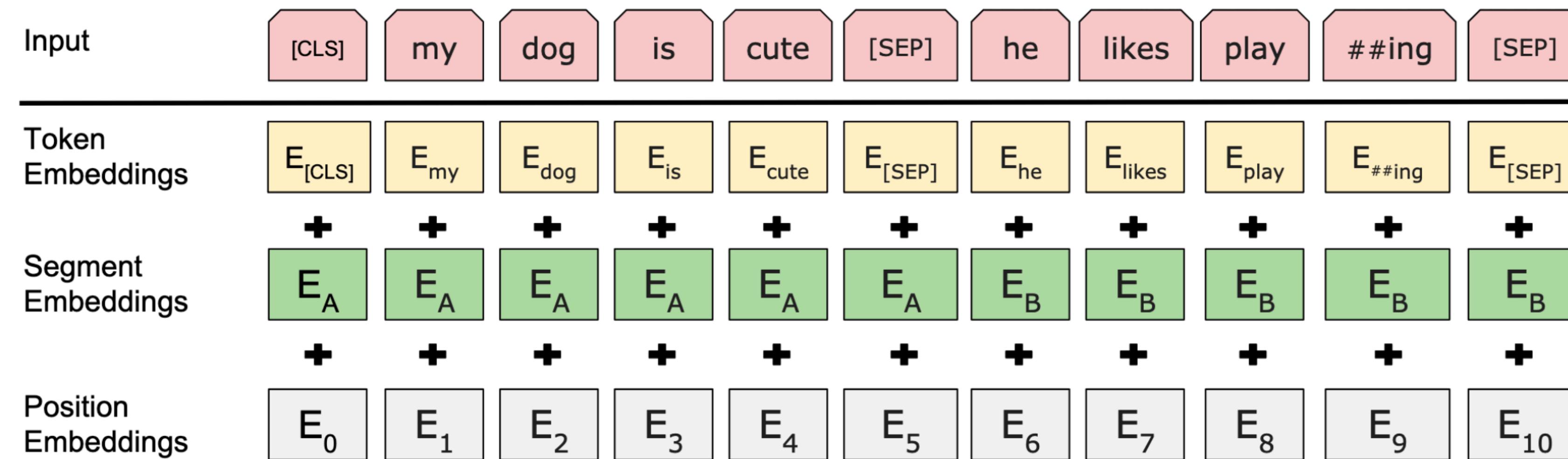


Input Representation



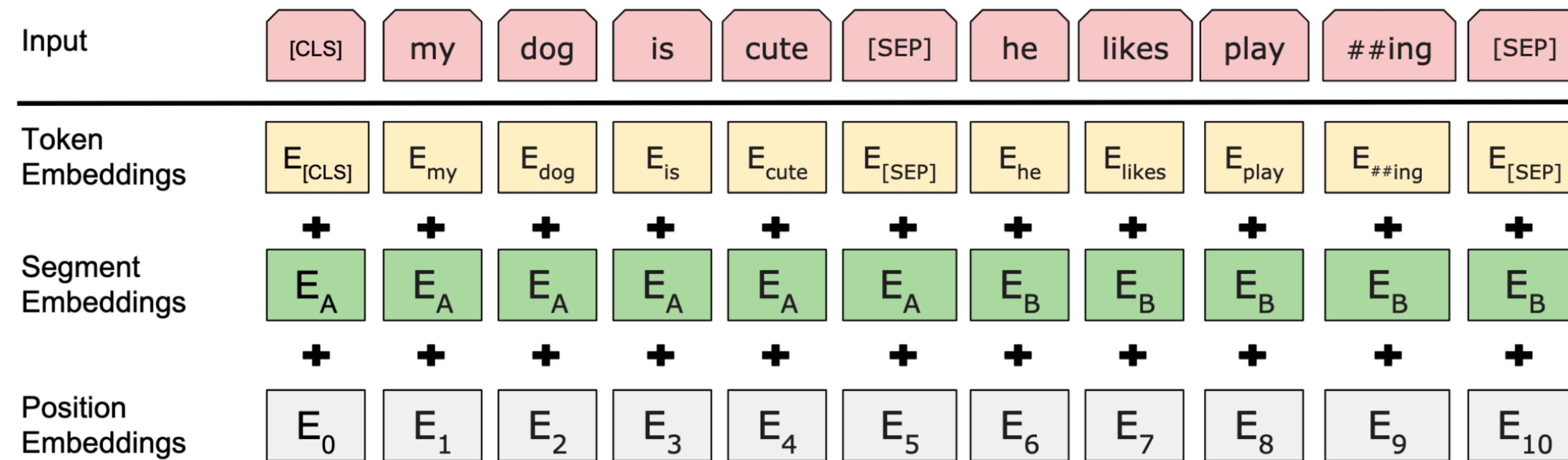
- [CLS], [SEP]: special tokens

Input Representation



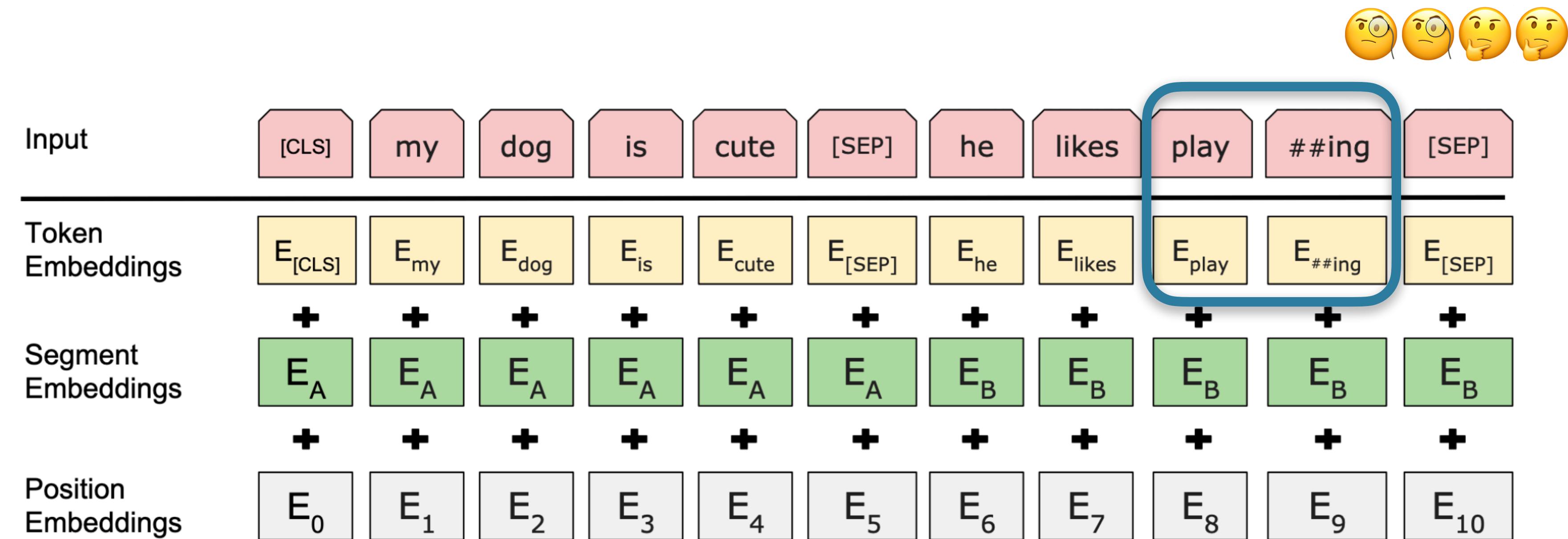
- [CLS], [SEP]: special tokens
- Segment: is this a token from sentence A or B?

Input Representation



- [CLS], [SEP]: special tokens
- Segment: is this a token from sentence A or B?
- Position embeddings: provide position in sequence (*learned* in this case, not fixed)

Input Representation



- [CLS], [SEP]: special tokens
- Segment: is this a token from sentence A or B?
- Position embeddings: provide position in sequence (*learned* in this case, not fixed)

Training Details

- BooksCorpus (800M words) + Wikipedia (2.5B)
- Masking the input text. 15% of all tokens are chosen. Then:
 - 80% of the time: replaced by designated '[MASK]' token
 - 10% of the time: replaced by random token
 - 10% of the time: unchanged
- Loss is cross-entropy of the prediction at the masked positions.
- Max seq length: 128 tokens for first 90%, 512 tokens for final 10%
- 1M training steps, batch size 256 = 4 days on 4 or 16 TPUs

Initial Results

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
BERT _{BASE}	84.6/83.4	71.2	90.5	93.5	52.1	85.8	88.9	66.4	79.6
BERT _{LARGE}	86.7/85.9	72.1	92.7	94.9	60.5	86.5	89.3	70.1	82.1

Ablations

Hyperparams			Dev Set Accuracy			
#L	#H	#A	LM (ppl)	MNLI-m	MRPC	SST-2
3	768	12	5.84	77.9	79.8	88.4
6	768	3	5.24	80.6	82.2	90.7
6	768	12	4.68	81.9	84.8	91.3
12	768	12	3.99	84.4	86.7	92.9
12	1024	16	3.54	85.7	86.9	93.3
24	1024	16	3.23	86.6	87.8	93.7

- Not a given (depth doesn't help ELMo); possibly a difference between fine-tuning vs. feature extraction

Tasks	Dev Set				
	MNLI-m (Acc)	QNLI (Acc)	MRPC (Acc)	SST-2 (Acc)	SQuAD (F1)
BERT _{BASE}	84.4	88.4	86.7	92.7	88.5
No NSP	83.9	84.9	86.5	92.6	87.9
LTR & No NSP	82.1	84.3	77.5	92.1	77.8
+ BiLSTM	82.1	84.1	75.7	91.6	84.9

- Many more variations to explore

Other Prominent Encoders

- RoBERTa: robustly optimized BERT approach
 - BERT was very *under-trained*: give it more data, train it longer [keep model the same otherwise]
 - Good default encoder
- ELECTRA: replace Masked Language Modeling with “replaced token detection”, trains just as well with much less data
- SpanBERT: mask out entire *spans* instead of single tokens
- DeBERTa: disentangled attention and novel position encoding

Limitation of Encoders

- No left-to-right modeling assumption
- Good for NLU (understanding/comprehension) tasks
- Does not straightforwardly *generate* text

Next Time

- Pre-training + FT, cont.
 - Decoder
 - Encoder-decoder
 - Risks
 - Accessing / using pre-trained LMs
 - In-context learning