Language Models with Transformers

Chenguang Wang, Mu Li, Alexander J. Smola

Amazon Web Services

Background

Language Model (LM)

- Predict what word comes next

Start to learn English

Language Model (LM)

- Predict what word comes next
- Useful in many NLP applications

Start to learn <u>English</u>

Language Model (LM)

- Predict what word comes next
- Useful in many NLP applications

Start to learn <u>English</u>

Learn to start <u>business</u>

Word order matters!

- Many NLP problems share similar definition

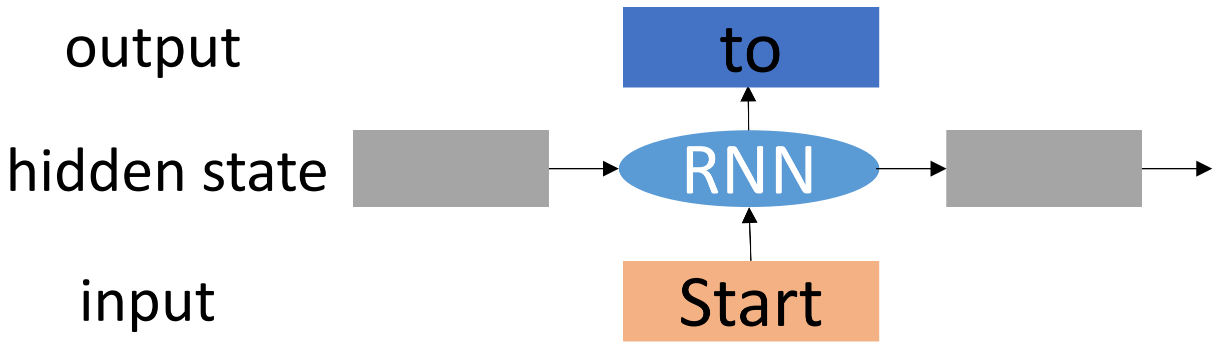
Language Model with RNNs

- RNN uses one-hot encoding

input Start

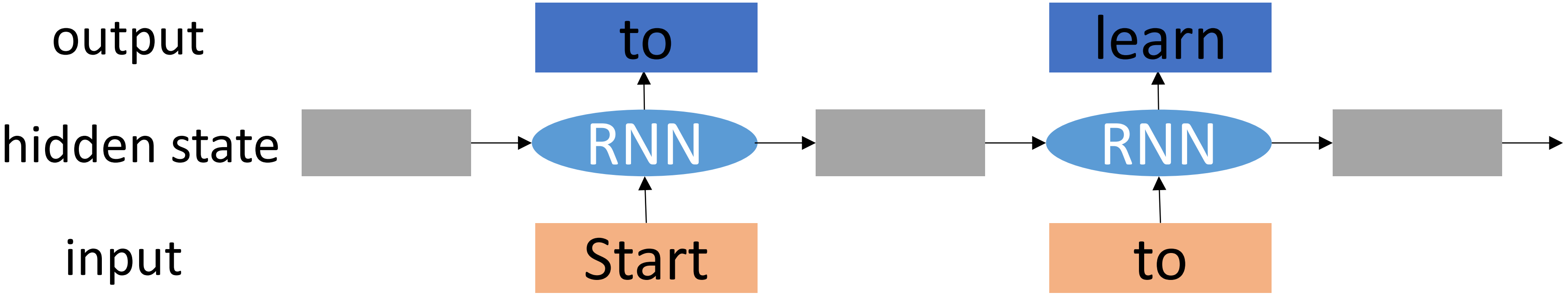
Language Model with RNNs

- RNN models the word order in hidden state



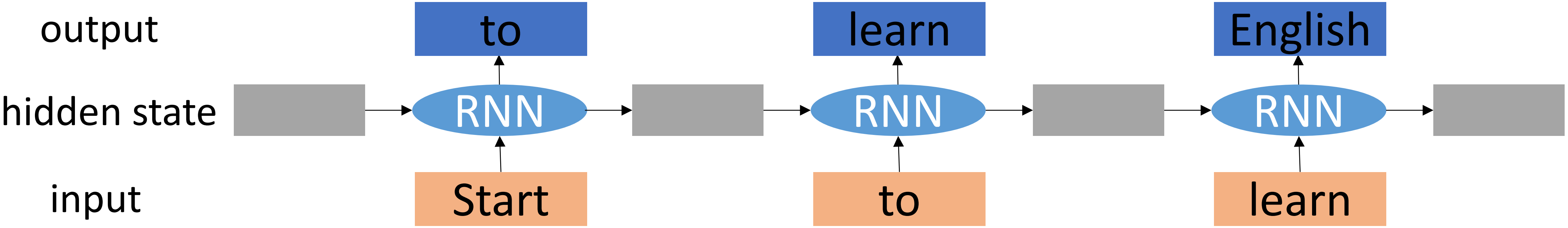
Language Model with RNNs

- RNN models the word order in hidden state

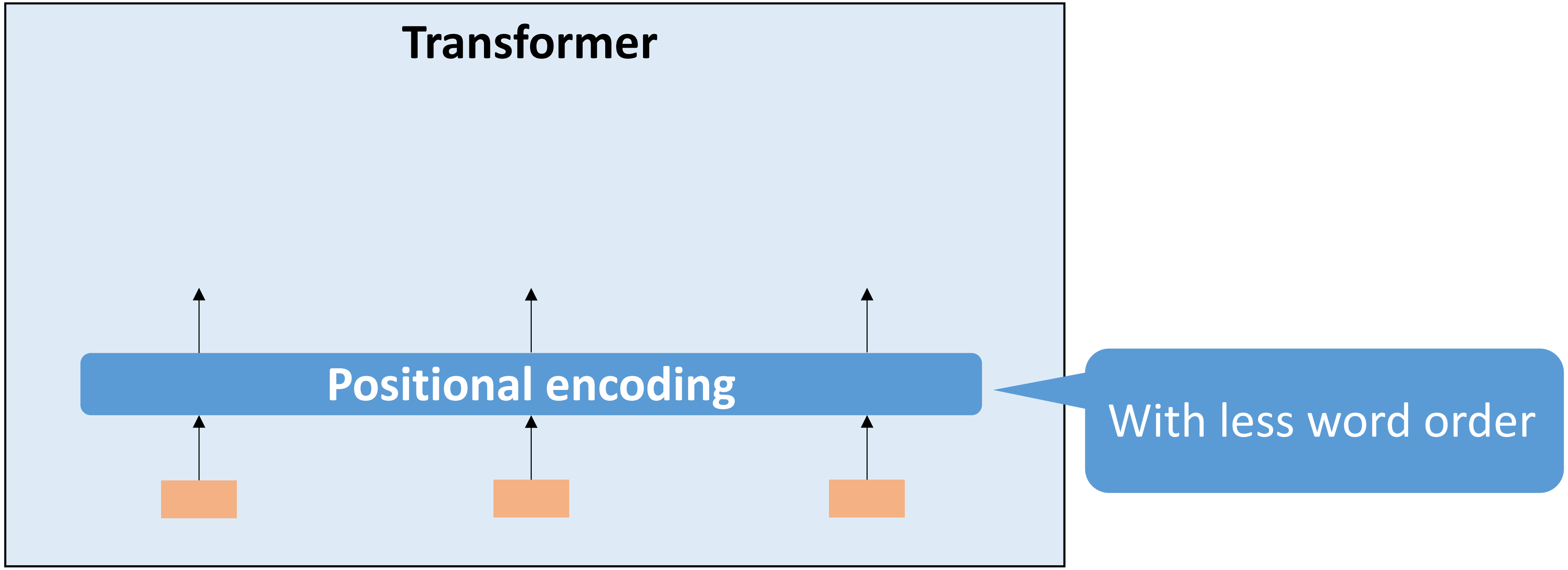


Language Model with RNNs

- RNN models the word order in hidden state

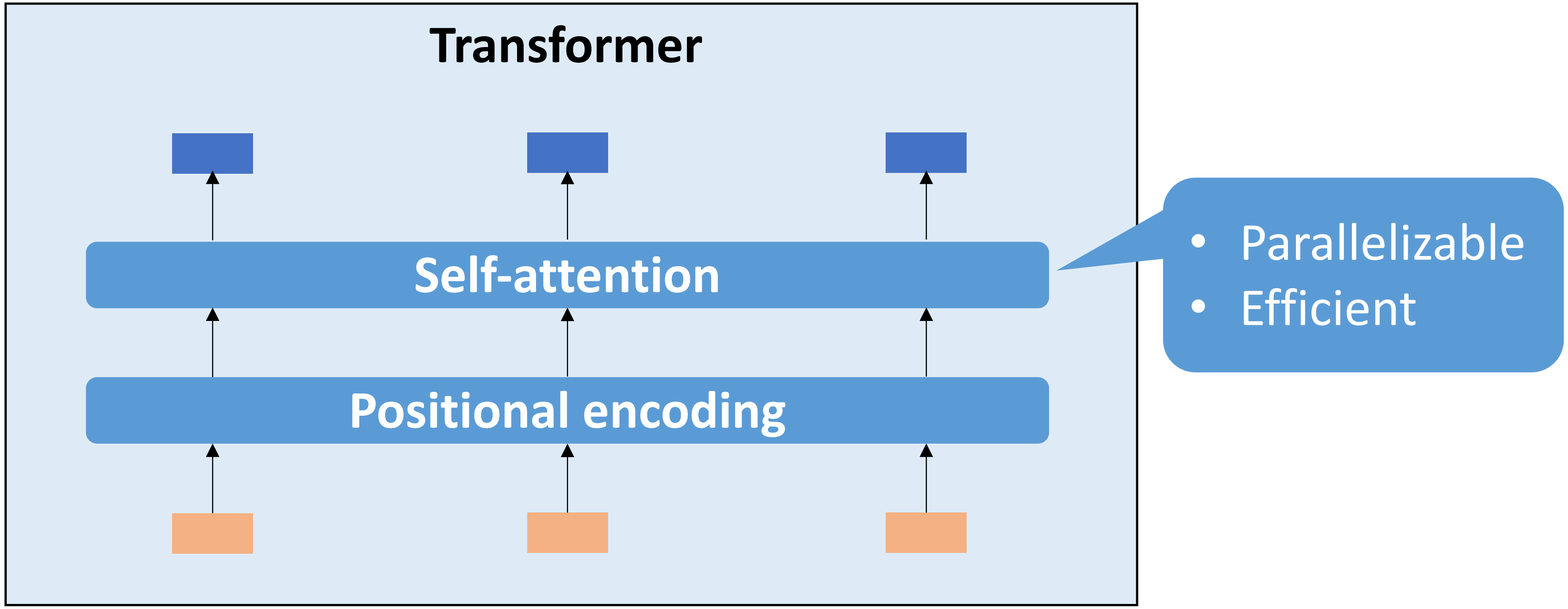


SOTA NLP with Transformers



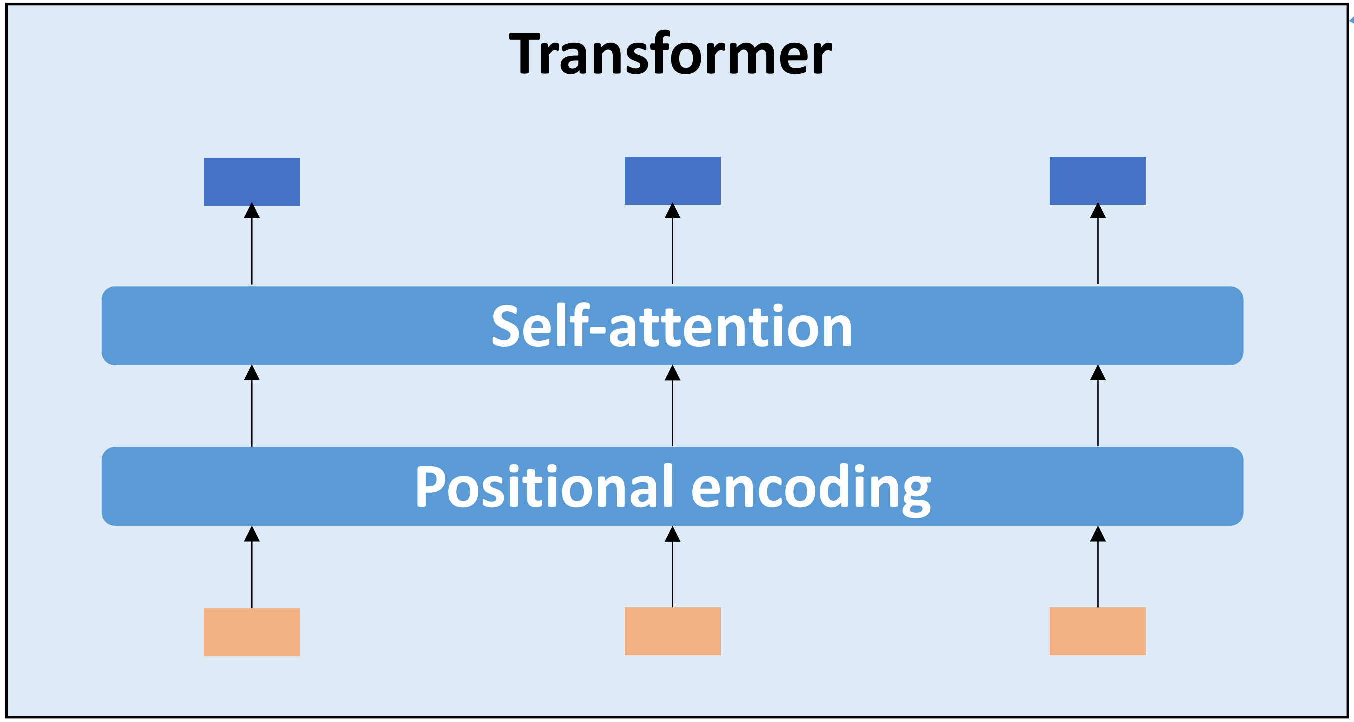
Other components are omitted
for simplicity [Devlin, Jacob, et al 2018]

SOTA NLP with Transformers



Other components are omitted
for simplicity [Devlin, Jacob, et al 2018]

SOTA NLP with Transformers

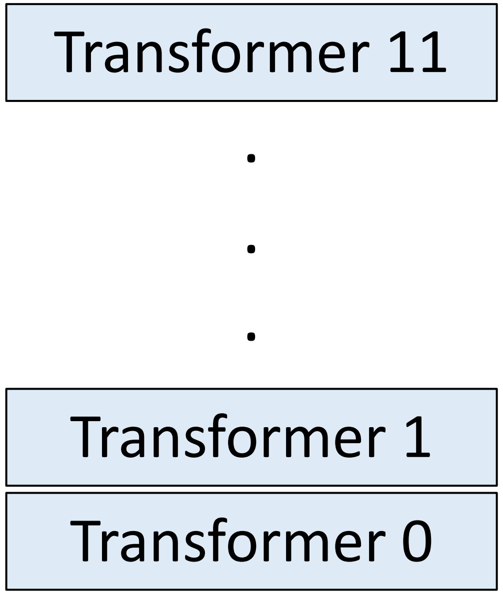


- **With less word order**
- Parallelizable
- Efficient

- **With word order**
- Sequential
- Less efficient

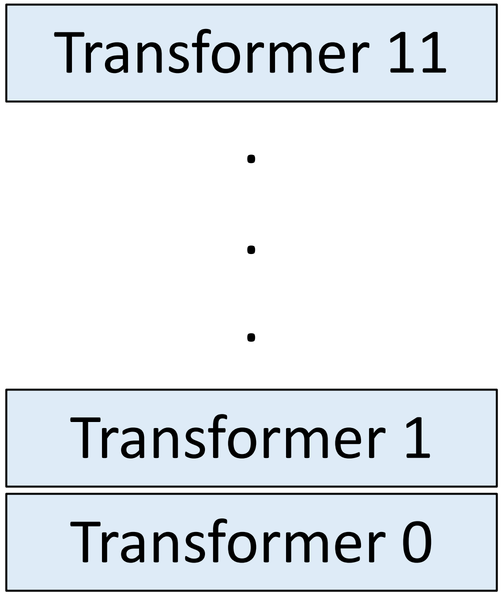
Other components are omitted
for simplicity [Devlin, Jacob, et al 2018]

SOTA NLP with Transformers



- BERT: a stack of 12 (or 24) Transformer blocks

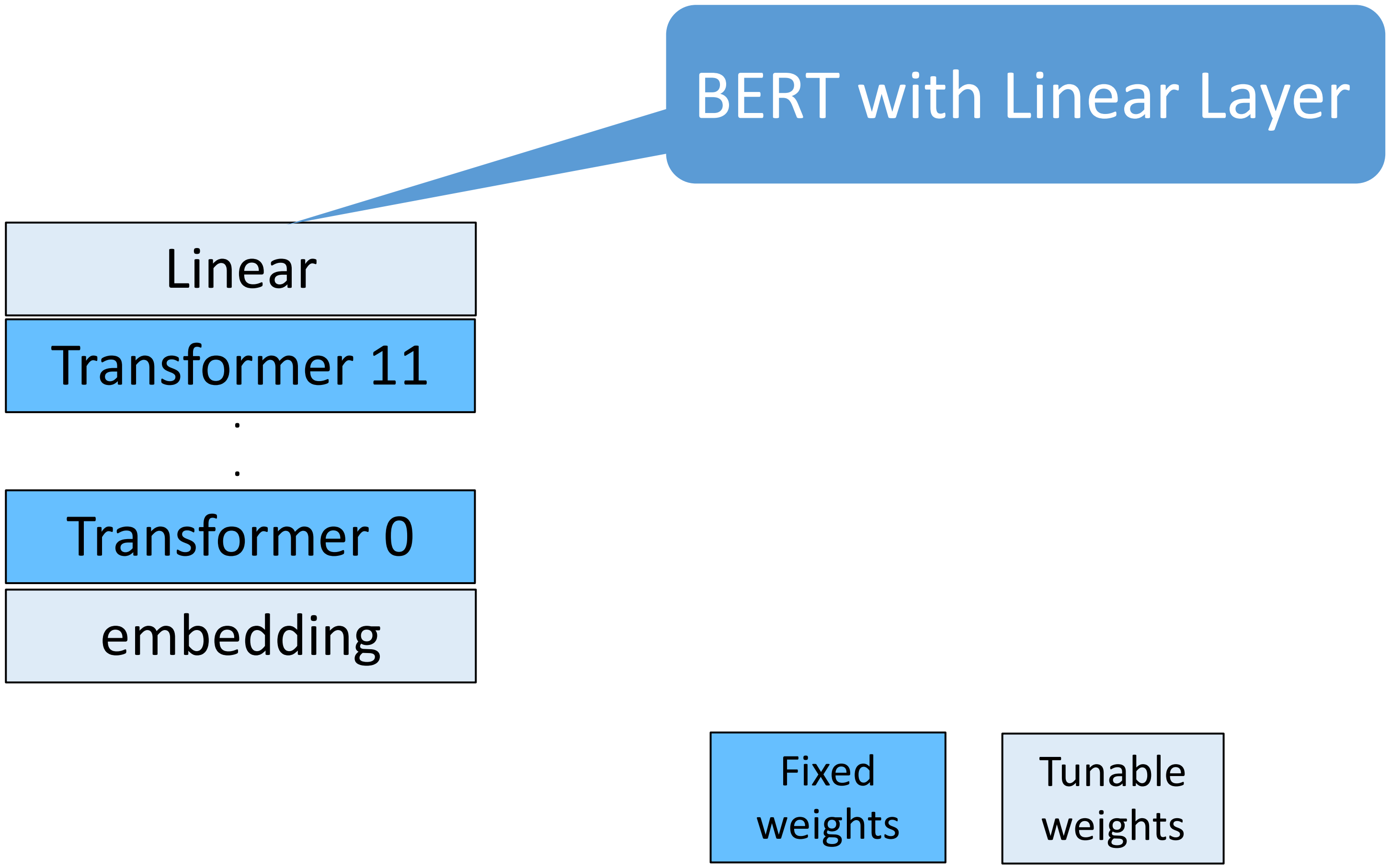
SOTA NLP with Transformers



- BERT: a stack of 12 (or 24) Transformer blocks
- Trained on large language model datasets
  - Full training cost in excess of $10,000 (16 TPU, 4 days)
- Achieved SOTA results on 11 NLP applications
  - Sentence level tasks: care less about word order

Approach:
Make Best Use of BERT for Language Model

LM: Adapted BERT



LM 1: Adapted BERT with Fixed Weights

Linear
Transformer 11
.
.
Transformer 0
embedding

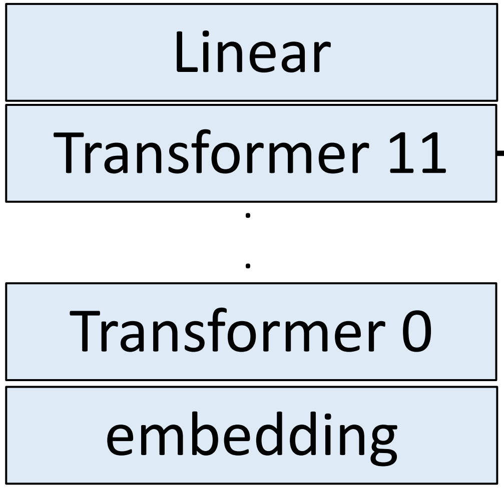
Model	Test PPL
BERT	69.32
RNN	42.25

Only moderate results
(the Lower, the Better)

Fixed weights

Tunable weights

LM 2: Adapted BERT with All Weights



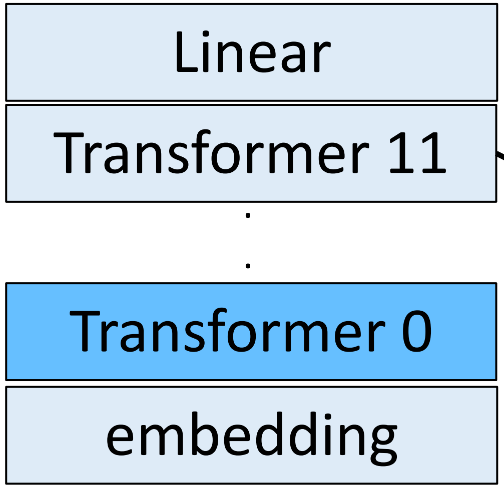
Model	Test PPL
BERT	69.32
BERT-All	67.43
RNN	42.25

Overfitting

Fixed weights

Tunable weights

LM 3: Adapted BERT with Partial Weights



Model	Test PPL
BERT	69.32
BERT-All	67.43
BERT-Subset	**40.56**
RNN	42.25

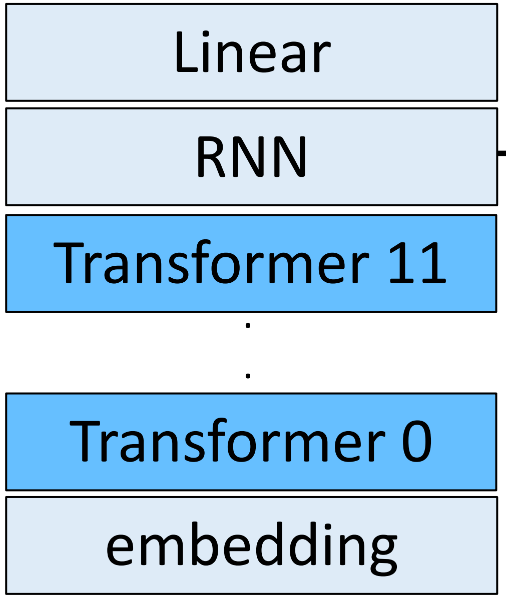
Fix a subset of weights is promising

However, enumerating is not feasible

Fixed weights

Tunable weights

LM 4: Adapted BERT with RNN



Model	Test PPL
BERT	69.32
BERT-RNN	**41.64**
RNN	42.25

Fixed weights

Tunable weights

Add RNN to capture word order is promising

However, enumerating is not feasible

- Where
- How many

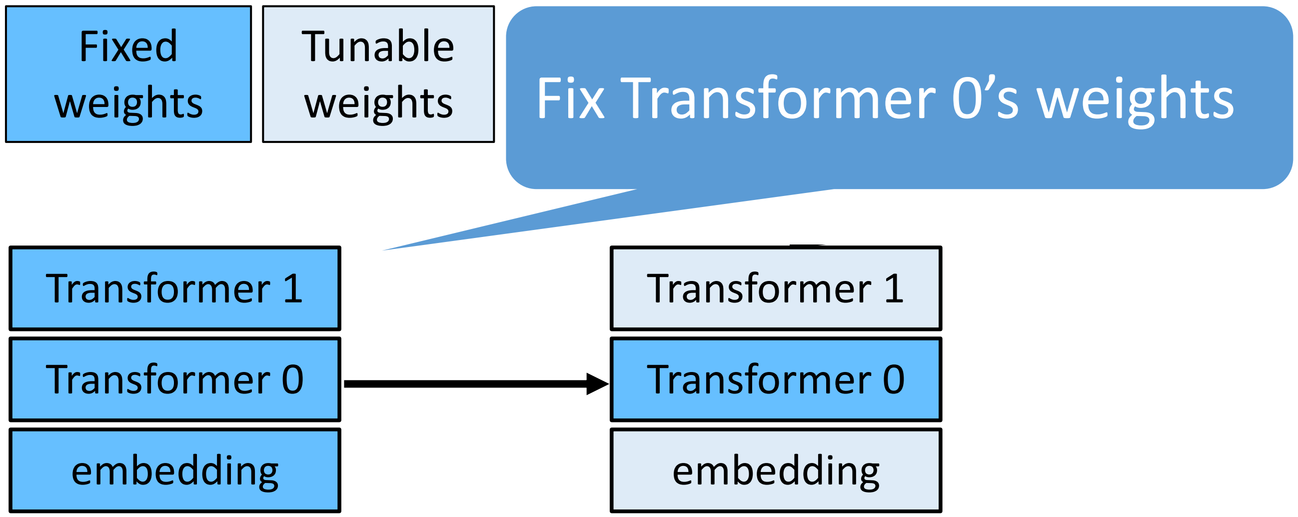
Where to add the RNN layers?

Which layer's pre-trained weights should be fixed?

Where to add the RNN layers?

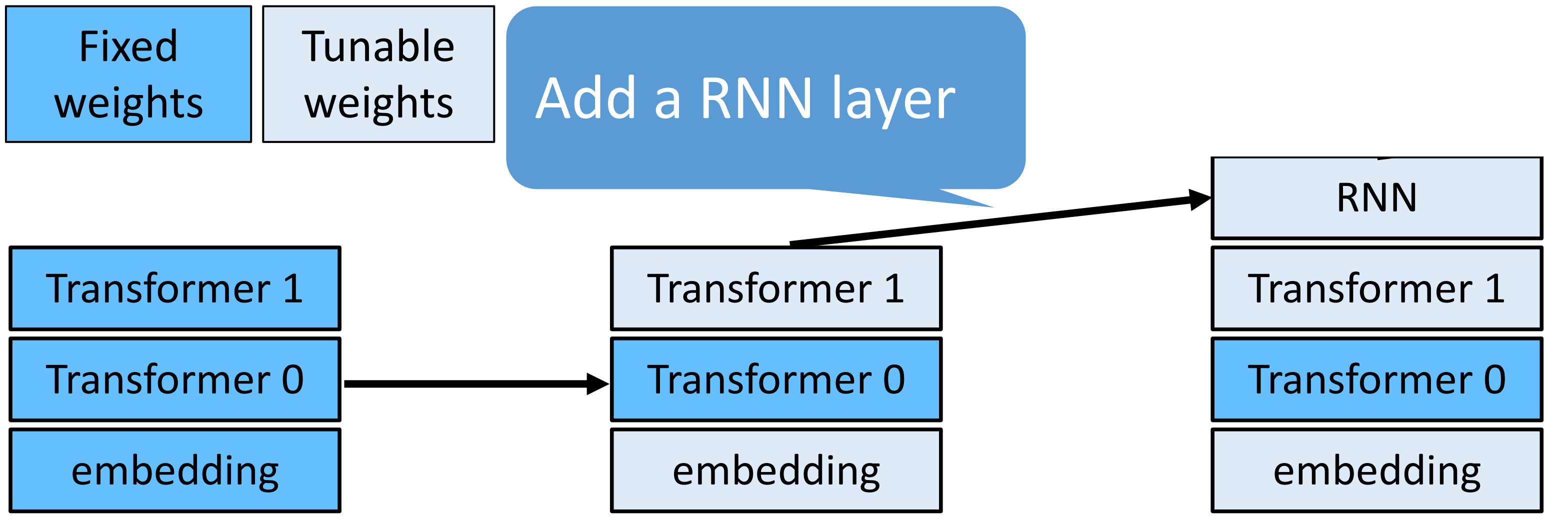
Coordinate Architecture Search (CAS)

- Step 1: Choose a layer’s weights to fix



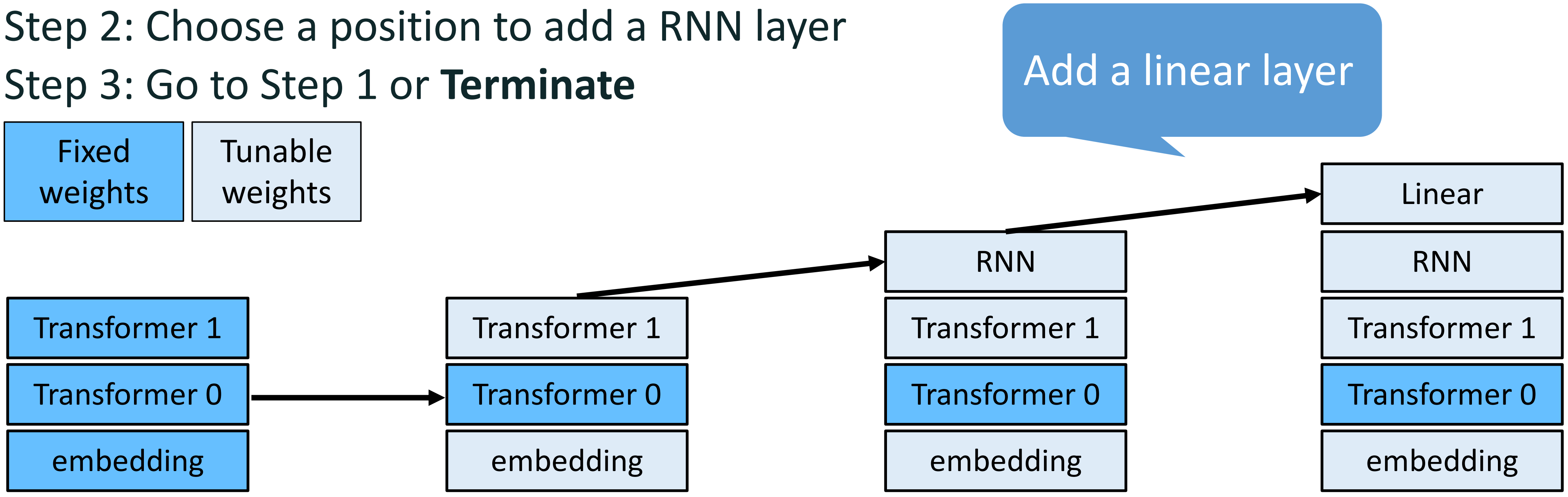
Coordinate Architecture Search (CAS)

- Step 1: Choose a layer's weights to fix
- Step 2: Choose a position to add a RNN layer



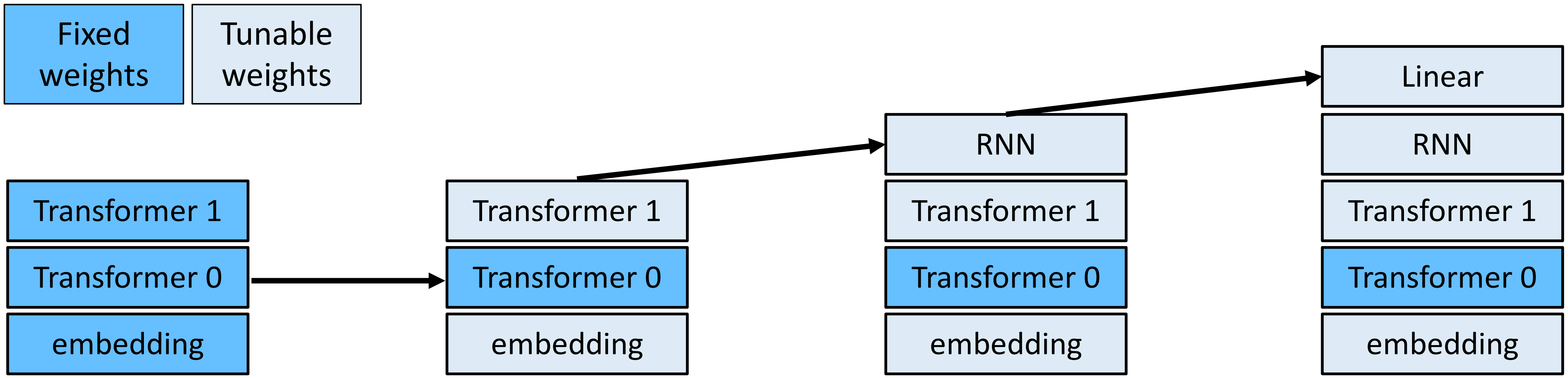
Coordinate Architecture Search (CAS)

- Step 1: Choose a layer's weights to fix
- Step 2: Choose a position to add a RNN layer
- Step 3: Go to Step 1 or **Terminate**



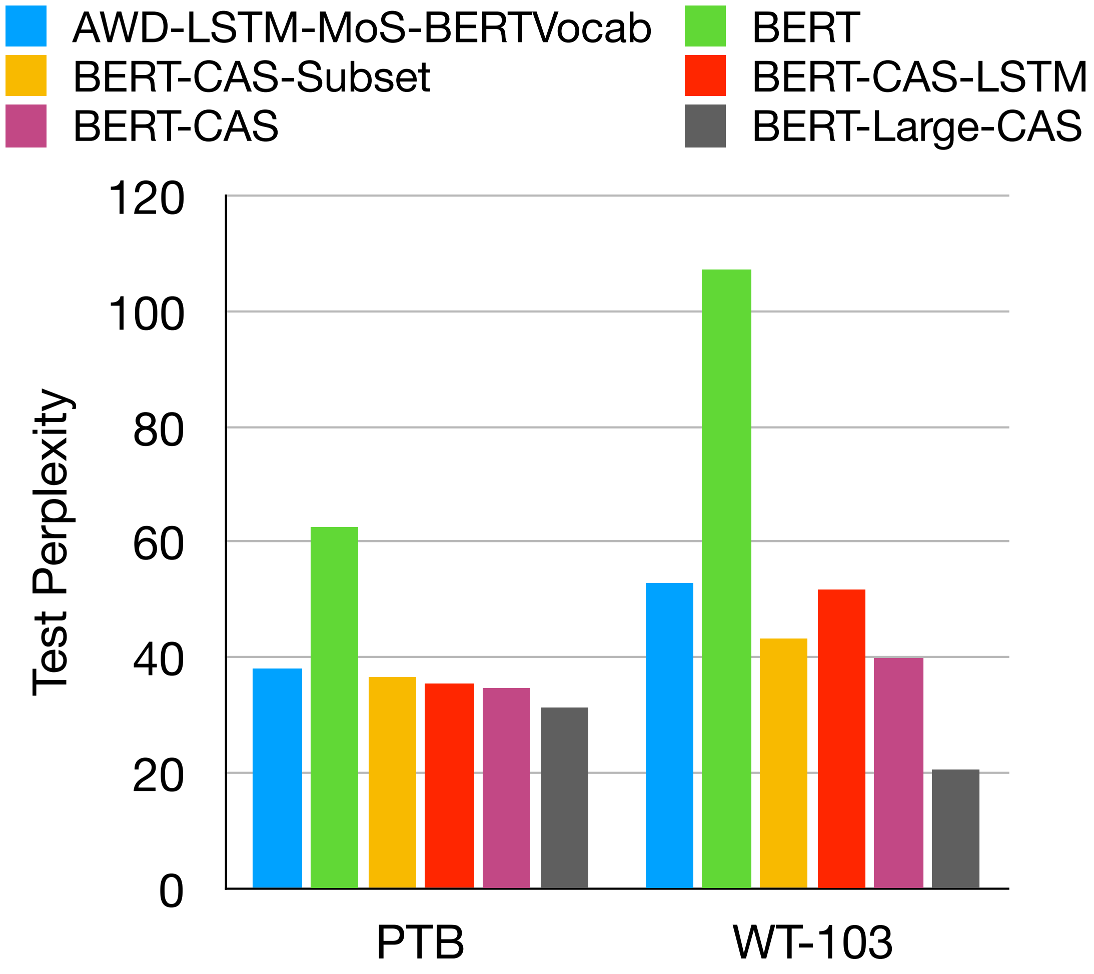
Coordinate Architecture Search (CAS)

- Step 1: Choose a layer's weights to fix
- Step 2: Choose a position to add a RNN layer
- Step 3: Go to Step 1 or Terminate

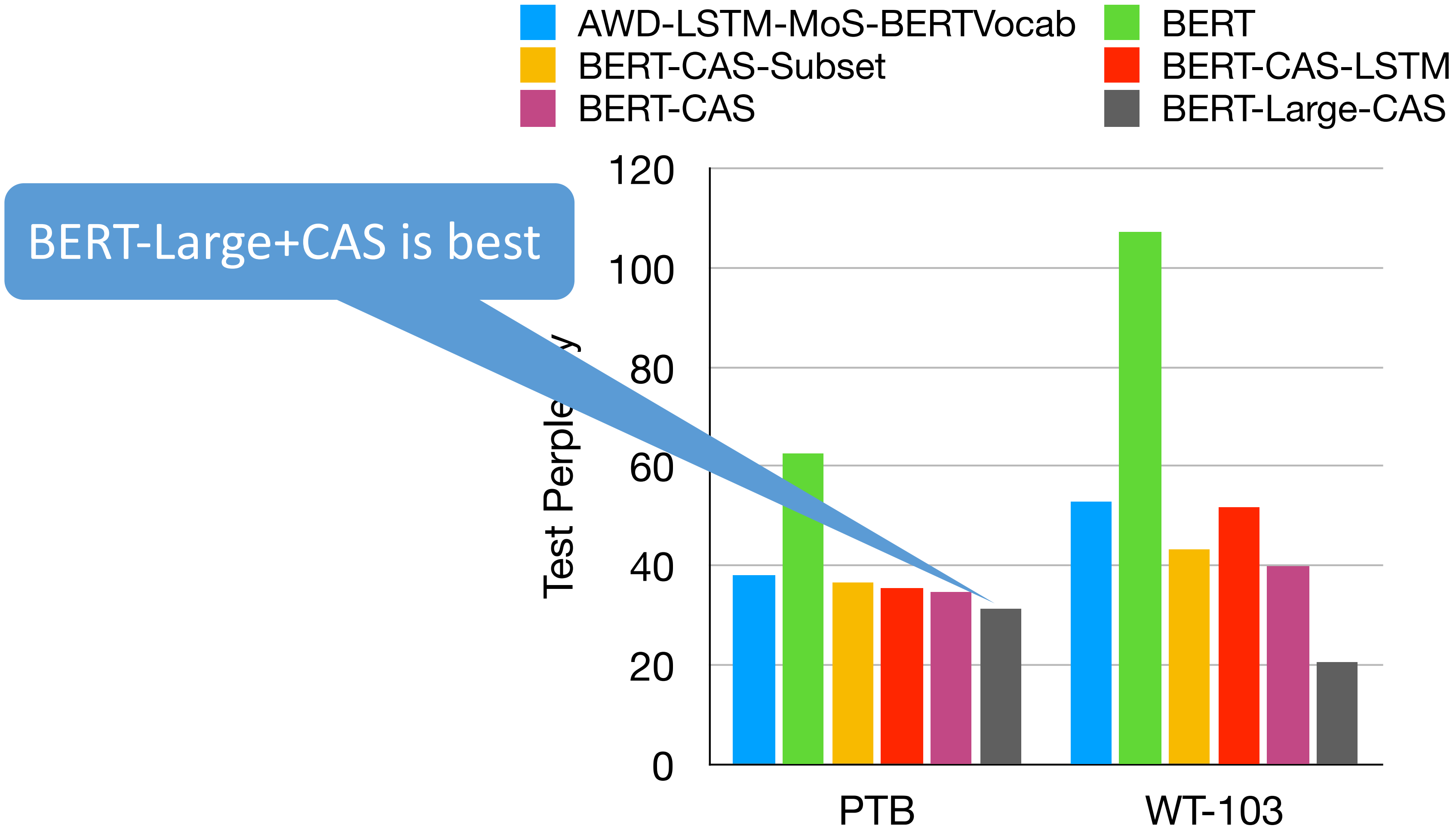


- Greedy strategy: fine-tune the resulting BERT and keep the best

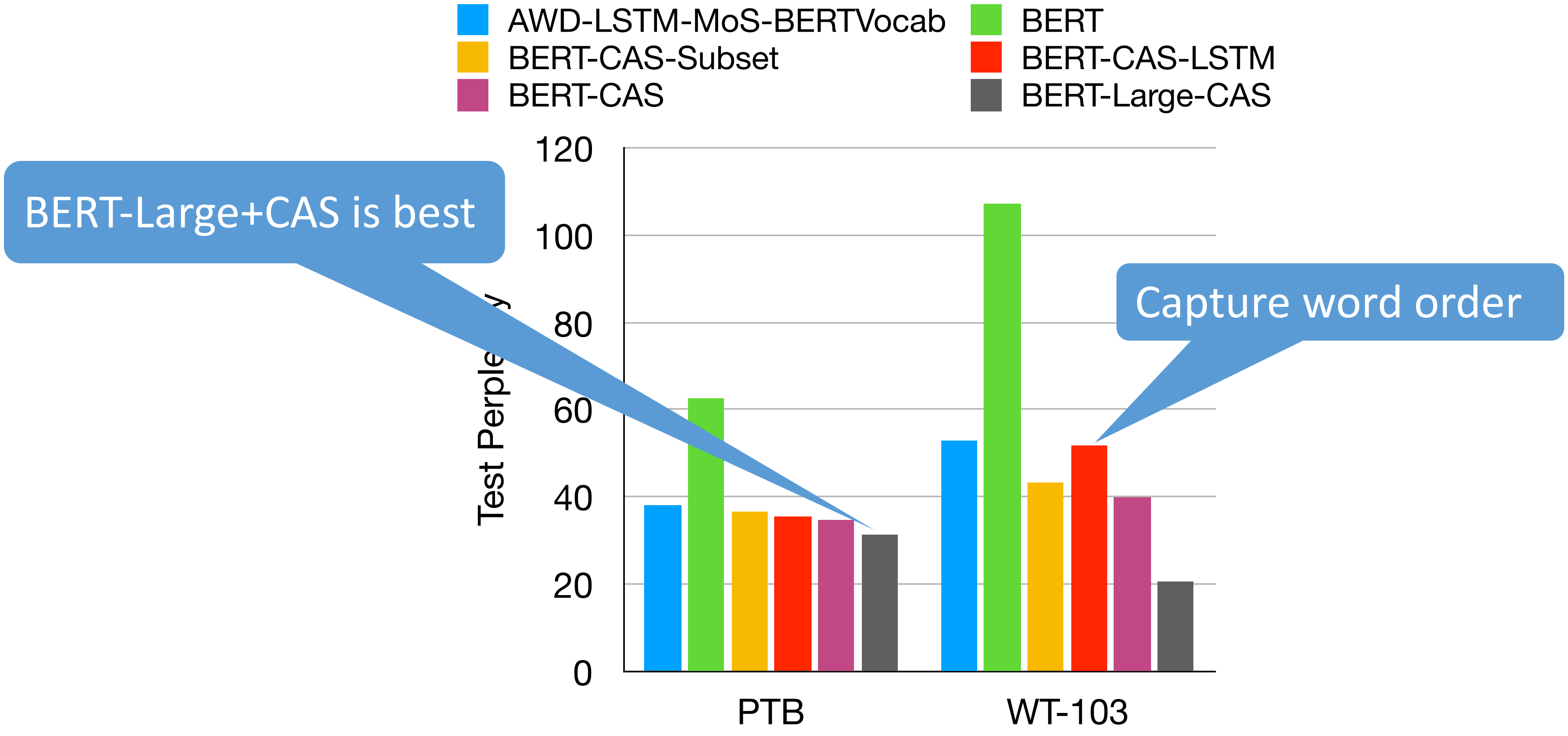
Best LM: Adapted BERT with CAS



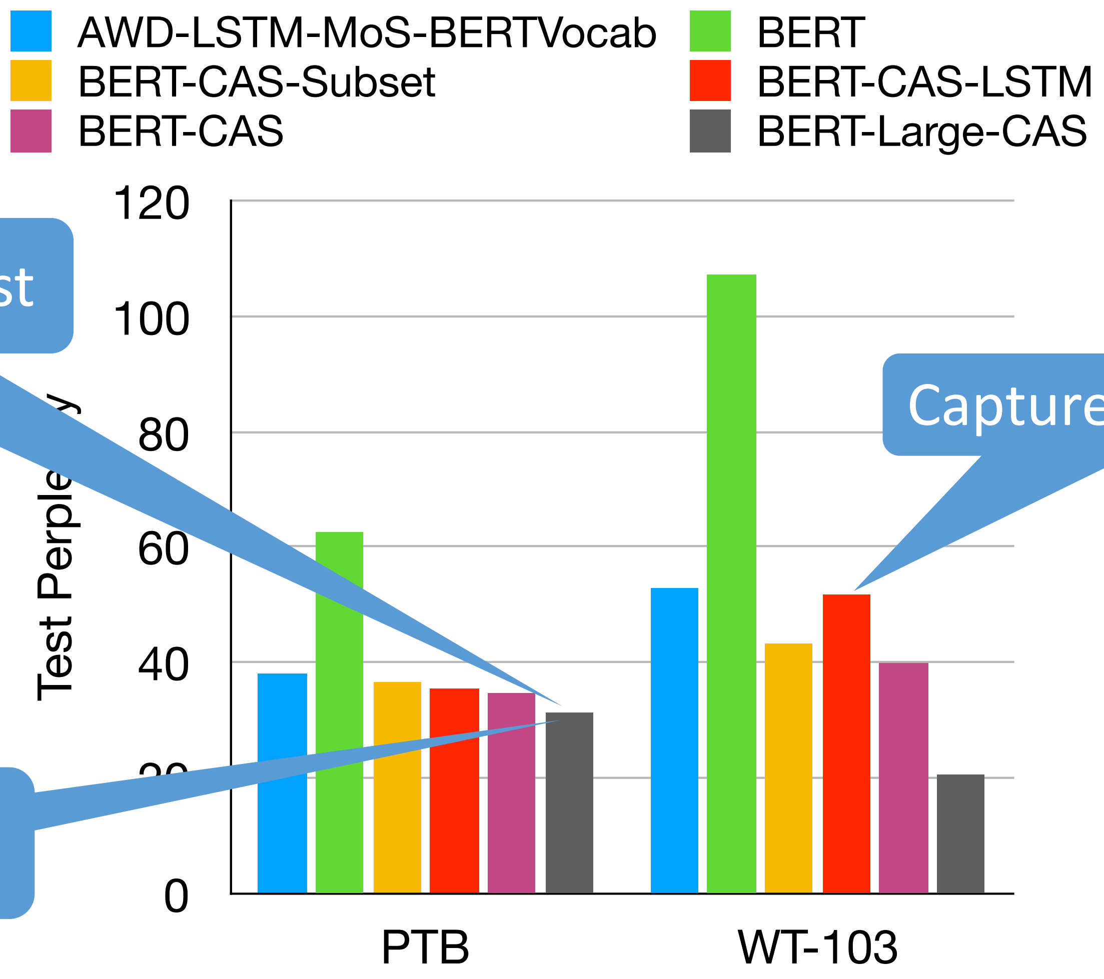
Best LM: Adapted BERT with CAS



Best LM: Adapted BERT with CAS



Best LM: Adapted BERT with CAS

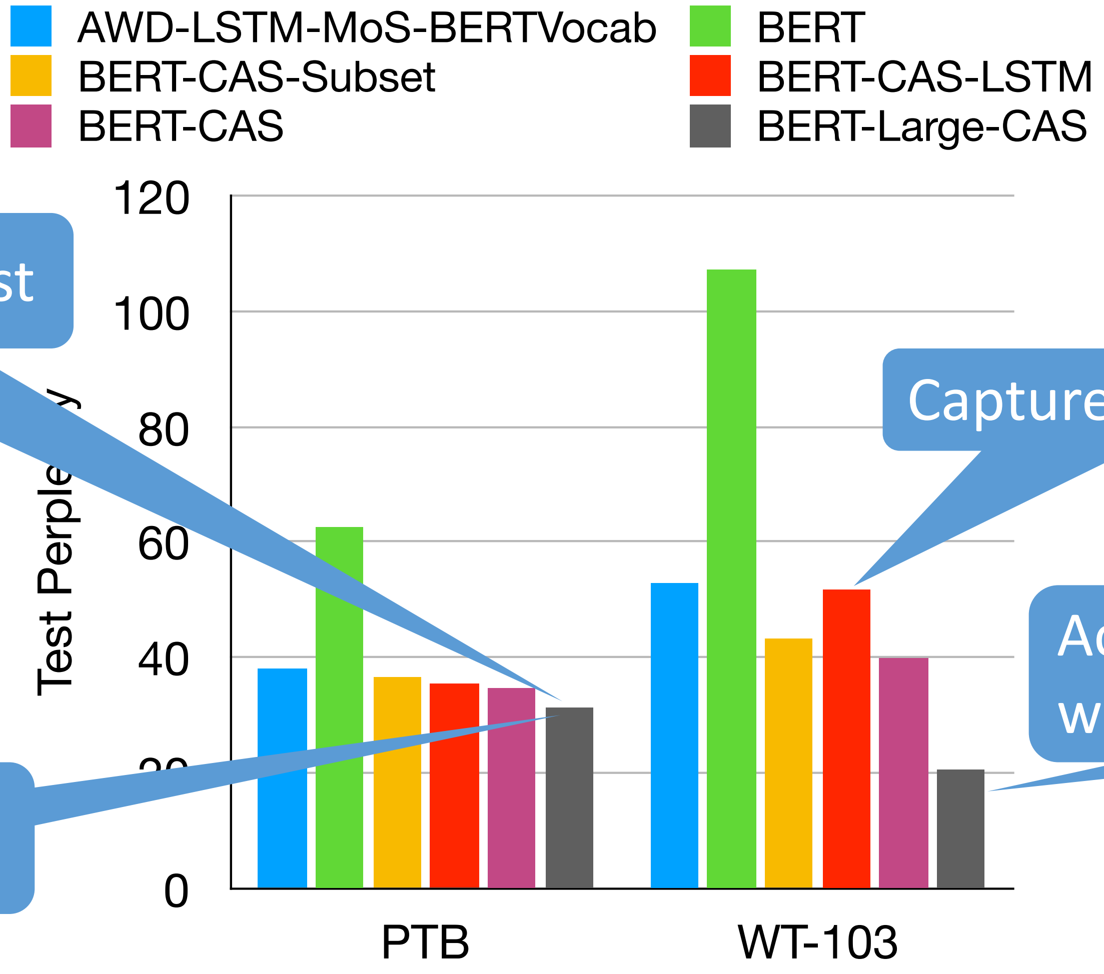


BERT-Large+CAS is best

Capture word order

Achieve SOTA: 31.34 PPL with 0.5 GPU days

Best LM: Adapted BERT with CAS



BERT-Large+CAS is best

Capture word order

Achieve 20.42 PPL with 1B tokens

Achieve SOTA: 31.34 PPL with 0.5 GPU days

Take-aways

- BERT needs to be adapted for language model
- Add RNN layers with neural architecture search works
- Fix pre-trained weights with neural architecture search works