Transformer-XL: Attentive Language Models Beyond a Fixed-Length Context

– Zihang Dai, Zhilin Yang, Yiming Yang, Jaime Carbonell, Quoc V. Le, Ruslan Salakhutdinov

Presentation author
M Saiful Bari

sbmaruf.github.io
Nanyang Technological University

July 5, 2019

Table of content

Proof that, The work done here is so **OBVIOUS**.
Try to convince you, **"Transformer is not a sequence model"**.

PREREQUISITE : Attention Is All You Need.

Lots of personal (my) interpretation included.

LSTM Difficulties

- BPTT, **Gradient vanishing** and **explosion**.
  - Graves et al, Generating sequences with recurrent neural networks.
  - **skipping i/o connection to/from the cell.**
  - **gradient clipping (-10,10).**
  - not sufficient –comment from transformer-XL paper.
- Empirically, previous work has found that LSTM language models use 200 context words on average (Khandelwal et al., 2018), indicating room for further improvement.
  - XLM uses 250 ... is it ? ? ?
- **Slow to train, challenge in parallelization (not good enough).**

LSTM Difficulties

- BPTT, **Gradient vanishing** and **explosion**.
  - Graves et al, Generating sequences with recurrent neural networks.
  - **skipping i/o connection to/from the cell.**
  - **gradient clipping (-10,10).**
  - not sufficient –comment from transformer-XL paper.
- Empirically, previous work has found that LSTM language models use 200 context words on average (Khandelwal et al., 2018), indicating room for further improvement.
  - XLM uses 250 ... is it ? ? ?
- Slow to train, challenge in parallelization (not good enough).

LSTM Difficulties

- BPTT, **Gradient vanishing** and **explosion**.
  - Graves et al, Generating sequences with recurrent neural networks.
  - **skipping i/o connection to/from the cell.**
  - **gradient clipping (-10,10).**
  - not sufficient –comment from transformer-XL paper.
- Empirically, previous work has found that LSTM language models use 200 context words on average (Khandelwal et al., 2018), indicating room for further improvement.
  - XLM uses 250 ... is it ? ? ?
- Slow to train, challenge in parallelization (not good enough).

LSTM Difficulties

- BPTT, **Gradient vanishing** and **explosion**.
  - Graves et al, Generating sequences with recurrent neural networks.
  - **skipping i/o connection to/from the cell.**
  - **gradient clipping (-10,10).**
  - not sufficient –comment from transformer-XL paper.
- Empirically, previous work has found that LSTM language models use 200 context words on average (Khandelwal et al., 2018), indicating room for further improvement.
  - XLM uses 250 ... is it ? ? ?
- **Slow to train, challenge in parallelization (not good enough).**

LSTM Difficulties

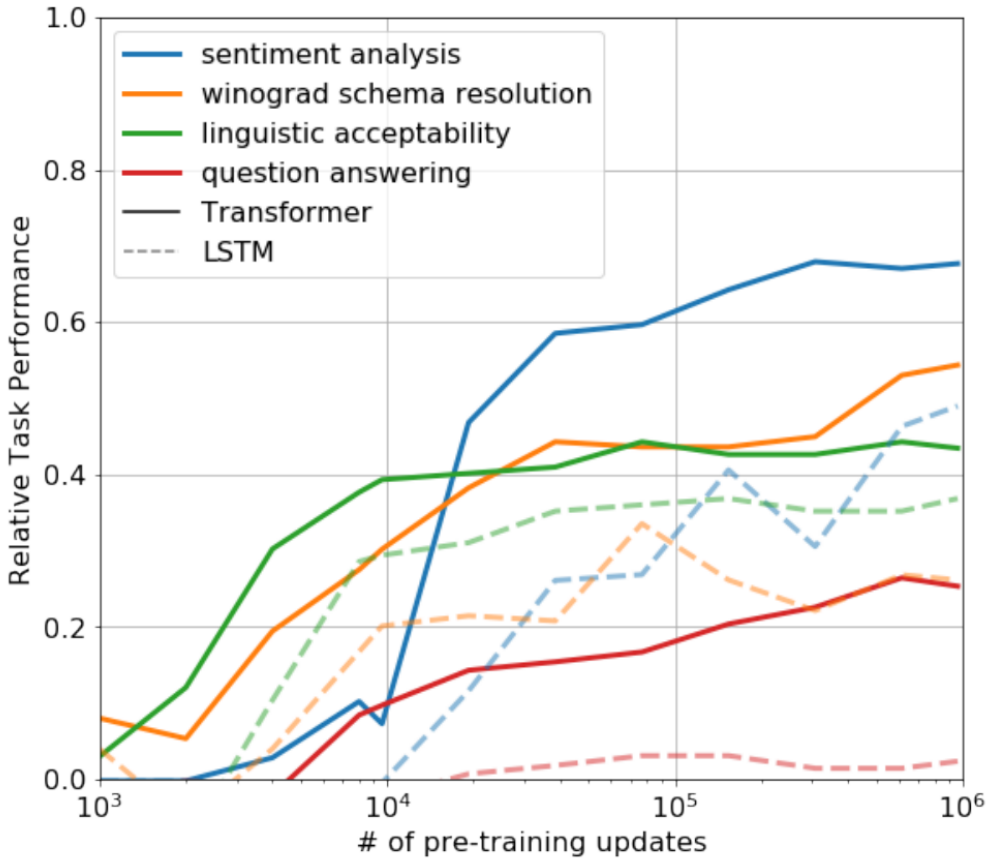


FIGURE – Plot showing the evolution of zero-shot performance on different tasks as a function of LM pre-training updates. – From GPT paper, "We also observe the **LSTM** exhibits higher variance in its zero-shot performance suggesting that the inductive bias of the Transformer architecture assists in transfer.

My Interpretation

LSTM

More of a **sequence encoder** than a **memory** based encoder.

CROSS-ATTENTION in SEQ2SEQ

Put additional **memory** by **attention** mechanism while decoding.

SELF-ATTENTION

Explore objective function within same hidden representation. - **self exploring memory**

Transformer Architecture

Attention is all you need

- Transformer - **MLP** with
  - **Properly scaled** -> ($\sqrt{d_k}$)
  - **Multi-channel** -> (head)
  - Multi-layered
  - Self -> (encoding) and cross attention -> (decoding)
  - **Sequential bias** -> (positional encoding)
- In short, "**Well engineered attentive MLP**".
- Why the name is "**TRANSFORMER**"?
  - Physics? <—> almost H shaped etc.
  - Movie? <—> ELMO, BERT, ERNIE etc.
  - Transform one sample into another sample.
    - All of the operation is transform operation instead of sequential operation, even that positional encoding.
  - No SEQUENCE MENTIONING in the name. Remember SEQ2SEQ.
  - That doesn't mean it won't be able to capture sequence.

Transformer Architecture Attention is all you need

- Transformer - **MLP** with
  - **Properly scaled** -> $(\sqrt{d_k})$
  - **Multi-channel** -> (head)
  - Multi-layered
  - Self -> (encoding) and cross attention -> (decoding)
  - **Sequential bias** -> (positional encoding)
- In short, "**Well engineered attentive MLP**".
- Why the name is "**TRANSFORMER**"?
  - Physics? <—> almost H shaped etc.
  - Movie? <—> ELMO, BERT, ERNIE etc.
  - Transform one sample into another sample.
    - All of the operation is transform opeeration instead of sequential operation, even that positionalencoding.
  - No SEQUENCE MENTIONING in the name. Remember SEQ2SEQ.
  - That doesn't mean it won't be able to capture sequence.

Transformer Architecture Attention is all you need

- Transformer - **MLP** with
  - **Properly scaled** -> ($\sqrt{d_k}$)
  - **Multi-channel** -> (head)
  - Multi-layered
  - Self -> (encoding) and cross attention -> (decoding)
  - **Sequential bias** -> (positional encoding)
- In short, "**Well engineered attentive MLP**".
- Why the name is "**TRANSFORMER**"?
  - Physics? <—> almost H shaped etc.
  - Movie? <—> ELMO, BERT, ERNIE etc.
  - Transform one sample into another sample.
    - All of the operation is transform operation instead of sequential operation, even that positional encoding.
  - No SEQUENCE MENTIONING in the name. Remember SEQ2SEQ.
  - That doesn't mean it won't be able to capture sequence.

Transformer Architecture Attention is all you need

- Transformer - **MLP** with
  - **Properly scaled** -> ($\sqrt{d_k}$)
  - **Multi-channel** -> (head)
  - Multi-layered
  - Self -> (encoding) and cross attention -> (decoding)
  - **Sequential bias** -> (positional encoding)
- In short, "**Well engineered attentive MLP**".
- Why the name is "**TRANSFORMER**"?
  - Physics? <—> almost H shaped etc.
  - Movie? <—> ELMO, BERT, ERNIE etc.
  - Transform one sample into another sample.
    - All of the operation is transform operation instead of sequential operation, even that positional encoding.
  - No SEQUENCE MENTIONING in the name. Remember SEQ2SEQ.
  - That doesn't mean it won't be able to capture sequence.

Transformer Architecture Attention is all you need

- Transformer - **MLP** with
  - **Properly scaled** -> ($\sqrt{d_k}$)
  - **Multi-channel** -> (head)
  - Multi-layered
  - Self -> (encoding) and cross attention -> (decoding)
  - **Sequential bias** -> (positional encoding)
- In short, "**Well engineered attentive MLP**".
- Why the name is "**TRANSFORMER**"?
  - Physics? <—> almost H shaped etc.
  - Movie? <—> ELMO, BERT, ERNIE etc.
  - Transform one sample into another sample.
    - All of the operation is transform operation instead of sequential operation, even that positional encoding.
  - No SEQUENCE MENTIONING in the name. Remember SEQ2SEQ.
  - That doesn't mean it won't be able to capture sequence.

Transformer Architecture Attention is all you need

- Transformer - **MLP** with
  - **Properly scaled** -> ($\sqrt{d_k}$)
  - **Multi-channel** -> (head)
  - Multi-layered
  - Self -> (encoding) and cross attention -> (decoding)
  - **Sequential bias** -> (positional encoding)
- In short, "**Well engineered attentive MLP**".
- Why the name is "**TRANSFORMER**" ?
  - Physics ? <—> almost H shaped etc.
  - Movie ? <—> ELMO, BERT, ERNIE etc.
  - Transform one sample into another sample.
  - No SEQUENCE MENTIONING in the name. Remember SEQ2SEQ.
  - That doesn't mean it won't be able to capture sequence.

Transformer Architecture Attention is all you need

- Transformer - **MLP** with
  - **Properly scaled ->** ($\sqrt{d_k}$)
  - **Multi-channel ->** (head)
  - Multi-layered
  - Self -> (encoding) and cross attention -> (decoding)
  - **Sequential bias ->** (positional encoding)
- In short, "**Well engineered attentive MLP**".
- Why the name is "**TRANSFORMER**"?
  - Physics? <—> almost H shaped etc.
  - Movie? <—> ELMO, BERT, ERNIE etc.
  - Transform one sample into another sample.
    - All of the operation is transform operation instead of sequential operation, even that positional encoding.
  - No SEQUENCE MENTIONING in the name. Remember SEQ2SEQ.
  - That doesn't mean it won't be able to capture sequence.

Transformer Architecture Attention is all you need

- Transformer - **MLP** with
  - **Properly scaled** -> ($\sqrt{d_k}$)
  - **Multi-channel** -> (head)
  - Multi-layered
  - Self -> (encoding) and cross attention -> (decoding)
  - **Sequential bias** -> (positional encoding)
- In short, "**Well engineered attentive MLP**".
- Why the name is "**TRANSFORMER**" ?
  - Physics ? <—> almost H shaped etc.
  - Movie ? <—> ELMO, BERT, ERNIE etc.
  - Transform one sample into another sample.
    - All of the operation is transform operation instead of sequential operation, even that positional encoding.
  - No SEQUENCE MENTIONING in the name. Remember SEQ2SEQ.
  - That doesn't mean it won't be able to capture sequence.

Transformer Architecture Attention is all you need

- Transformer - **MLP** with
  - **Properly scaled** -> ($\sqrt{d_k}$)
  - **Multi-channel** -> (head)
  - Multi-layered
  - Self -> (encoding) and cross attention -> (decoding)
  - **Sequential bias** -> (positional encoding)
- In short, "**Well engineered attentive MLP**".
- Why the name is "**TRANSFORMER**" ?
  - Physics ? <—> almost H shaped etc.
  - Movie ? <—> ELMO, BERT, ERNIE etc.
  - Transform one sample into another sample.
    - All of the operation is transform operation instead of sequential operation, even that positional encoding.
  - No SEQUENCE MENTIONING in the name. Remember SEQ2SEQ.
  - That doesn't mean it won't be able to capture sequence.

Transformer Architecture Attention is all you need

- Transformer - **MLP** with
  - **Properly scaled** -> ($\sqrt{d_k}$)
  - **Multi-channel** -> (head)
  - Multi-layered
  - Self -> (encoding) and cross attention -> (decoding)
  - **Sequential bias** -> (positional encoding)
- In short, "**Well engineered attentive MLP**".
- Why the name is "**TRANSFORMER**" ?
  - Physics ? <—> almost H shaped etc.
  - Movie ? <—> ELMO, BERT, ERNIE etc.
  - Transform one sample into another sample.
    - All of the operation is transform operation instead of sequential operation, even that positional encoding.
  - No SEQUENCE MENTIONING in the name. Remember SEQ2SEQ.
  - That doesn't mean it won't be able to capture sequence.

Transformer Architecture Attention is all you need

- Transformer - **MLP** with
  - **Properly scaled** -> $(\sqrt{d_k})$
  - **Multi-channel** -> (head)
  - Multi-layered
  - Self -> (encoding) and cross attention -> (decoding)
  - **Sequential bias** -> (positional encoding)
- In short, "**Well engineered attentive MLP**".
- Why the name is "**TRANSFORMER**" ?
  - Physics ? <—> almost H shaped etc.
  - Movie ? <—> ELMO, BERT, ERNIE etc.
  - Transform one sample into another sample.
    - All of the operation is **transform operation** instead of sequential operation, even that positional encoding.
  - No SEQUENCE MENTIONING in the name. Remember SEQ2SEQ.
  - That doesn't mean it won't be able to capture sequence.

Transformer Architecture Attention is all you need

- Transformer - **MLP** with
  - **Properly scaled** -> $(\sqrt{d_k})$
  - **Multi-channel** -> (head)
  - Multi-layered
  - Self -> (encoding) and cross attention -> (decoding)
  - **Sequential bias** -> (positional encoding)
- In short, "**Well engineered attentive MLP**".
- Why the name is "**TRANSFORMER**" ?
  - Physics ? <—> almost H shaped etc.
  - Movie ? <—> ELMO, BERT, ERNIE etc.
  - Transform one sample into another sample.
    - All of the operation is **transform operation** instead of sequential operation, even that positional encoding.
  - No SEQUENCE MENTIONING in the name. Remember SEQ2SEQ.
  - That doesn't mean it won't be able to capture sequence.

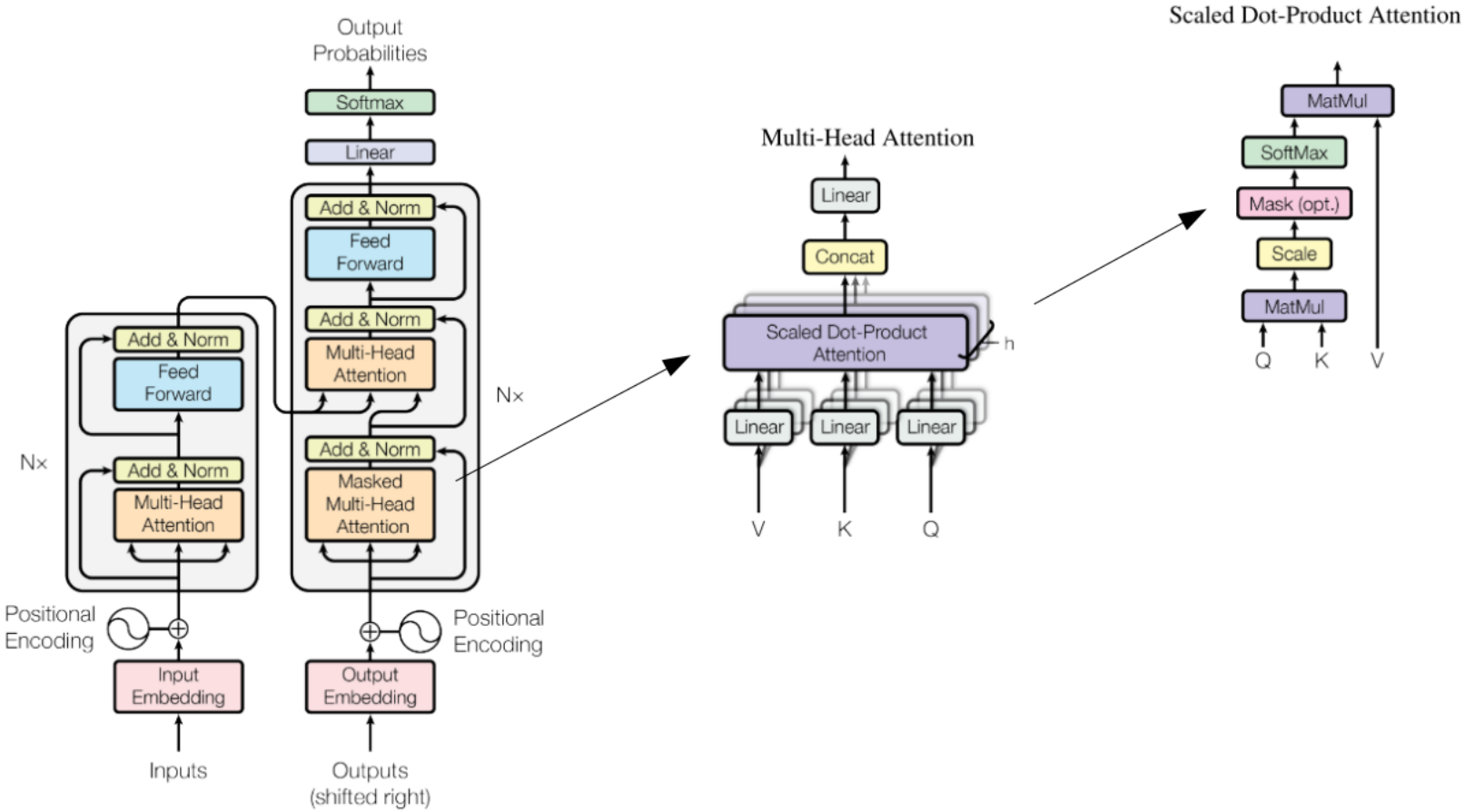
Transformer Architecture Attention is all you need

- Transformer - **MLP** with
  - **Properly scaled** -> ($\sqrt{d_k}$)
  - **Multi-channel** -> (head)
  - Multi-layered
  - Self -> (encoding) and cross attention -> (decoding)
  - **Sequential bias** -> (positional encoding)
- In short, "**Well engineered attentive MLP**".
- Why the name is "**TRANSFORMER**" ?
  - Physics ? <—> almost H shaped etc.
  - Movie ? <—> ELMO, BERT, ERNIE etc.
  - Transform one sample into another sample.
    - All of the operation is **transform operation** instead of sequential operation, even that positional encoding.
  - No SEQUENCE MENTIONING in the name. Remember SEQ2SEQ.
  - That doesn't mean it won't be able to capture sequence.

Transformer Architecture Attention is all you need

- Transformer - **MLP** with
  - **Properly scaled** -> ($\sqrt{d_k}$)
  - **Multi-channel** -> (head)
  - Multi-layered
  - Self -> (encoding) and cross attention -> (decoding)
  - **Sequential bias** -> (positional encoding)
- In short, "**Well engineered attentive MLP**".
- Why the name is "**TRANSFORMER**" ?
  - Physics ? <—> almost H shaped etc.
  - Movie ? <—> ELMO, BERT, ERNIE etc.
  - Transform one sample into another sample.
    - All of the operation is **transform operation** instead of sequential operation, even that positional encoding.
  - No SEQUENCE MENTIONING in the name. Remember SEQ2SEQ.
  - That doesn't mean it won't be able to capture sequence.

Transformer Architecture Attention is all you need



Transformer Architecture Attention is all you need

- **Transformers** have a potential of learning **longer-term dependency**, but are limited by a **fixed-length** context in the setting of **language modeling**. – first line of transformer-XL paper.

Transformer Architecture Attention is all you need

- **Transformers** have a potential of learning **longer-term dependency**, but are limited by a **fixed-length** context in the setting of **language modeling**. – first line of transformer-XL paper.

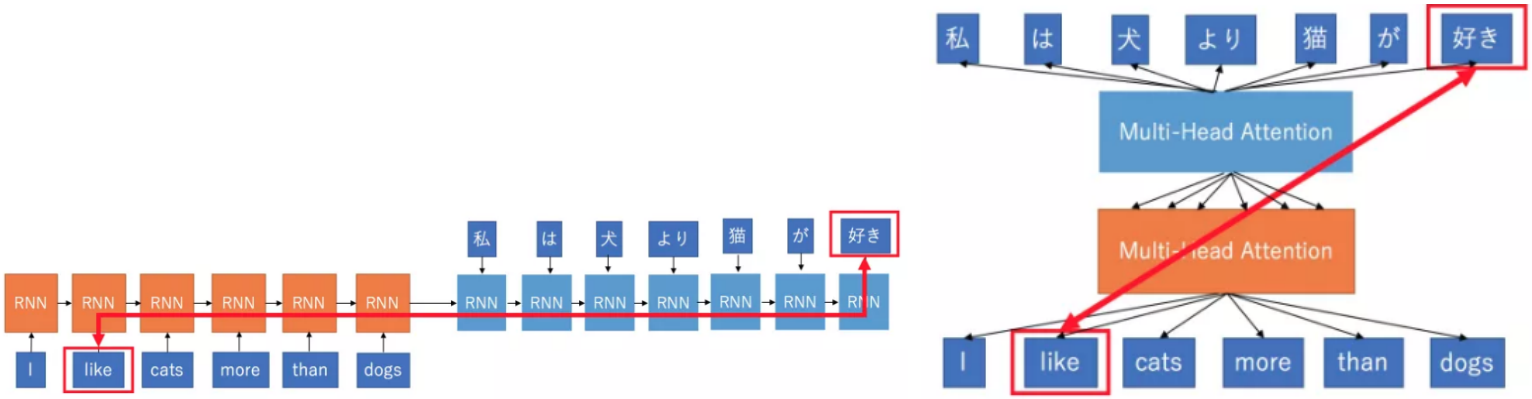


FIGURE – Traditional sequence to sequence model vs multihead attentional model (figure source)

Transformer Architecture Attention is all you need

- **Transformers** have a potential of learning **longer-term dependency**, but are limited by a **fixed-length** context in the setting of **language modeling**. – first line of transformer-XL paper.
- This model choice provides us with **a more structured memory** for handling long-term dependencies in text, compared to alternatives like recurrent networks. – GPT page 2 second para 3rd line.
- The **sequential objective** of transformer is done with only **bias** (shifting or injecting new feature) which does not **mimic** conditional probability density calculation.
  - Why as a model it has good potential for sequence modeling apart from better **gradient flow** ?

Transformer Architecture Attention is all you need

- **Transformers** have a potential of learning **longer-term dependency**, but are limited by a **fixed-length** context in the setting of **language modeling**. – first line of transformer-XL paper.
- This model choice provides us with **a more structured memory** for handling long-term dependencies in text, compared to alternatives like recurrent networks. – GPT page 2 second para 3rd line.
- The **sequential objective** of transformer is done with only **bias** (shifting or injecting new feature) which does not **mimic** conditional probability density calculation.
  - Why as a model it has good potential for sequence modeling apart from better **gradient flow** ?

Transformer Architecture Attention is all you need

- **Transformers** have a potential of learning **longer-term dependency**, but are limited by a **fixed-length** context in the setting of **language modeling**. – first line of transformer-XL paper.
- This model choice provides us with **a more structured memory** for handling long-term dependencies in text, compared to alternatives like recurrent networks. – GPT page 2 second para 3rd line.
- The **sequential objective** of transformer is done with only **bias** (shifting or injecting new feature) which does not **mimic** conditional probability density calculation.
    - Why as a model it has good potential for sequence modeling apart from better **gradient flow** ?

Transformer Architecture Attention is all you need

- **Transformers** have a potential of learning **longer-term dependency**, but are limited by a **fixed-length** context in the setting of **language modeling**. – first line of transformer-XL paper.
- This model choice provides us with **a more structured memory** for handling long-term dependencies in text, compared to alternatives like recurrent networks. – GPT page 2 second para 3rd line.
- The **sequential objective** of transformer is done with only **bias** (shifting or injecting new feature) which does not **mimic** conditional probability density calculation.
  - Why as a model it has good potential for sequence modeling apart from better **gradient flow** ?

Transformer Architecture Attention is all you need

- The **sequential objective** of transformer is done with only **bias** (shifting or injecting new feature) which does not **mimic** conditional probability density calculation.
  - Why as a model it has good potential for sequence modeling apart from better **gradient flow** ?
  - More **structured memory** (according to GPT), in my opinion, multi-channel, attention architecture.
  - without **positional encoding**, **Attention** is more of a ordered-set (sorted with importance not sequence) based memory.
  - **Lots of params.**, lots of way to **tweak**. Gives more flexibility to design **task based biased** model.
  - **Ability to scale**.

Transformer Architecture Attention is all you need

- The **sequential objective** of transformer is done with only **bias** (shifting or injecting new feature) which does not **mimic** conditional probability density calculation.
  - Why as a model it has good potential for sequence modeling apart from better **gradient flow** ?
  - More **structured memory** (according to GPT), in my opinion, multi-channel, attention architecture.
  - without **positional encoding**, **Attention** is more of a ordered-set (sorted with importance not sequence) based memory.
  - **Lots of params.**, lots of way to **tweak**. Gives more flexibility to design **task based biased** model.
  - **Ability to scale**.

Transformer Architecture Attention is all you need

- The **sequential objective** of transformer is done with only **bias** (shifting or injecting new feature) which does not **mimic** conditional probability density calculation.
  - Why as a model it has good potential for sequence modeling apart from better **gradient flow** ?
  - More **structured memory** (according to GPT), in my opinion, multi-channel, attention architecture.
  - without **positional encoding**, **Attention** is more of a ordered-set (sorted with importance not sequence) based memory.
  - **Lots of params.**, lots of way to **tweak**. Gives more flexibility to design **task based biased** model.
  - **Ability to scale**.

Transformer Architecture Attention is all you need

- The **sequential objective** of transformer is done with only **bias** (shifting or injecting new feature) which does not **mimic** conditional probability density calculation.
  - Why as a model it has good potential for sequence modeling apart from better **gradient flow** ?
  - More **structured memory** (according to GPT), in my opinion, multi-channel, attention architecture.
  - without **positional encoding**, **Attention** is more of a ordered-set (sorted with importance not sequence) based memory.
  - **Lots of params.**, lots of way to **tweak**. Gives more flexibility to design **task based biased** model.
  - **Ability to scale**.

Transformer Architecture Attention is all you need

- The **sequential objective** of transformer is done with only **bias** (shifting or injecting new feature) which does not **mimic** conditional probability density calculation.
  - Why as a model it has good potential for sequence modeling apart from better **gradient flow** ?
  - More **structured memory** (according to GPT), in my opinion, multi-channel, attention architecture.
  - without **positional encoding**, **Attention** is more of a ordered-set (sorted with importance not sequence) based memory.
  - **Lots of params.**, lots of way to **tweak**. Gives more flexibility to design **task based biased** model.
  - Ability to scale.

Transformer Architecture Attention is all you need

- The **sequential objective** of transformer is done with only **bias** (shifting or injecting new feature) which does not **mimic** conditional probability density calculation.
  - Why as a model it has good potential for sequence modeling apart from better **gradient flow** ?
  - More **structured memory** (according to GPT), in my opinion, multi-channel, attention architecture.
  - without **positional encoding**, **Attention** is more of a ordered-set (sorted with importance not sequence) based memory.
  - **Lots of params.**, lots of way to **tweak**. Gives more flexibility to design **task based biased** model.
  - **Ability to scale**.

Transformer Architecture Attention is all you need

Josh Tenenbaum

Reverse-engineer how intelligence work in human mind and brain. Reverse engineering means doing science like Engineers. – CBMM motivation

- **Transformer** more of a **Engineered** model than **neuro-motivated** model.

Transformer Architecture Attention is all you need

- **Transformer** more of a **Engineered** model than **neuro-motivated** model.
- **(bad idea) :** If transformers works who cares if it's mimics conditional probability density or not.
- We should look for **what's missing**. Example problem : Why one head works better for **NER** training, but not for BERT ?
- Network might be **over-performing** on **memory** but **under-performing** capturing **sequence**.
- May be **over-performing** on **memory** gives enough boost to get SOTA. but still can be improved with better **sequence encoding** mechanism.
- Try to see what **compromises** done in the **transformer** to make it **scalable**.

Transformer Architecture Attention is all you need

- **Transformer** more of a **Engineered** model than **neuro-motivated** model.
- **(bad idea) :** If transformers works who cares if it's mimics conditional probability density or not.
- We should look for **what's missing**. Example problem : Why one head works better for **NER** training, but not for BERT ?
- Network might be **over-performing** on **memory** but **under-performing** capturing **sequence**.
- May be **over-performing** on **memory** gives enough boost to get SOTA. but still can be improved with better **sequence encoding** mechanism.
- Try to see what **compromises** done in the **transformer** to make it **scalable**.

Transformer Architecture Attention is all you need

- **Transformer** more of a **Engineered** model than **neuro-motivated** model.
- **(bad idea) :** If transformers works who cares if it's mimics conditional probability density or not.
- We should look for **what's missing**. Example problem : Why one head works better for **NER** training, but not for BERT ?
- Network might be **over-performing** on **memory** but **under-performing** capturing **sequence**.
- May be **over-performing** on **memory** gives enough boost to get SOTA. but still can be improved with better **sequence encoding** mechanism.
- Try to see what **compromises** done in the **transformer** to make it **scalable**.

Transformer Architecture Attention is all you need

- **Transformer** more of a **Engineered** model than **neuro-motivated** model.
- **(bad idea) :** If transformers works who cares if it's mimics conditional probability density or not.
- We should look for **what's missing**. Example problem : Why one head works better for **NER** training, but not for BERT ?
- Network might be **over-performing** on **memory** but **under-performing** capturing **sequence**.
- May be **over-performing** on **memory** gives enough boost to get SOTA. but still can be improved with better **sequence encoding** mechanism.
- Try to see what **compromises** done in the **transformer** to make it **scalable**.

Transformer Architecture Attention is all you need

- **Transformer** more of a **Engineered** model than **neuro-motivated** model.
- **(bad idea) :** If transformers works who cares if it's mimics conditional probability density or not.
- We should look for **what's missing**. Example problem : Why one head works better for **NER** training, but not for BERT ?
- Network might be **over-performing** on **memory** but **under-performing** capturing **sequence**.
- May be **over-performing** on **memory** gives enough boost to get SOTA. but still can be improved with better **sequence encoding** mechanism.
- Try to see what **compromises** done in the **transformer** to make it **scalable**.

Transformer Architecture Attention is all you need

- **Transformer** more of a **Engineered** model than **neuro-motivated** model.
- **(bad idea) :** If transformers works who cares if it's mimics conditional probability density or not.
- We should look for **what's missing**. Example problem : Why one head works better for **NER** training, but not for BERT ?
- Network might be **over-performing** on **memory** but **under-performing** capturing **sequence**.
- May be **over-performing** on **memory** gives enough boost to get SOTA. but still can be improved with better **sequence encoding** mechanism.
- Try to see what **compromises** done in the **transformer** to make it **scalable**.

Transformer-XL

Works on those compromises.

Transformer-XL, XL means extra long.

It Proposes,

- It consists of a segment-level (text stream) recurrence mechanism.
- A novel positional encoding scheme.

It Solves,

- Resolves the context fragmentation problem.
- Propose learning dependency beyond a fixed length without disrupting **temporal coherence** (???).
- Better capturing longer-term dependency.
- Better sequential feature injection.

Transformer-XL

Works on those compromises.

Transformer-XL, XL means extra long.

It Proposes,

- It consists of a segment-level (text stream) recurrence mechanism.
- A novel positional encoding scheme.

It Solves,

- Resolves the context fragmentation problem.
- Propose learning dependency beyond a fixed length without disrupting **temporal coherence** (???).
- Better capturing longer-term dependency.
- Better sequential feature injection.

Transformer-XL

Works on those compromises.

Transformer-XL, XL means extra long.

It Proposes,

- It consists of a segment-level (text stream) recurrence mechanism.
- A novel positional encoding scheme.

It Solves,

- Resolves the context fragmentation problem.
- Propose learning dependency beyond a fixed length without disrupting **temporal coherence** (???).
- Better capturing longer-term dependency.
- Better sequential feature injection.

Transformer-XL

Works on those compromises.

Transformer-XL, XL means extra long.

It Proposes,

- It consists of a segment-level (text stream) recurrence mechanism.
- A novel positional encoding scheme.

It Solves,

- Resolves the context fragmentation problem.
- Propose learning dependency beyond a fixed length without disrupting **temporal coherence (???)**.
- Better capturing longer-term dependency.
- Better sequential feature injection.

Transformer-XL

Works on those compromises.

Transformer-XL, XL means extra long.

It Proposes,

- It consists of a segment-level (text stream) recurrence mechanism.
- A novel positional encoding scheme.

It Solves,

- Resolves the context fragmentation problem.
- Propose learning dependency beyond a fixed length without disrupting **temporal coherence (???)**.
- Better capturing longer-term dependency.
- Better sequential feature injection.

Transformer-XL

Works on those compromises.

Transformer-XL, XL means extra long.

It Proposes,

- It consists of a segment-level (text stream) recurrence mechanism.
- A novel positional encoding scheme.

It Solves,

- Resolves the context fragmentation problem.
- Propose learning dependency beyond a fixed length without disrupting **temporal coherence (???)**.
- Better capturing longer-term dependency.
- Better sequential feature injection.

Numbers

- **TransformerXL learns dependency that is 80% longer than RNNs and 450% longer than vanilla Transformers.**
- Achieves better performance on both short and long sequences.
- **Up to 1,800+ times faster than vanilla Transformers during evaluation.**
- We improve the state-of-the-art results of bpc/perplexity to 0.99 on enwiki8, 1.08 on text8, 18.3 on WikiText-103, 21.8 on One Billion Word, and 54.5 on Penn Treebank (without finetuning) evaluation.
- When trained only on WikiText-103, Transformer-XL manages to generate reasonably coherent, novel text articles with thousands of tokens.

Numbers

- **TransformerXL learns dependency that is 80% longer than RNNs and 450% longer than vanilla Transformers.**
- Achieves better performance on both short and long sequences.
- **Up to 1,800+ times faster than vanilla Transformers during evaluation.**
- We improve the state-of-the-art results of bpc/perplexity to 0.99 on enwiki8, 1.08 on text8, 18.3 on WikiText-103, 21.8 on One Billion Word, and 54.5 on Penn Treebank (without finetuning) evaluation.
- When trained only on WikiText-103, Transformer-XL manages to generate reasonably coherent, novel text articles with thousands of tokens.

Numbers

- **TransformerXL learns dependency that is 80% longer than RNNs and 450% longer than vanilla Transformers.**
- Achieves better performance on both short and long sequences.
- **Up to 1,800+ times faster than vanilla Transformers during evaluation.**
- We improve the state-of-the-art results of bpc/perplexity to 0.99 on enwiki8, 1.08 on text8, 18.3 on WikiText-103, 21.8 on One Billion Word, and 54.5 on Penn Treebank (without finetuning) evaluation.
- When trained only on WikiText-103, Transformer-XL manages to generate reasonably coherent, novel text articles with thousands of tokens.

Numbers

- **TransformerXL learns dependency that is 80% longer than RNNs and 450% longer than vanilla Transformers.**
- Achieves better performance on both short and long sequences.
- **Up to 1,800+ times faster than vanilla Transformers during evaluation.**
- We improve the state-of-the-art results of bpc/perplexity to 0.99 on enwiki8, 1.08 on text8, 18.3 on WikiText-103, 21.8 on One Billion Word, and 54.5 on Penn Treebank (without finetuning) evaluation.
- When trained only on WikiText-103, Transformer-XL manages to generate reasonably coherent, novel text articles with thousands of tokens.

Numbers

- **TransformerXL learns dependency that is 80% longer than RNNs and 450% longer than vanilla Transformers.**
- Achieves better performance on both short and long sequences.
- **Up to 1,800+ times faster than vanilla Transformers during evaluation.**
- We improve the state-of-the-art results of bpc/perplexity to 0.99 on enwiki8, 1.08 on text8, 18.3 on WikiText-103, 21.8 on One Billion Word, and 54.5 on Penn Treebank (without finetuning) evaluation.
- When trained only on WikiText-103, Transformer-XL manages to generate reasonably coherent, novel text articles with thousands of tokens.

Motivation

- **Character-Level Language Modeling with Deeper Self-Attention** – Al-Rfou (XLM also cited) et al.
  - Trained deep transformer networks for **character-level language modeling**, which outperform LSTMs by a large margin.
  - Separated **fixed-length segments** of a few hundred characters.
  - No information flow across segments as it is sample based training.
  - Text segmented without sentential information (with fixed-length) but not considering context between segments. (**context fragmentation problem**)

Motivation

- **Character-Level Language Modeling with Deeper Self-Attention** – Al-Rfou (XLM also cited) et al.
  - Trained deep transformer networks for **character-level language modeling**, which outperform LSTMs by a large margin.
  - Separated **fixed-length segments** of a few hundred characters.
  - No information flow across segments as it is sample based training.
  - Text segmented without sentential information (with fixed-length) but not considering context between segments. (**context fragmentation problem**)

Motivation

- **Character-Level Language Modeling with Deeper Self-Attention** – Al-Rfou (XLM also cited) et al.
  - Trained deep transformer networks for **character-level language modeling**, which outperform LSTMs by a large margin.
  - Separated **fixed-length segments** of a few hundred characters.
  - No information flow across segments as it is sample based training.
  - Text segmented without sentential information (with fixed-length) but not considering context between segments. (**context fragmentation problem**)

Motivation

- **Character-Level Language Modeling with Deeper Self-Attention** – Al-Rfou (XLM also cited) et al.
  - Trained deep transformer networks for **character-level language modeling**, which outperform LSTMs by a large margin.
  - Separated **fixed-length segments** of a few hundred characters.
  - No information flow across segments as it is sample based training.
  - Text segmented without sentential information (with fixed-length) but not considering context between segments. (**context fragmentation problem**)

Motivation

- **Character-Level Language Modeling with Deeper Self-Attention** – Al-Rfou (XLM also cited) et al.
  - Trained deep transformer networks for **character-level language modeling**, which outperform LSTMs by a large margin.
  - Separated **fixed-length segments** of a few hundred characters.
  - No information flow across segments as it is sample based training.
  - Text segmented without sentential information (with fixed-length) but not considering context between segments. (**context fragmentation problem**)

Transformer-XL

It Proposes,

- **It consists of a segment-level (text stream) recurrence mechanism.**
- A novel positional encoding scheme.

It Solves,

- Propose learning dependency beyond a fixed length without disrupting temporal coherence (? ? ?).
- Better capturing longer-term dependency.
- **Resolves the context fragmentation problem.**

Conditional probability for LM

Given a **corpus** of token, $x = (x_1, x_2, ..., x_T)$, for language modeling task,

- **Estimate** the joint probability $P(x)$
- Which is auto-regressively factored as $P(x) = \prod_t (x_t | x_{<t})$
- A trainable NN is used to **encode** the context $x_{<t}$ into a fixed size hidden state
- which is multiplied with the **word embeddings** to obtain the logits.
- The logits are then fed into the **Softmax function** yeilding a categorical probability distribution over the next token.

Conditional probability for LM

Given a **corpus** of token, $x = (x_1, x_2, ..., x_T)$, for language modeling task,

- **Estimate** the joint probability $P(x)$
- Which is auto-regressively factored as $P(x) = \prod_t (x_t | x_{<t})$
- A trainable NN is used to **encode** the context $x_{<t}$ into a fixed size hidden state
- which is multiplied with the **word embeddings** to obtain the logits.
- The logits are then fed into the **Softmax function** yeilding a categorical probability distribution over the next token.

Conditional probability for LM

Given a **corpus** of token, $x = (x_1, x_2, ..., x_T)$, for language modeling task,

- **Estimate** the joint probability $P(x)$
- Which is auto-regressively factored as $P(x) = \prod_t (x_t | x_{<t})$
- A trainable NN is used to **encode** the context $x_{<t}$ into a fixed size hidden state
- which is multiplied with the **word embeddings** to obtain the logits.
- The logits are then fed into the **Softmax function** yeilding a categorical probability distribution over the next token.

Conditional probability for LM

Given a **corpus** of token, $x = (x_1, x_2, ..., x_T)$, for language modeling task,

- **Estimate** the joint probability $P(x)$
- Which is auto-regressively factored as $P(x) = \prod_t (x_t | x_{<t})$
- A trainable NN is used to **encode** the context $x_{<t}$ into a fixed size hidden state
- which is multiplied with the **word embeddings** to obtain the logits.
- The logits are then fed into the **Softmax function** yeilding a categorical probability distribution over the next token.

Conditional probability for LM

Given a **corpus** of token, $x = (x_1, x_2, ..., x_T)$, for language modeling task,

- **Estimate** the joint probability $P(x)$
- Which is auto-regressively factored as $P(x) = \prod_t (x_t | x_{<t})$
- A trainable NN is used to **encode** the context $x_{<t}$ into a fixed size hidden state
- which is multiplied with the **word embeddings** to obtain the logits.
- The logits are then fed into the **Softmax function** yeilding a categorical probability distribution over the next token.

Vanilla Transformer LM During Training

- Divide dataset into **multiple segments**.
- Use each of the segment as a sample to train the model.
- **Ignore the context between two segments.**

Comment :

- The largest possible **dependency length** is **upper bounded** by the segment length.
- It is possible to use **padding** or **chunking**.
  - Does not solve **context fragmentation** problem fully.

Vanilla Transformer LM During Training

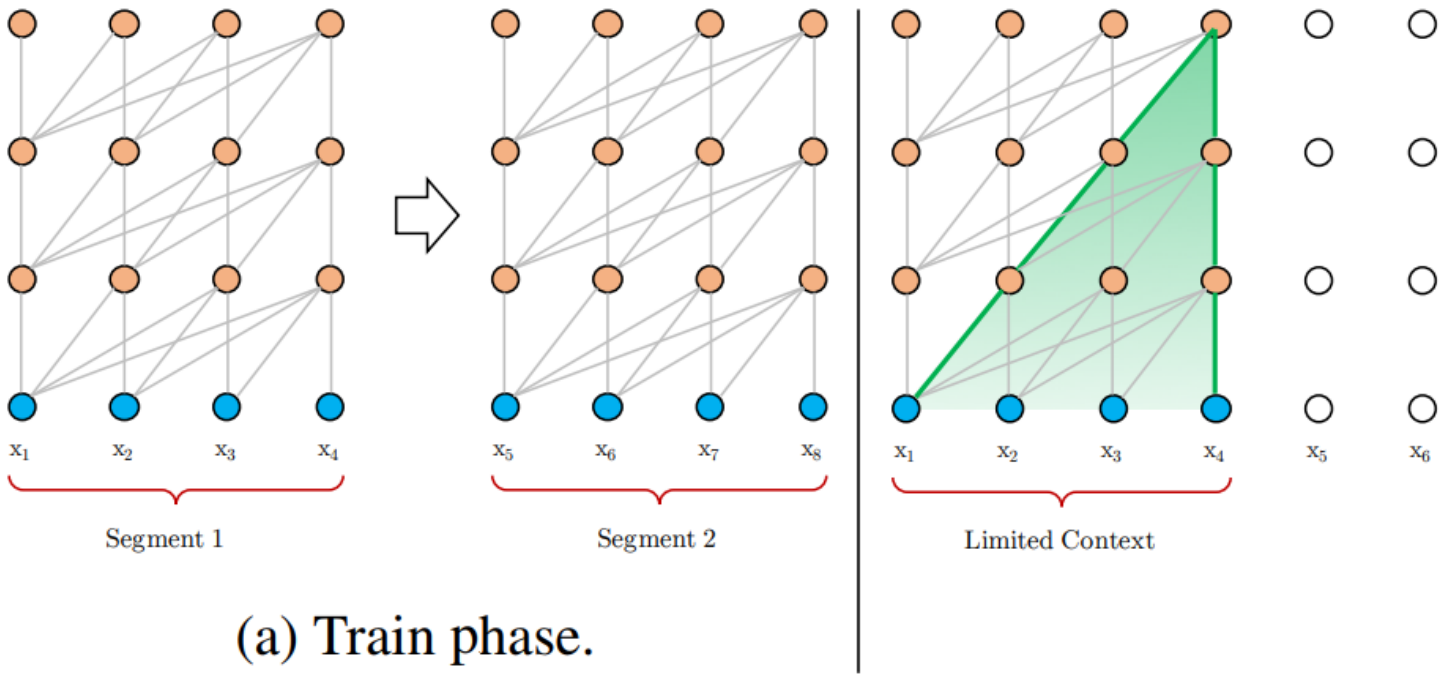


FIGURE – Illustration of the vanilla model with a **segment length 4**.

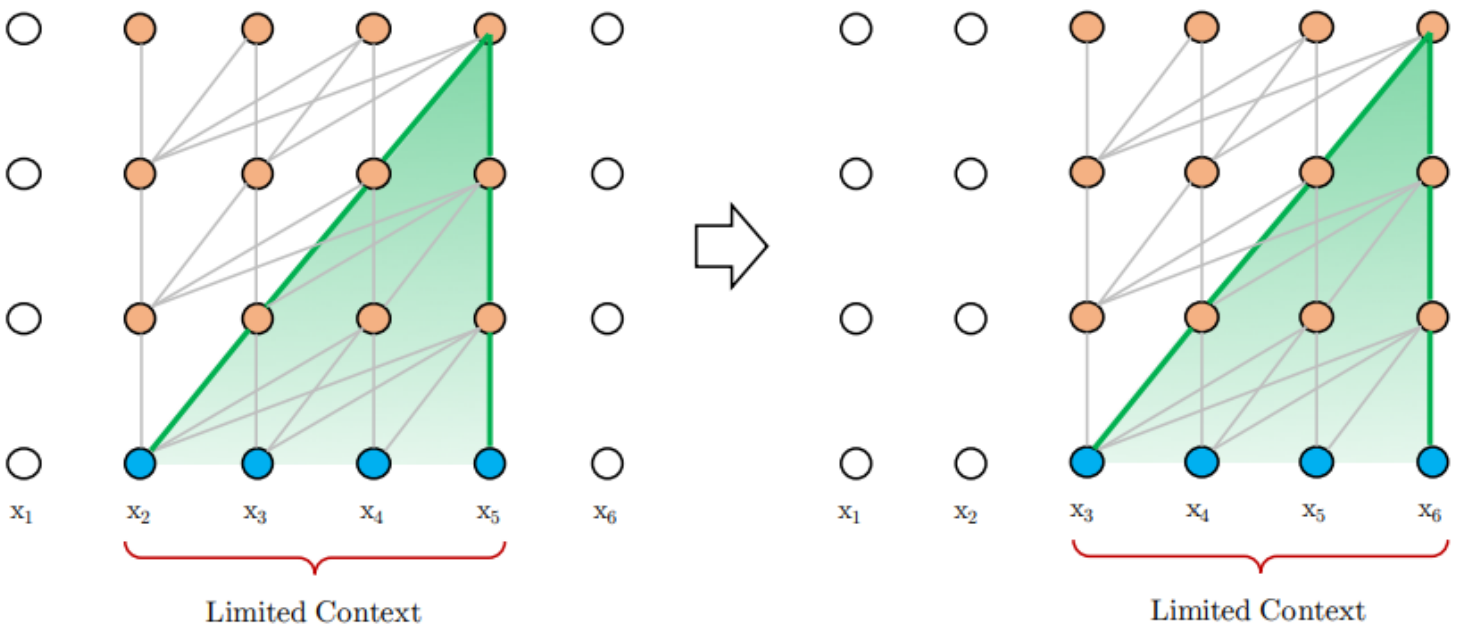
Vanilla Transformer LM During Evaluation

- At each step, the vanilla model also consumes a **segment of the same length** as in training.
- Then makes a prediction.
- Then, at the next step, the segment is shifted to the right by only one position.
- and the new segment has to be processed all from scratch.

Comment :

- **This procedure ensures that each prediction utilizes the longest possible context exposed during training.**
- **Relieves context fragmentation issue.**
- However, this evaluation procedure is extremely **expensive**.

Vanilla Transformer LM During Evaluation



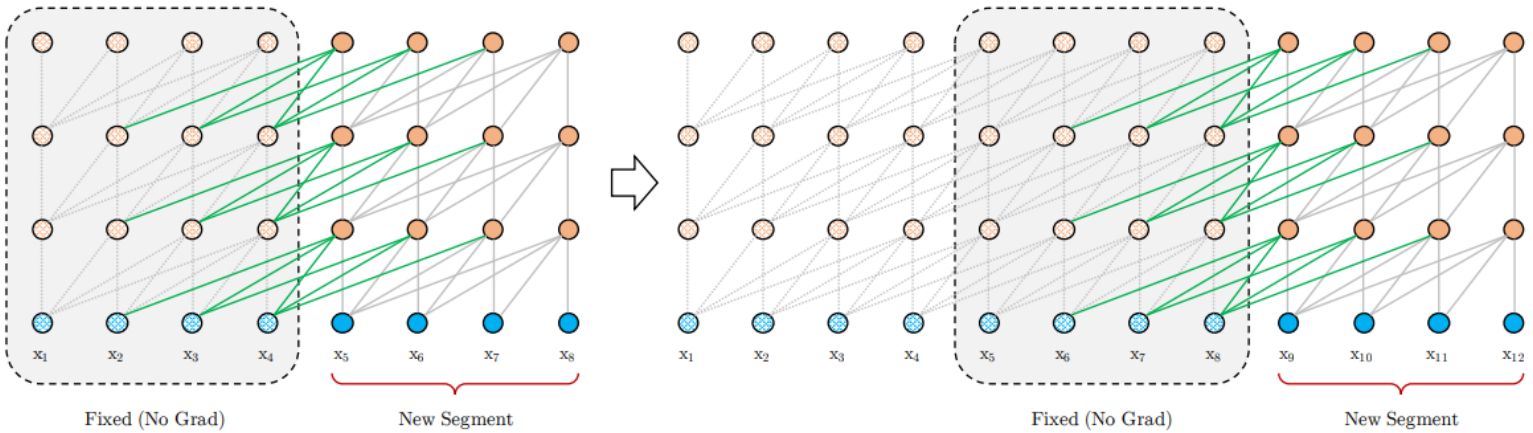
(b) Evaluation phase.

FIGURE – Illustration of the vanilla model with a **segment length 4**.

Segment-Level Recurrence with State Reuse

Recurrence mechanism :

- The hidden state sequence computed for the previous segment is **fixed** and **cached** to be reused as an extended context when the model processes the next new segment.



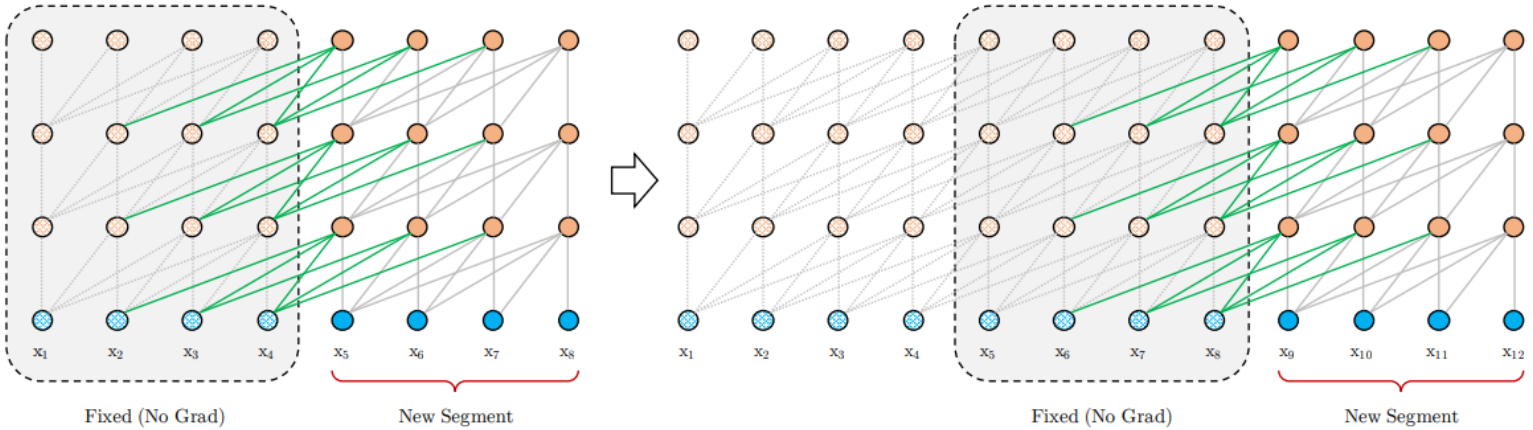
(a) Training phase.

FIGURE – Illustration of the Transformer-XL model with a **segment length 4**.

Segment-Level Recurrence with State Reuse

Recurrence mechanism : After first segment computation, from second segment, each hidden layer receives,

- Output of the previous hidden layer of current segment (if there is any)-> gray edges.
- if there is no previous token (ie, first token etc.), use cached data from the network-> green edges.

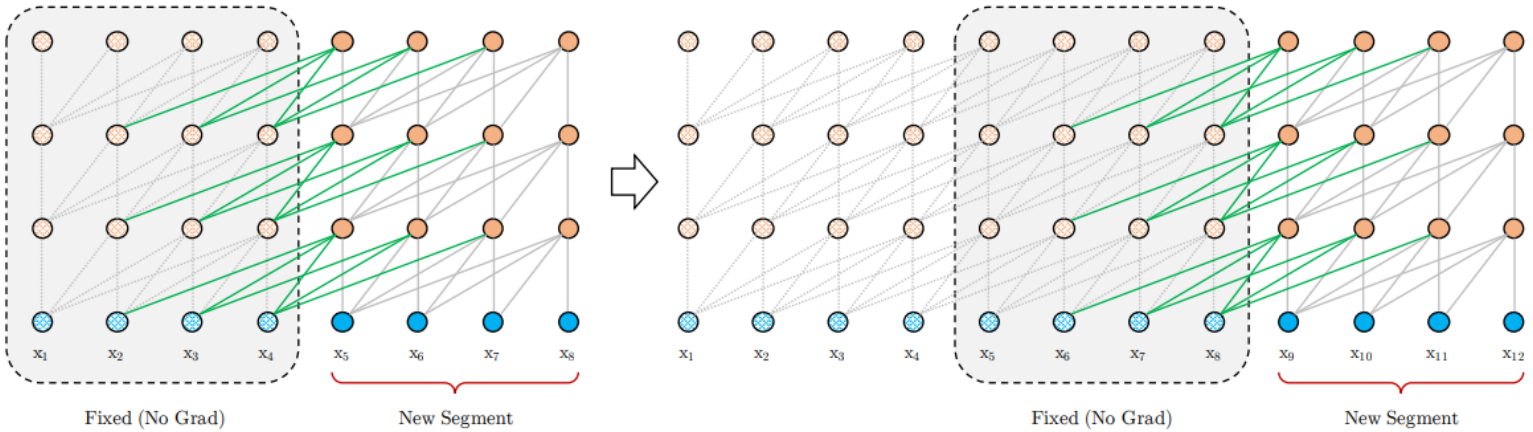


(a) Training phase.

Segment-Level Recurrence with State Reuse

Recurrence mechanism : Comments

- Can not flow gradient.
  - Works like a memory.
- Receptive field increased by memory.
- Main importance is **connectivity** with previous segment.



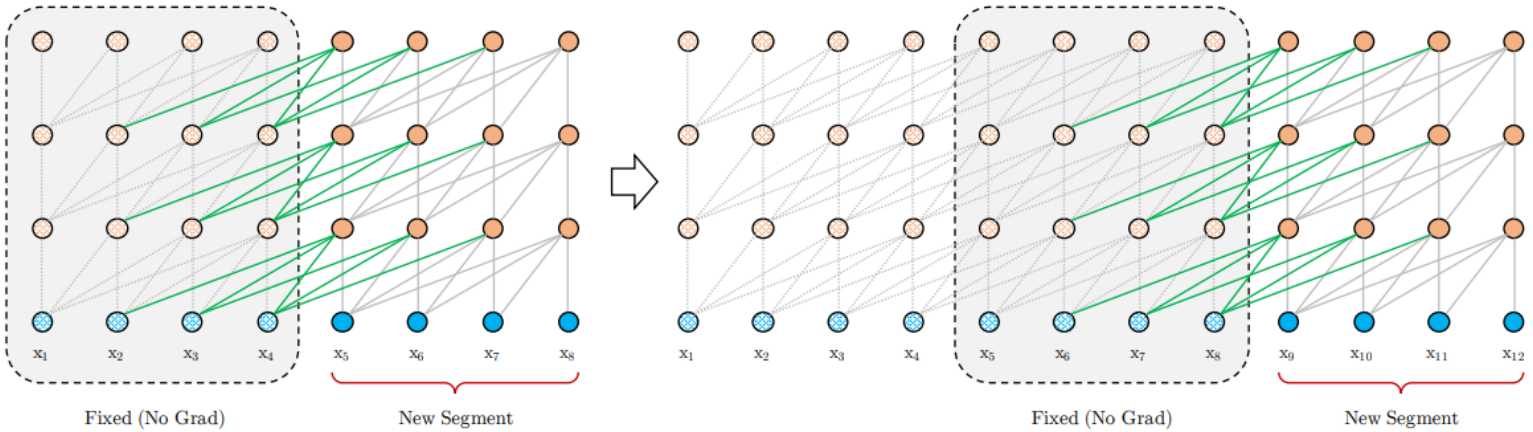
(a) Training phase.

FIGURE – Illustration of the Transformer-XL model with a **segment length 4**.

Segment-Level Recurrence with State Reuse

Recurrence mechanism : After first segment computation, from second segment, each hidden layer receives,

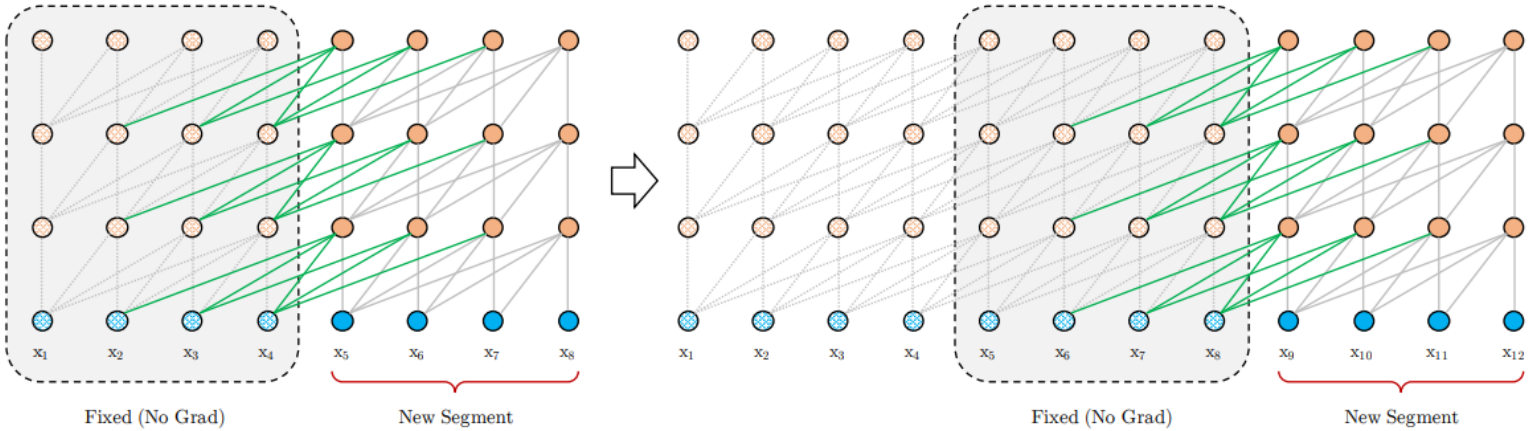
- notice that the recurrent dependency between $h_{\tau+1}^{n}$ and h_{τ}^{n-1} shifts one layer downwards per-segment, which differs from the same-layer recurrence in conventional RNN-LMs.
- Consequently, the largest possible dependency length grows linearly w.r.t. the number of layers as well as the segment length, i.e., $O(N \times L)$, $N = 3, L = 4$



Segment-Level Recurrence with State Reuse

Recurrence mechanism : After first segment computation, from second segment, each hidden layer receives,

- significantly faster evaluation.
- during evaluation, the representations from the previous segments can be reused.



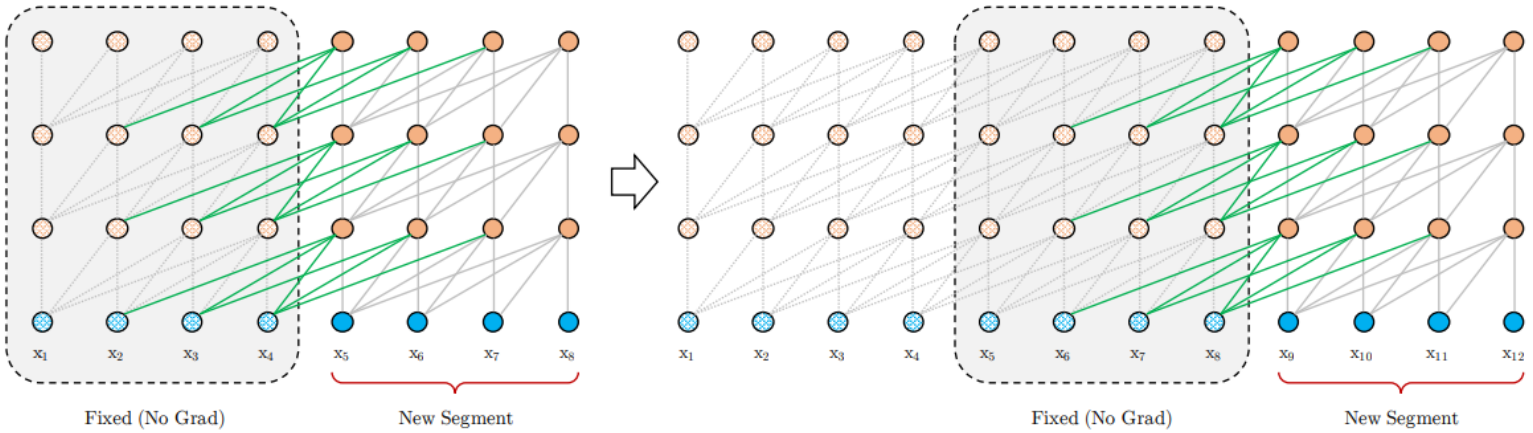
(a) Training phase.

FIGURE – Illustration of the Transformer-XL model with a **segment length 4**.

Segment-Level Recurrence with State Reuse

Recurrence mechanism : After first segment computation, from second segment, each hidden layer receives,

- the recurrence scheme does not need to be restricted to only the previous segment
- we can cache as many previous segments as the GPU memory allows, and reuse all of them as the extra context when processing the current segment.

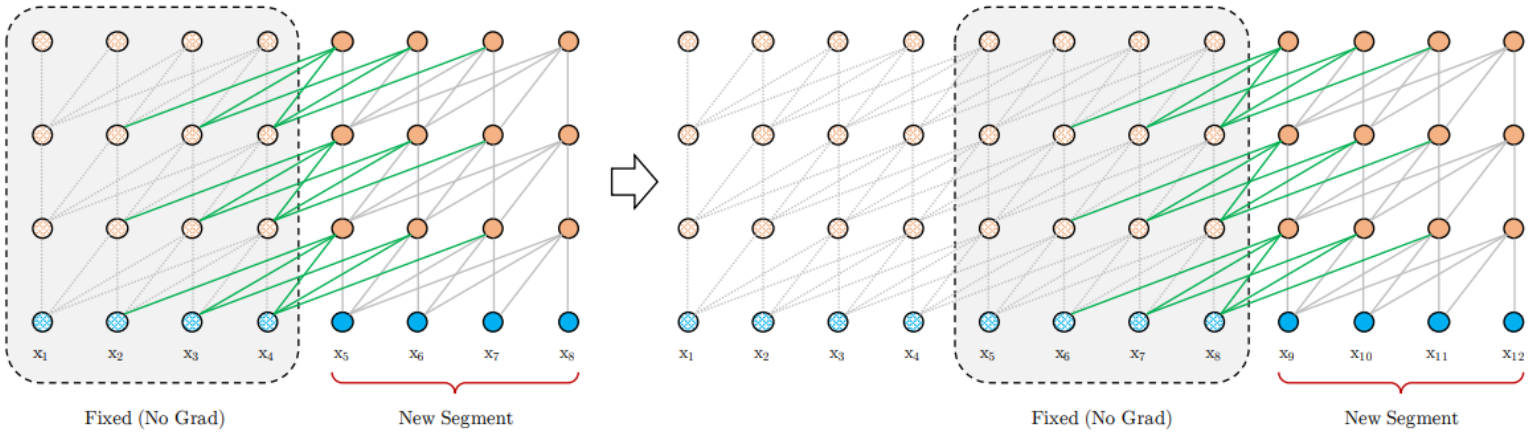


(a) Training phase.

Segment-Level Recurrence with State Reuse

Recurrence mechanism : After first segment computation, from second segment, each hidden layer receives,

- We can cache a predefined length-M old hidden states spanning (possibly) multiple segments, and refer to them as the memory $m_\tau^n \in R^{M \times d}$
- M = segment length used in the experiment.
- Increase it by multiple times during evaluation.

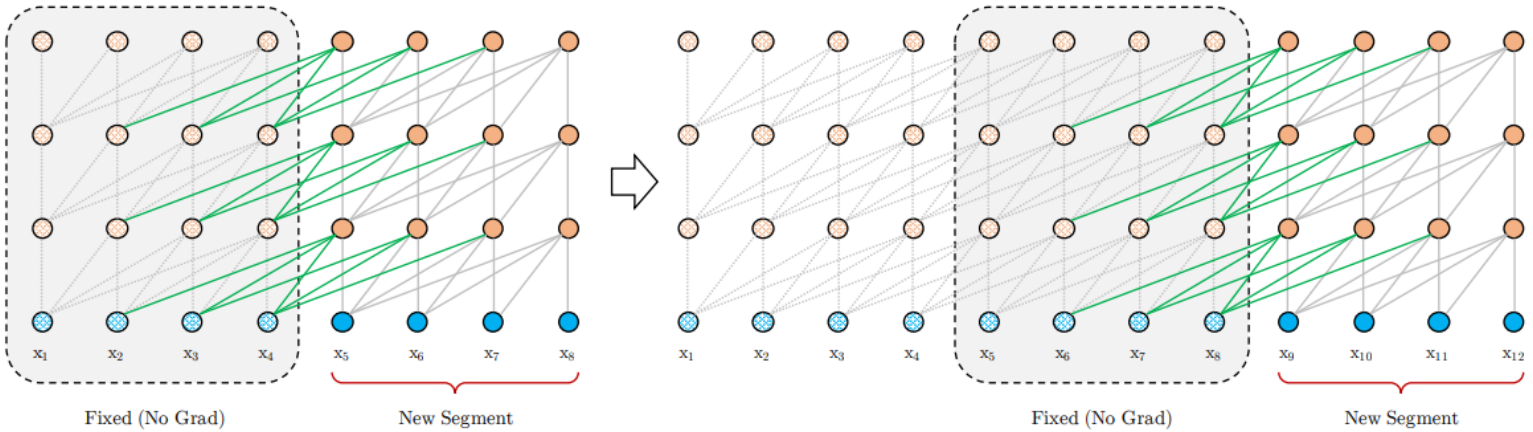


(a) Training phase.

Segment-Level Recurrence with State Reuse

Recurrence mechanism :

- MEMORY AUGMENTED RECURRENCE.
- NOT GRADIENT BASED.



(a) Training phase.

FIGURE – Illustration of the Transformer-XL model with a **segment length 4**.

Segment-Level Recurrence with State Reuse

Recurrence mechanism : Equations τ-th segment of length L,

$$S_\tau = [x_{\tau,1}, x_{\tau,2}, x_{\tau,3}, ..., x_{\tau,L}]$$

Denoting the n-th layer hidden state sequence produced for the τ -th segment s_τ by $h^n_{\tau+1} \in R^{L\times D}$ where d is the hidden dimension.

$$\widetilde{\mathbf{h}}^{n-1}_{\tau+1} = \left[\text{SG}(\mathbf{h}^{n-1}_{\tau}) \circ \mathbf{h}^{n-1}_{\tau+1}\right],$$
$$\mathbf{q}^n_{\tau+1}, \mathbf{k}^n_{\tau+1}, \mathbf{v}^n_{\tau+1} = \mathbf{h}^{n-1}_{\tau+1}\mathbf{W}^\top_q, \widetilde{\mathbf{h}}^{n-1}_{\tau+1}\mathbf{W}^\top_k, \widetilde{\mathbf{h}}^{n-1}_{\tau+1}\mathbf{W}^\top_v,$$
$$\mathbf{h}^n_{\tau+1} = \text{Transformer-Layer}\left(\mathbf{q}^n_{\tau+1}, \mathbf{k}^n_{\tau+1}, \mathbf{v}^n_{\tau+1}\right).$$

- SG$(\cdot)$ stands for stop-gradient
- $[h_u o h_v]$ indicates the concatenation of two hidden sequences along the length dimension.

Transformer architecture

PE at the embedding level : absolute position within a segment.

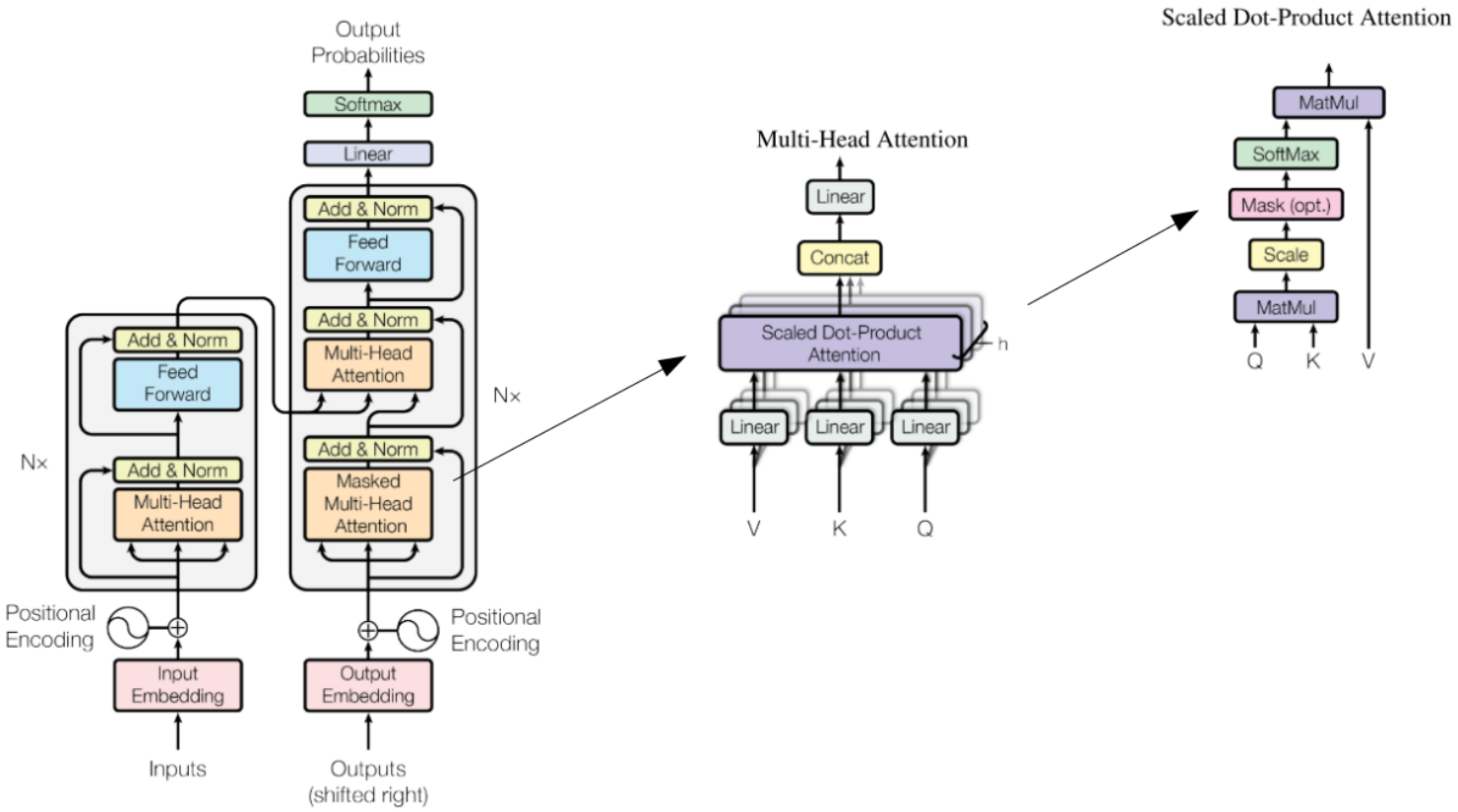
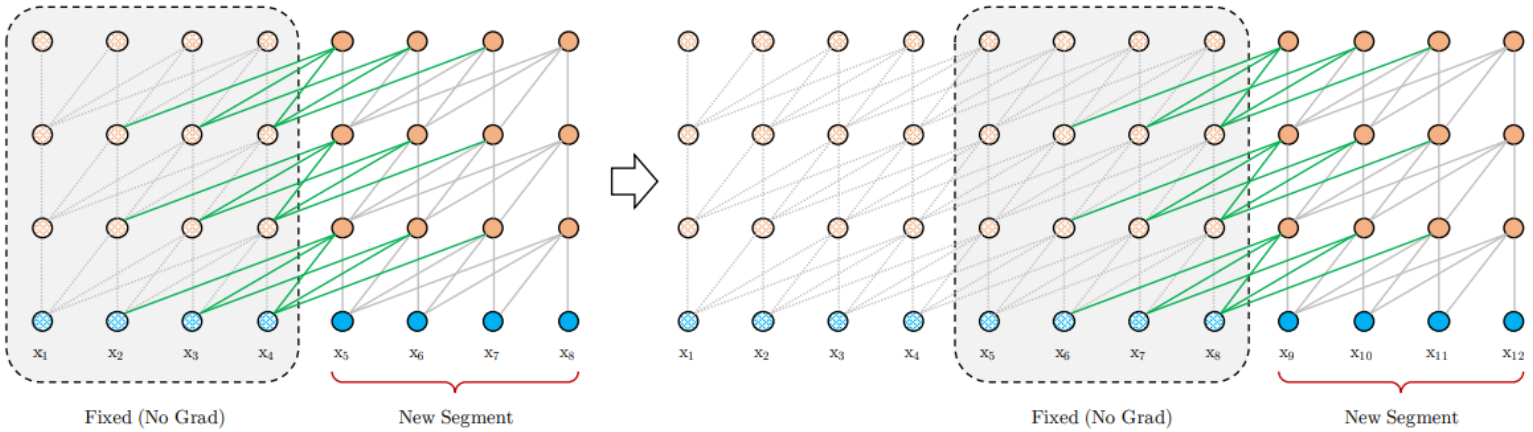


FIGURE – transformer architecture.

Relative Positional Encodings

- Recurrent Memory need to be position-ed
- Need attention level positional encoding.
- Proposed in "Self-Attention with Relative Position Representations" – shaw er al from google.



(a) Training phase.

FIGURE – Illustration of the Transformer-XL model with a **segment length 4**.

Relative Positional Encodings : :Motivation

- When a query vector attends on the key vectors it does not need to know the absolute position of each key vector to identify the temporal order of the segment.
- temporal order (? ?) : : arrangement of events in time.
- It suffices to know the relative distance between each key vector $k_{\tau,j}$ and itself $q_{\tau,i}$, i.e. $i - j$.
- We can create a set of relative positional encodings $\mathbf{R} \in R^{L_{max} \times d}$, where the i-th row R_i indicates a relative distance of i between two positions.
- By injecting the relative distance dynamically into the attention score, the query vector can easily distinguish the representations of $x_{\tau,j}$ and $x_{\tau+1,j}$ from their different distances.
- Not same as Shaw et al (Used in MT), Huang et al (used in music) but important. – MOTIVATED, here different derivation.

Positional Encodings

According to Vaswani et al,

$$A_{i,j}^{abs} = (E_{x_i} + U_i)^T W_q^T W_k (E_{x_j} + U_j)$$

$$\mathbf{A}_{i,j}^{\text{abs}} = \underbrace{\mathbf{E}_{x_i}^\top \mathbf{W}_q^\top \mathbf{W}_k \mathbf{E}_{x_j}}_{(a)} + \underbrace{\mathbf{E}_{x_i}^\top \mathbf{W}_q^\top \mathbf{W}_k \mathbf{U}_j}_{(b)} + \underbrace{\mathbf{U}_i^\top \mathbf{W}_q^\top \mathbf{W}_k \mathbf{E}_{x_j}}_{(c)} + \underbrace{\mathbf{U}_i^\top \mathbf{W}_q^\top \mathbf{W}_k \mathbf{U}_j}_{(d)}.$$

Positional Encodings

$$\mathbf{A}^{\text{abs}}_{i,j} = \underbrace{\mathbf{E}_{x_i}^\top \mathbf{W}_q^\top \mathbf{W}_k \mathbf{E}_{x_j}}_{(a)} + \underbrace{\mathbf{E}_{x_i}^\top \mathbf{W}_q^\top \mathbf{W}_k \mathbf{U}_j}_{(b)} + \underbrace{\mathbf{U}_i^\top \mathbf{W}_q^\top \mathbf{W}_k \mathbf{E}_{x_j}}_{(c)} + \underbrace{\mathbf{U}_i^\top \mathbf{W}_q^\top \mathbf{W}_k \mathbf{U}_j}_{(d)}.$$

for two different positon i, j, (below, content i means content at position i).

- (a) contributes to attention between content i and j.
- (b) contributes to attention between content i and positional embedding j.
- (c) contributes to attention between positional embedding i and content i.
- (d) contributes to attention between positional embedding i and positional embedding j.

Positional Encodings

$$\mathbf{A}^{\text{rel}}_{i,j} = \underbrace{\mathbf{E}^{\top}_{x_i}\mathbf{W}^{\top}_{q}\mathbf{W}_{k,E}\mathbf{E}_{x_j}}_{(a)} + \underbrace{\mathbf{E}^{\top}_{x_i}\mathbf{W}^{\top}_{q}\mathbf{W}_{k,R}\mathbf{R}_{i-j}}_{(b)}$$
$$+ \underbrace{u^{\top}\mathbf{W}_{k,E}\mathbf{E}_{x_j}}_{(c)} + \underbrace{v^{\top}\mathbf{W}_{k,R}\mathbf{R}_{i-j}}_{(d)}.$$

- Replace all appearances of the absolute positional embedding U_j for computing key vectors in term (b) and (d) with its relative counterpart R_{i-j}.
- This essentially reflects the prior that only the relative distance matters for where to attend.
- Note that R is a **sinusoid encoding matrix** (Very important).

Positional Encodings

$$\mathbf{A}^{\text{rel}}_{i,j} = \underbrace{\mathbf{E}^{\top}_{x_i}\mathbf{W}^{\top}_{q}\mathbf{W}_{k,E}\mathbf{E}_{x_j}}_{(a)} + \underbrace{\mathbf{E}^{\top}_{x_i}\mathbf{W}^{\top}_{q}\mathbf{W}_{k,R}\mathbf{R}_{i-j}}_{(b)}$$
$$+ \underbrace{u^{\top}\mathbf{W}_{k,E}\mathbf{E}_{x_j}}_{(c)} + \underbrace{v^{\top}\mathbf{W}_{k,R}\mathbf{R}_{i-j}}_{(d)}.$$

- Replace all appearances of the absolute positional embedding U_j for computing key vectors in term (b) and (d) with its relative counterpart $Ri - j$.
- This essentially reflects the prior that only the relative distance matters for where to attend.
- Note that R is a **sinusoid encoding matrix** **without learnable parameters.**

Positional Encodings

$$\mathbf{A}^{\text{rel}}_{i,j} = \underbrace{\mathbf{E}^{\top}_{x_i}\mathbf{W}^{\top}_{q}\mathbf{W}_{k,E}\mathbf{E}_{x_j}}_{(a)} + \underbrace{\mathbf{E}^{\top}_{x_i}\mathbf{W}^{\top}_{q}\mathbf{W}_{k,R}\mathbf{R}_{i-j}}_{(b)}$$
$$+ \underbrace{u^{\top}\mathbf{W}_{k,E}\mathbf{E}_{x_j}}_{(c)} + \underbrace{v^{\top}\mathbf{W}_{k,R}\mathbf{R}_{i-j}}_{(d)}.$$

- Introduces a trainable parameter $u \in R^d$ to replace the query $U_i^T W_q^T$ in terms of (c).
- **Do we really need positional information for query vector ?**

From paper

Since the query vector is the same for all query positions, it suggests that the attentive bias towards different words should remain the same regardless of the query position.

Positional Encodings

$$\mathbf{A}^{\text{rel}}_{i,j} = \underbrace{\mathbf{E}^{\top}_{x_i}\mathbf{W}^{\top}_{q}\mathbf{W}_{k,E}\mathbf{E}_{x_j}}_{(a)} + \underbrace{\mathbf{E}^{\top}_{x_i}\mathbf{W}^{\top}_{q}\mathbf{W}_{k,R}\mathbf{R}_{i-j}}_{(b)}$$
$$+ \underbrace{u^{\top}\mathbf{W}_{k,E}\mathbf{E}_{x_j}}_{(c)} + \underbrace{v^{\top}\mathbf{W}_{k,R}\mathbf{R}_{i-j}}_{(d)}.$$

- $W_{k,E}$ and $W_{k,R}$ for producing the content-based key vectors and location-based.
- Different learning parameters for content and location.
- Only for key vectors has biases.

Positional Encodings

$$\mathbf{A}^{\text{rel}}_{i,j} = \underbrace{\mathbf{E}^{\top}_{x_i}\mathbf{W}^{\top}_{q}\mathbf{W}_{k,E}\mathbf{E}_{x_j}}_{(a)} + \underbrace{\mathbf{E}^{\top}_{x_i}\mathbf{W}^{\top}_{q}\mathbf{W}_{k,R}\mathbf{R}_{i-j}}_{(b)}$$
$$+ \underbrace{u^{\top}\mathbf{W}_{k,E}\mathbf{E}_{x_j}}_{(c)} + \underbrace{v^{\top}\mathbf{W}_{k,R}\mathbf{R}_{i-j}}_{(d)}.$$

- (a) represents content based addressing.
- (b) captures a content dependent positional bias.
- (c) governs a global content bias.
- (d) encodes a global positional bias.

Positional Encodings

$$\mathbf{A}^{\text{rel}}_{i,j} = \underbrace{\mathbf{E}^{\top}_{x_i}\mathbf{W}^{\top}_{q}\mathbf{W}_{k,E}\mathbf{E}_{x_j}}_{(a)} + \underbrace{\mathbf{E}^{\top}_{x_i}\mathbf{W}^{\top}_{q}\mathbf{W}_{k,R}\mathbf{R}_{i-j}}_{(b)}$$
$$+ \underbrace{u^{\top}\mathbf{W}_{k,E}\mathbf{E}_{x_j}}_{(c)} + \underbrace{v^{\top}\mathbf{W}_{k,R}\mathbf{R}_{i-j}}_{(d)}.$$

- Shaw et al. (2018) (Self-Attention with Relative Position Representations) only has terms (a) and (b).
- Shaw et al. (2018) merge the multiplication $W_k R$ into a single trainable matrix $\hat{R}$.
- Why sinusoidal ? ? ?
- If sinusoidal how this inductive bias helps in language model rather than learning learnable parameter ?

Equations of the model

For $n = 1, 2, ..., N$,

$$\widetilde{\mathbf{h}}_{\tau}^{n-1} = \left[\text{SG}(\mathbf{m}_{\tau}^{n-1}) \circ \mathbf{h}_{\tau}^{n-1}\right]$$

$$\mathbf{q}_{\tau}^{n}, \mathbf{k}_{\tau}^{n}, \mathbf{v}_{\tau}^{n} = \mathbf{h}_{\tau}^{n-1}\mathbf{W}_{q}^{n\top}, \widetilde{\mathbf{h}}_{\tau}^{n-1}\mathbf{W}_{k,E}^{n\top}, \widetilde{\mathbf{h}}_{\tau}^{n-1}\mathbf{W}_{v}^{n\top}$$

$$\mathbf{A}_{\tau,i,j}^{n} = \mathbf{q}_{\tau,i}^{n\top}\mathbf{k}_{\tau,j}^{n} + \mathbf{q}_{\tau,i}^{n\top}\mathbf{W}_{k,R}^{n}\mathbf{R}_{i-j} + u^{\top}\mathbf{k}_{\tau,j} + v^{\top}\mathbf{W}_{k,R}^{n}\mathbf{R}_{i-j}$$

$$\mathbf{a}_{\tau}^{n} = \text{Masked-Softmax}(\mathbf{A}_{\tau}^{n})\mathbf{v}_{\tau}^{n}$$

$$\mathbf{o}_{\tau}^{n} = \text{LayerNorm}(\text{Linear}(\mathbf{a}_{\tau}^{n}) + \mathbf{h}_{\tau}^{n-1})$$

$$\mathbf{h}_{\tau}^{n} = \text{Positionwise-Feed-Forward}(\mathbf{o}_{\tau}^{n})$$

Thoughts

Question

You still believe TRANSFORMER is a sequence model ?

Design you transformer on your own task.

Numbers

- **TransformerXL learns dependency that is 80% longer than RNNs and 450% longer than vanilla Transformers.**
- Achieves better performance on both short and long sequences.
- **Up to 1,800+ times faster than vanilla Transformers during evaluation.**
- We improve the state-of-the-art results of bpc/perplexity to 0.99 on enwiki8, 1.08 on text8, 18.3 on WikiText-103, 21.8 on One Billion Word, and 54.5 on Penn Treebank (without finetuning) evaluation.
- When trained only on WikiText-103, Transformer-XL manages to generate reasonably coherent, novel text articles with thousands of tokens.

Numbers

- **TransformerXL learns dependency that is 80% longer than RNNs and 450% longer than vanilla Transformers.**
- Achieves better performance on both short and long sequences.
- **Up to 1,800+ times faster than vanilla Transformers during evaluation.**
- We improve the state-of-the-art results of bpc/perplexity to 0.99 on enwiki8, 1.08 on text8, 18.3 on WikiText-103, 21.8 on One Billion Word, and 54.5 on Penn Treebank (without finetuning) evaluation.
- When trained only on WikiText-103, Transformer-XL manages to generate reasonably coherent, novel text articles with thousands of tokens.

Numbers

- **TransformerXL learns dependency that is 80% longer than RNNs and 450% longer than vanilla Transformers.**
- Achieves better performance on both short and long sequences.
- **Up to 1,800+ times faster than vanilla Transformers during evaluation.**
- We improve the state-of-the-art results of bpc/perplexity to 0.99 on enwiki8, 1.08 on text8, 18.3 on WikiText-103, 21.8 on One Billion Word, and 54.5 on Penn Treebank (without finetuning) evaluation.
- When trained only on WikiText-103, Transformer-XL manages to generate reasonably coherent, novel text articles with thousands of tokens.

Numbers

- **TransformerXL learns dependency that is 80% longer than RNNs and 450% longer than vanilla Transformers.**
- Achieves better performance on both short and long sequences.
- **Up to 1,800+ times faster than vanilla Transformers during evaluation.**
- We improve the state-of-the-art results of bpc/perplexity to 0.99 on enwiki8, 1.08 on text8, 18.3 on WikiText-103, 21.8 on One Billion Word, and 54.5 on Penn Treebank (without finetuning) evaluation.
- When trained only on WikiText-103, Transformer-XL manages to generate reasonably coherent, novel text articles with thousands of tokens.

Numbers

- **TransformerXL learns dependency that is 80% longer than RNNs and 450% longer than vanilla Transformers.**
- Achieves better performance on both short and long sequences.
- **Up to 1,800+ times faster than vanilla Transformers during evaluation.**
- We improve the state-of-the-art results of bpc/perplexity to 0.99 on enwiki8, 1.08 on text8, 18.3 on WikiText-103, 21.8 on One Billion Word, and 54.5 on Penn Treebank (without finetuning) evaluation.
- When trained only on WikiText-103, Transformer-XL manages to generate reasonably coherent, novel text articles with thousands of tokens.

Relative Effective Context Length

- **TransformerXL learns dependency that is 80% longer than RNNs and 450% longer than vanilla Transformers.**

How do you measure that. RNN still powerfull than vanilla transformer ?

- Khandelwal et al. (2018) proposed a method to evaluate the Effective Context Length (ECL) of a sequence model.
- ECL is the longest length to which increasing the context span would lead to a gain more than a threshold.
- ECL ignores the fact ....
- Proposes Relative Effective Context Length (RECL) which solves the problem.
- Both the recurrence mechanism and positional encodings contribute to a longer RECL.

Visualization

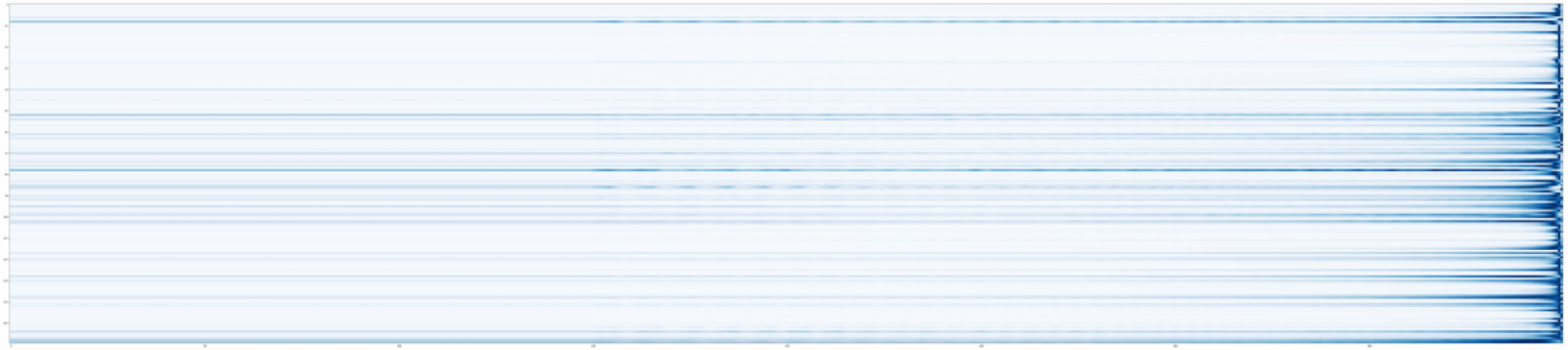
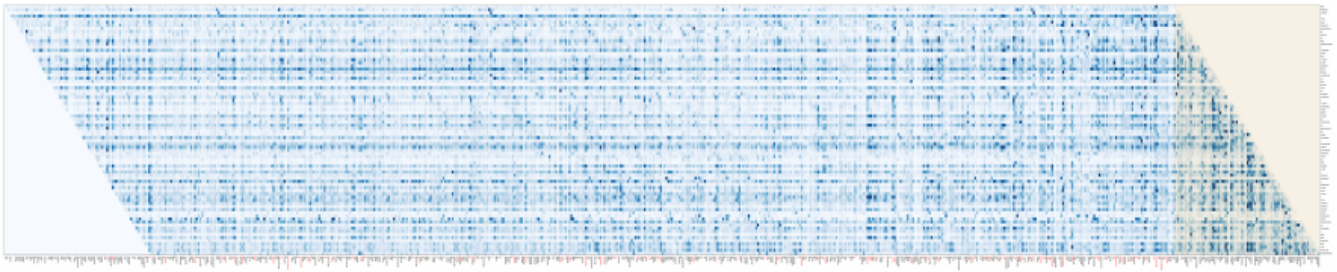
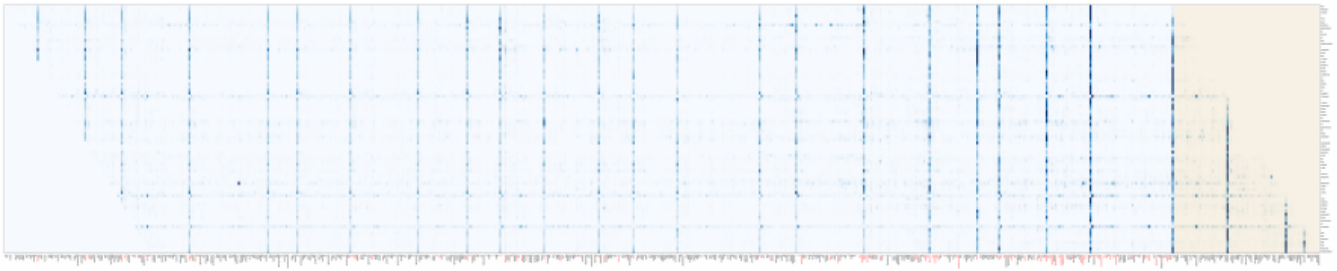


Figure 5: Average attention over the previous 640 tokens, where each row corresponds to a attention head and each column corresponds to a relative location. There are totally 160 attention heads, and every 10 heads come from a single layer. Darker colors indicate higher values.

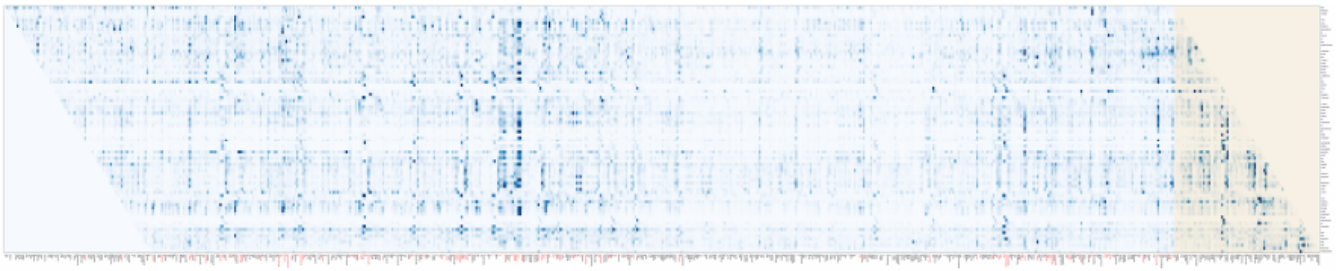
Visualization



(a) Head 8 from layer 1.



(b) Head 78 from layer 8.



(c) Head 158 from layer 16.

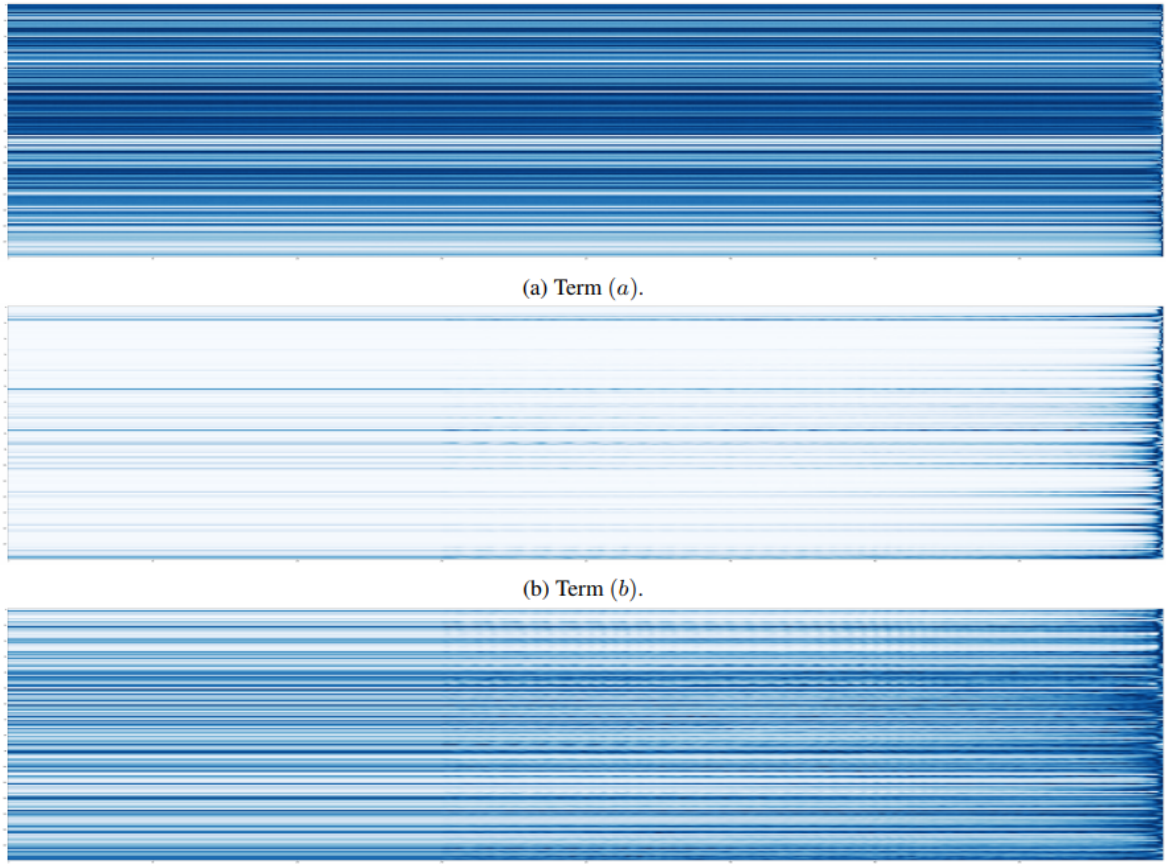
Figure 6: Visualization of the three heads with a wide attention range. Each row corresponds to a target location/token and each column corresponds to a context location/token. Tokens in the memory that have top 20% attention values are highlighted in red.

Positional Encodings

$$\mathbf{A}^{\text{rel}}_{i,j} = \underbrace{\mathbf{E}^{\top}_{x_i}\mathbf{W}^{\top}_{q}\mathbf{W}_{k,E}\mathbf{E}_{x_j}}_{(a)} + \underbrace{\mathbf{E}^{\top}_{x_i}\mathbf{W}^{\top}_{q}\mathbf{W}_{k,R}\mathbf{R}_{i-j}}_{(b)} + \underbrace{u^{\top}\mathbf{W}_{k,E}\mathbf{E}_{x_j}}_{(c)} + \underbrace{v^{\top}\mathbf{W}_{k,R}\mathbf{R}_{i-j}}_{(d)}.$$

- (a) represents content based addressing.
- (b) captures a content dependent positional bias.
- (c) governs a global content bias.
- (d) encodes a global positional bias.

Visualization



(a) Term (a).

(b) Term (b).

(c) Term (d).

Figure 7: Visualization of the three terms in computing the attention score. Each row corresponds to a attention head and each column corresponds to a relative location.

Transformer-XL Summary

Transformer-XL, XL means extra long.

It Proposes,

- a segment-level (text stream) recurrence mechanism.
- A novel positional encoding scheme.
- Proposes a new metric Relative Effective Context Length (RECL).

It Solves,

- Resolves the context fragmentation problem.
- Propose learning dependency beyond a fixed length without disrupting **temporal coherence**.
- Better capturing longer-term dependency.
- Better sequential feature injection.
- Measure sequential performance in a encoder.

Current trends and takeaways

A lot of invention or growth follows,

Moore's law

The number of transistors in a dense integrated circuit doubles about every two years.

Current trends and takeaways,

- **Data is the king** –> XLnet.
- Alec Radford : Scrape the whole internet. —> (OMG -> OMM)
- **Exponential growth** of NVIDIA GPU technology is not able to cope up.
- What is the growth of AI publication and technology ? (not certainly exponential).

Current trends and takeaways

A lot of invention or growth follows,

Moore's law

The number of transistors in a dense integrated circuit doubles about every two years.

Current trends and takeaways,

- **Data is the king** –> XLnet.
- Alec Radford : Scrape the whole internet. —> (OMG -> OMM)
- **Exponential growth** of NVIDIA GPU technology is not able to cope up.
- What is the growth of AI publication and technology ? (not certainly exponential).

Current trends and takeaways

A lot of invention or growth follows,

Moore's law

The number of transistors in a dense integrated circuit doubles about every two years.

Current trends and takeaways,

- **Data is the king** –> XLnet.
- Alec Radford : Scrape the whole internet. —> (OMG -> OMM)
- **Exponential growth** of NVIDIA GPU technology is not able to cope up.
- What is the growth of AI publication and technology ? (not certainly exponential).

Current trends and takeaways

A lot of invention or growth follows,

Moore's law

The number of transistors in a dense integrated circuit doubles about every two years.

Current trends and takeaways,

- **Data is the king** –> XLnet.
- Alec Radford : Scrape the whole internet. —> (OMG -> OMM)
- **Exponential growth** of NVIDIA GPU technology is not able to cope up.
- What is the growth of AI publication and technology ? (not certainly exponential).

Future trend

Current trends and takeaways,

- **Data is the king**.
- Alec Radford : Scrape the whole internet. —> (OMG)
- **Exponential growth** of NVIDIA GPU technology is not able to cope up.
- What is the growth of AI publication and technology ? (not certainly exponential).

Future trend in my opinion,

- **Learn so much for so little.**
  - Does Transformer a worthy contender ?

Future trend

Current trends and takeaways,

- **Data is the king**.
- Alec Radford : Scrape the whole internet. —> (OMG)
- **Exponential growth** of NVIDIA GPU technology is not able to cope up.
- What is the growth of AI publication and technology ? (not certainly exponential).

Future trend in my opinion,

- **Learn so much for so little.**
  - Does Transformer a worthy contender ?

Scarcity

Current trends and takeaways,

- **Data is the king**.
- Alec Radford : Scrape the whole internet. —> (OMG)
- **Exponential growth** of NVIDIA GPU technology is not able to cope up.
- What is the growth of AI publication and technology ? (not certainly exponential).

Future trend,

- **Learn so much with little.**
  - Does Transformer a worthy contender ?

Scarcity,

- Need more **dataset** on more task.
  - On real life problem to solve real life problem.
  - Different type of real dataset or at least **artificially generated data-set** to comment how good is the density function estimation.
    - understand more our encoder, decoder and training mechanism.
- Need better **metric**. perplexity, BLUE etc.

Scarcity

Current trends and takeaways,

- **Data is the king**.
- Alec Radford : Scrape the whole internet. —> (OMG)
- **Exponential growth** of NVIDIA GPU technology is not able to cope up.
- What is the growth of AI publication and technology ? (not certainly exponential).

Future trend,

- **Learn so much with little.**
  - Does Transformer a worthy contender ?

Scarcity,

- Need more **dataset** on more task.
  - On real life problem to solve real life problem.
  - Different type of real dataset or at least **artificially generated data-set** to comment how good is the density function estimation.
    - understand more our encoder, decoder and training mechanism.
- Need better **metric**. perplexity, BLUE etc.

Scarcity

Current trends and takeaways,

- **Data is the king**.
- Alec Radford : Scrape the whole internet. —> (OMG)
- **Exponential growth** of NVIDIA GPU technology is not able to cope up.
- What is the growth of AI publication and technology ? (not certainly exponential).

Future trend,

- **Learn so much with little.**
  - Does Transformer a worthy contender ?

Scarcity,

- Need more **dataset** on more task.
  - On real life problem to solve real life problem.
  - Different type of real dataset or at least **artificially generated data-set** to comment how good is the density function estimation.
    - understand more our encoder, decoder and training mechanism.
- Need better **metric**. perplexity, BLUE etc.

Scarcity

Current trends and takeaways,

- **Data is the king**.
- Alec Radford : Scrape the whole internet. —> (OMG)
- **Exponential growth** of NVIDIA GPU technology is not able to cope up.
- What is the growth of AI publication and technology ? (not certainly exponential).

Future trend,

- **Learn so much with little.**
  - Does Transformer a worthy contender ?

Scarcity,

- Need more **dataset** on more task.
  - On real life problem to solve real life problem.
  - Different type of real dataset or at least **artificially generated data-set** to comment how good is the density function estimation.
    - understand more our encoder, decoder and training mechanism.
- Need better **metric**. perplexity, BLUE etc.

Scarcity

Current trends and takeaways,

- **Data is the king**.
- Alec Radford : Scrape the whole internet. —> (OMG)
- **Exponential growth** of NVIDIA GPU technology is not able to cope up.
- What is the growth of AI publication and technology ? (not certainly exponential).

Future trend,

- **Learn so much with little.**
  - Does Transformer a worthy contender ?

Scarcity,

- Need more **dataset** on more task.
  - On real life problem to solve real life problem.
  - Different type of real dataset or at least **artificially generated data-set** to comment how good is the density function estimation.
    - understand more our encoder, decoder and training mechanism.
- Need better **metric**. perplexity, BLUE etc.

Question & Where to work