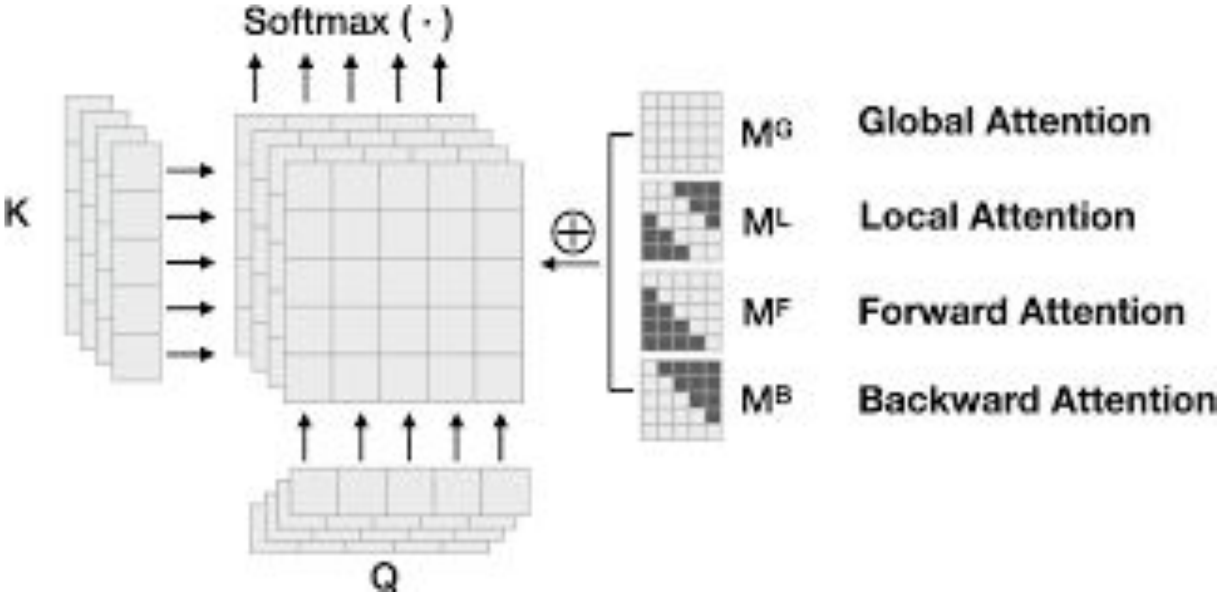
MEGABYTE: Predicting Million-byte Sequences with Multiscale Transformers

Sijuade

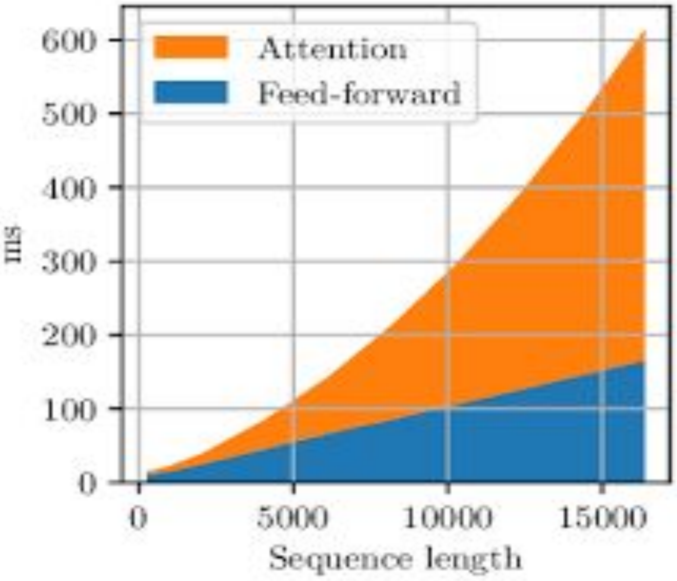
Motivation

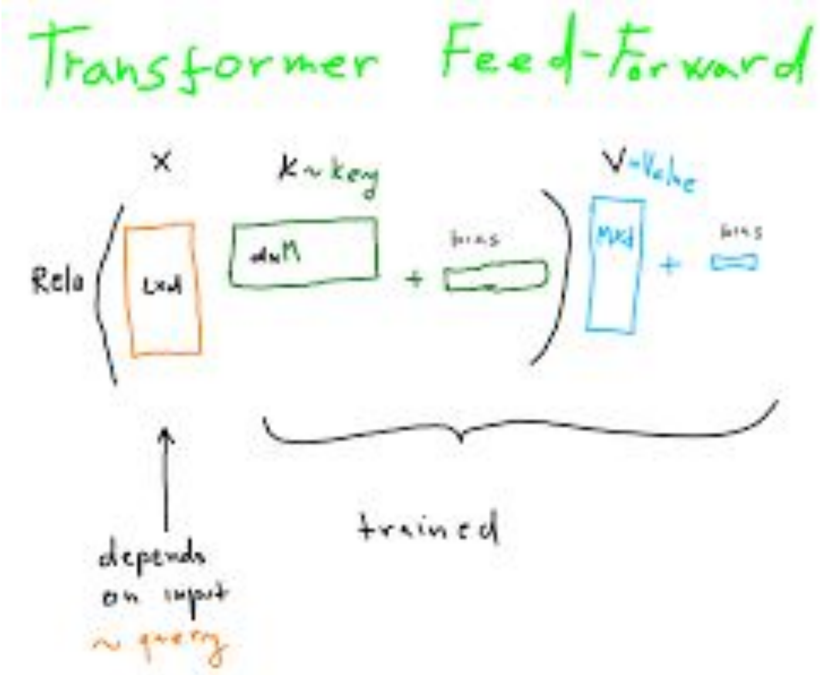
- Modeling long byte sequences
- Efficient and effective models for long sequences



Challenges

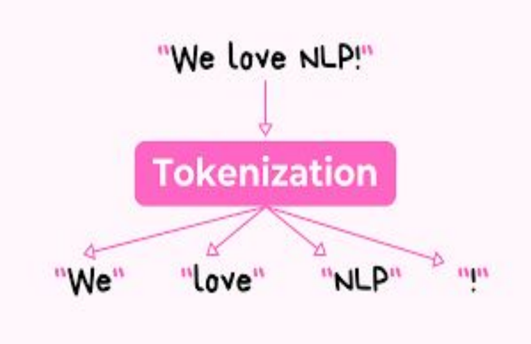
- **Quadratic cost of self-attention**:Poor scaling with long sequences
- **Feedforward networks**: Large computational overhead

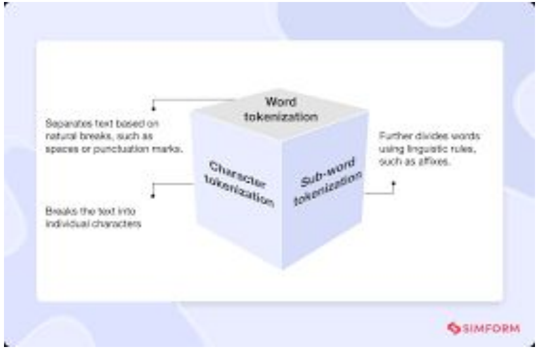


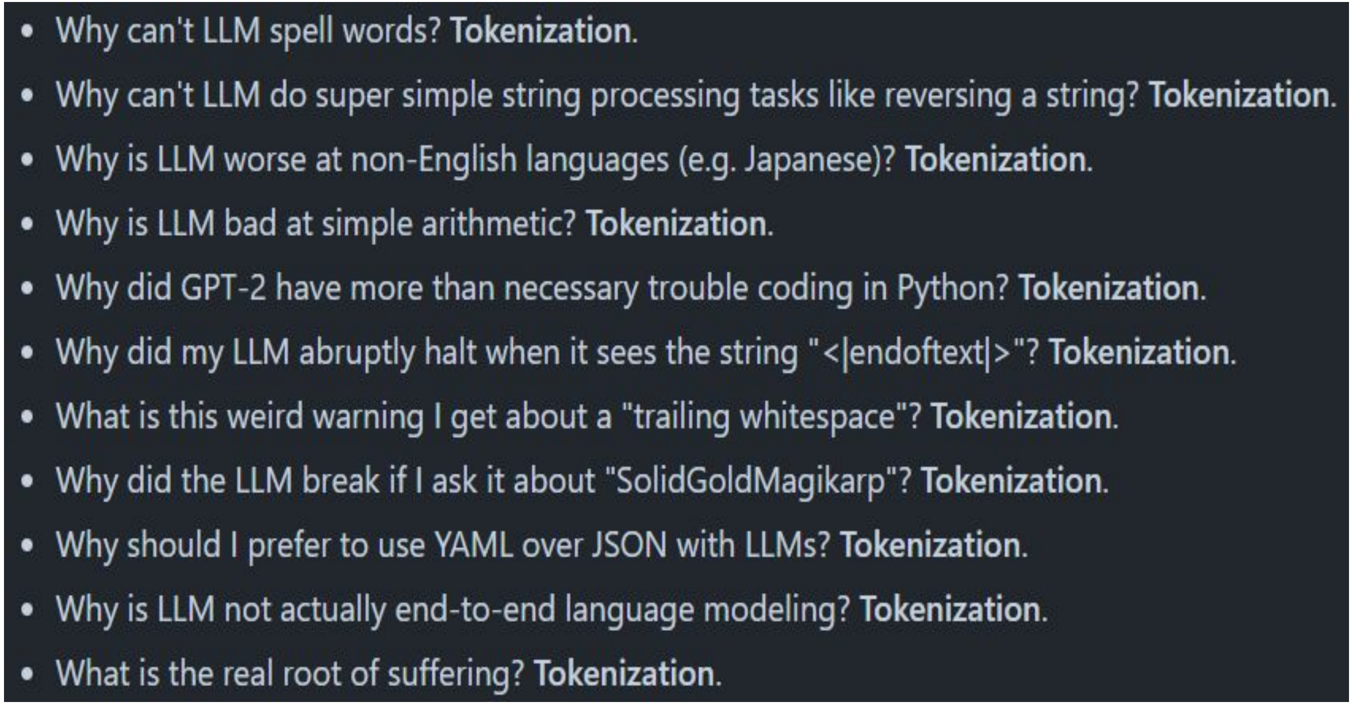


Challenges

- **Tokenization**: Introduce information loss and complexity - Multimodality
- **Contextual information**: Capture long-range dependencies and contextual information effectively




Word tokenization
Separates text based on natural breaks, such as spaces or punctuation marks.
Character tokenization
Breaks the text into individual characters
Sub-word tokenization
Further divides words using linguistic rules, such as affixes.
SIMFORM



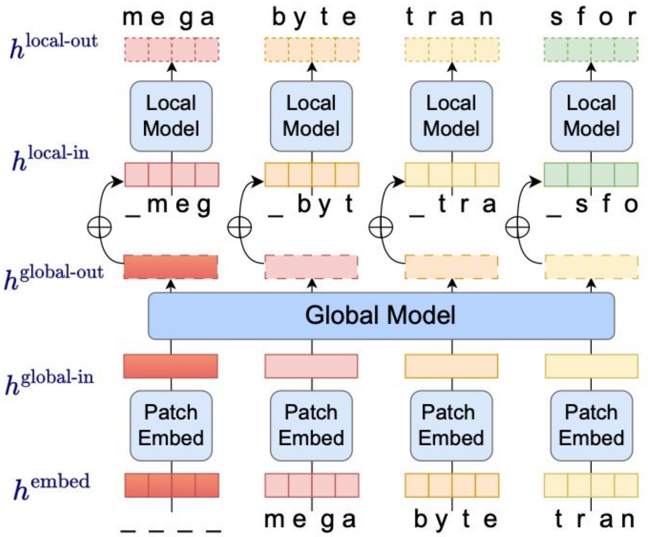
- Why can't LLM spell words? **Tokenization.**
- Why can't LLM do super simple string processing tasks like reversing a string? **Tokenization.**
- Why is LLM worse at non-English languages (e.g. Japanese)? **Tokenization.**
- Why is LLM bad at simple arithmetic? **Tokenization.**
- Why did GPT-2 have more than necessary trouble coding in Python? **Tokenization.**
- Why did my LLM abruptly halt when it sees the string "<|endoftext|>"? **Tokenization.**
- What is this weird warning I get about a "trailing whitespace"? **Tokenization.**
- Why did the LLM break if I ask it about "SolidGoldMagikarp"? **Tokenization.**
- Why should I prefer to use YAML over JSON with LLMs? **Tokenization.**
- Why is LLM not actually end-to-end language modeling? **Tokenization.**
- What is the real root of suffering? **Tokenization.**

MEGABYTE Overview & Benefits

- Reduced cost for longer sequences and larger models
- Faster sequence generation due to parallel processing of patches
- Sub-quadratic self-attention, larger and more expressive feedforward layers, greater parallelism during generation

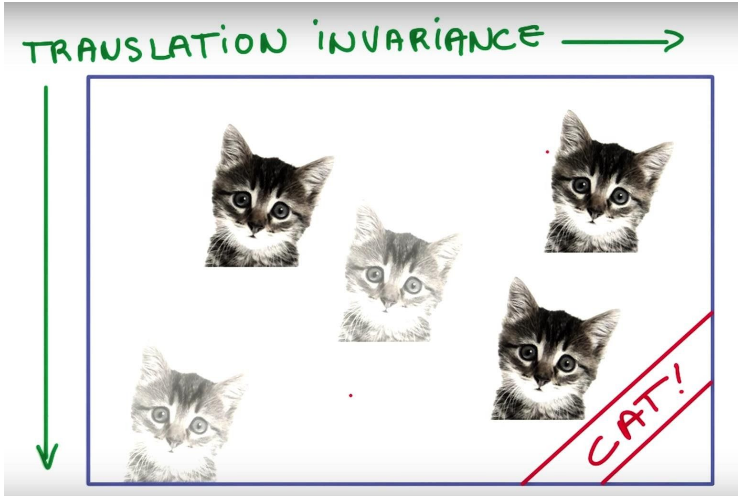
Components

- **Patch Embedder**: Concatenate bytes into patches
- **Global Model**: Inter Patch transformer processing
- **Local Model:** Intra-patch transformer processing



Extensions

- **Convolutional Patch Encoder**: Translation invariance with fixed patches
- **Strided Inference**: Performance drop at patch edges
- **Cross-Patch Attention**: Conditioning on elements from the previous patch



Cat, near right side

Cat, near left side

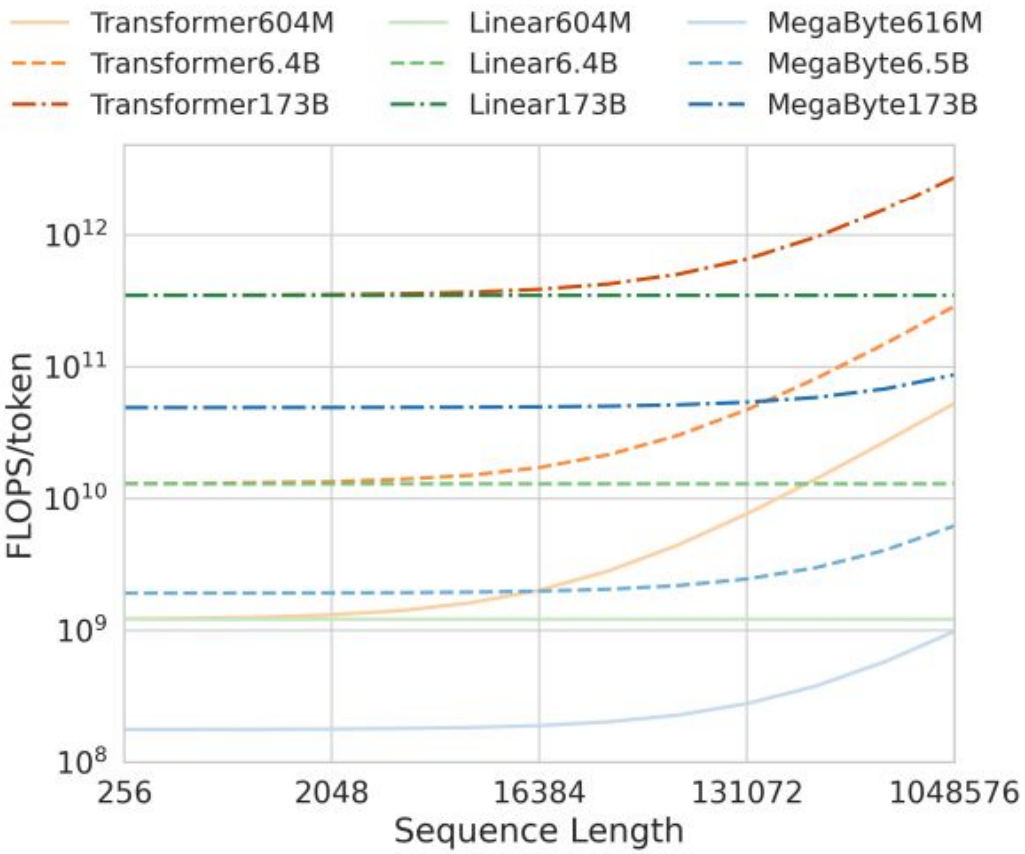
Efficiency - Training

Standard Transformer:
Attention: O(T^2*d)
FF: O(T*d^2)

T -> Sequence Length
d -> model dim

Global Model:
Attention: O((T/P)^2 * d)
FF: O((T/P) * d^2)

Local Model:
Attention: O(p^2*d)
FF: O(p*d^2)



Efficiency

- Parallelism during decoding
- Multiple patches can be processed simultaneously
- Reduced computational overhead leads to faster generation times