Special Topics: Machine Learning (ML) for Networking

COL867

Holi, 2025

Foundation Model

Tarun Mangla

ML for Networks

Module 1: Case studies of specific network learning tasks

Module 2: Task-agnostic automatic ML pipelines for networks

- Generalized data representation
- **Generalized ML model(s)**

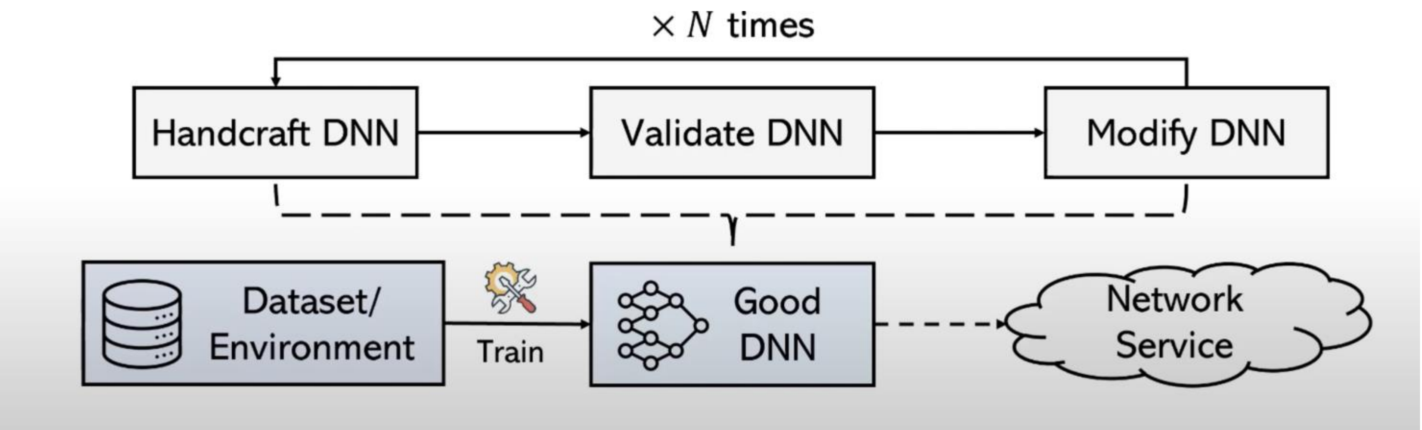
Module 3: Beyond feature engineering and modeling

One model to rule them all

- Foundation models: trained on large corpora of unlabeled data using self-supervised learning
- Adapted to different downstream tasks with minimal tuning

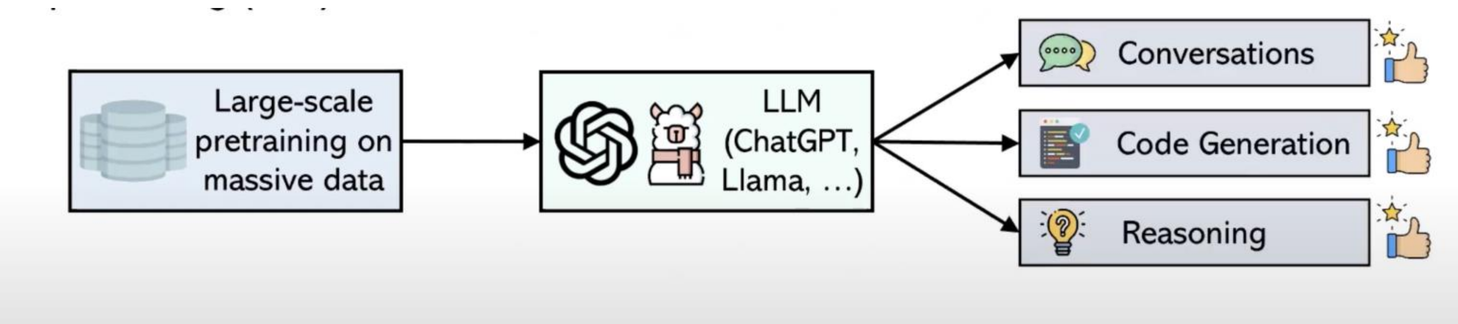
Why Foundation Models

- Reduced manual effort in data representation and modeling



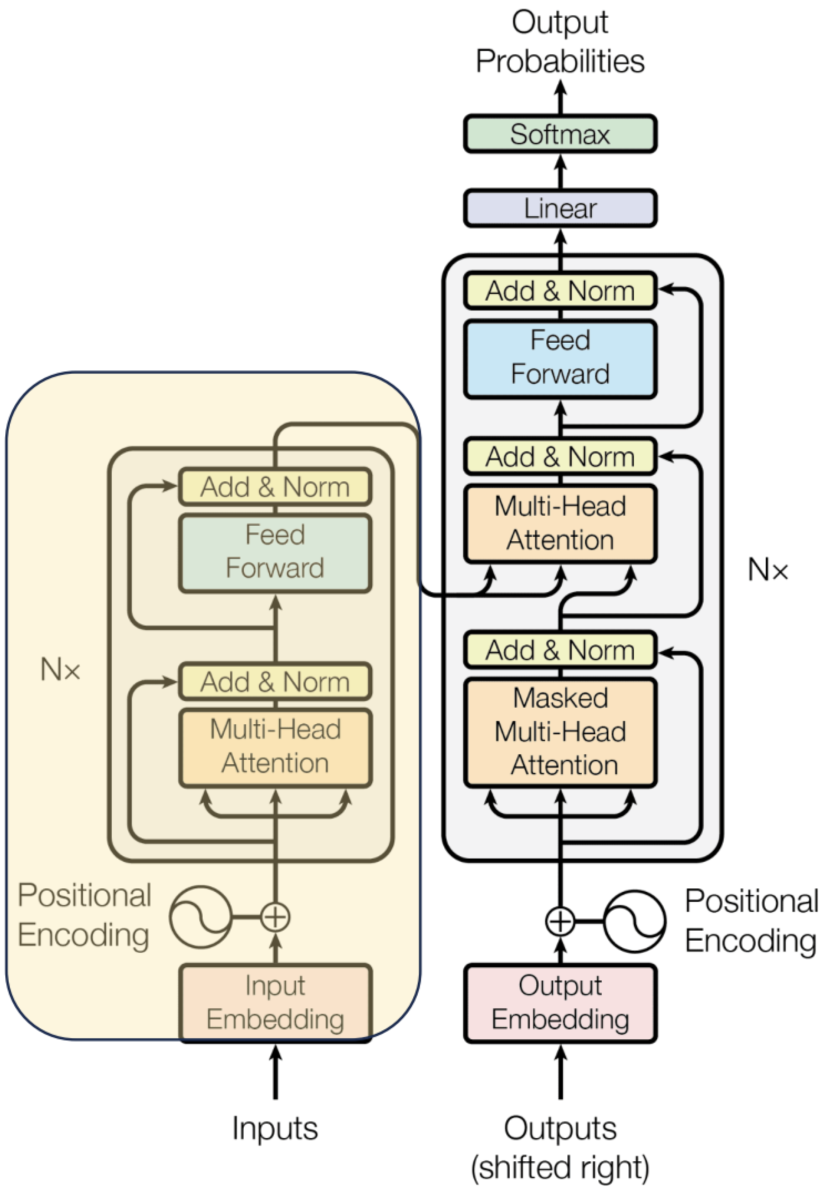
Why Foundation Models

- Reduced manual effort in data representation and modeling
- Requires lesser labeled data
- Emergent abilities



Example from NLP:
Bidirectional Encoder Representation (BERT)

- Goal: Build effective language representation that can be applied for a variety of downstream tasks
- Pre-trained on a large corpus
- Uses transformer as the underlying model
- Two important aspects:
  - Tokenization
  - Pre-training task



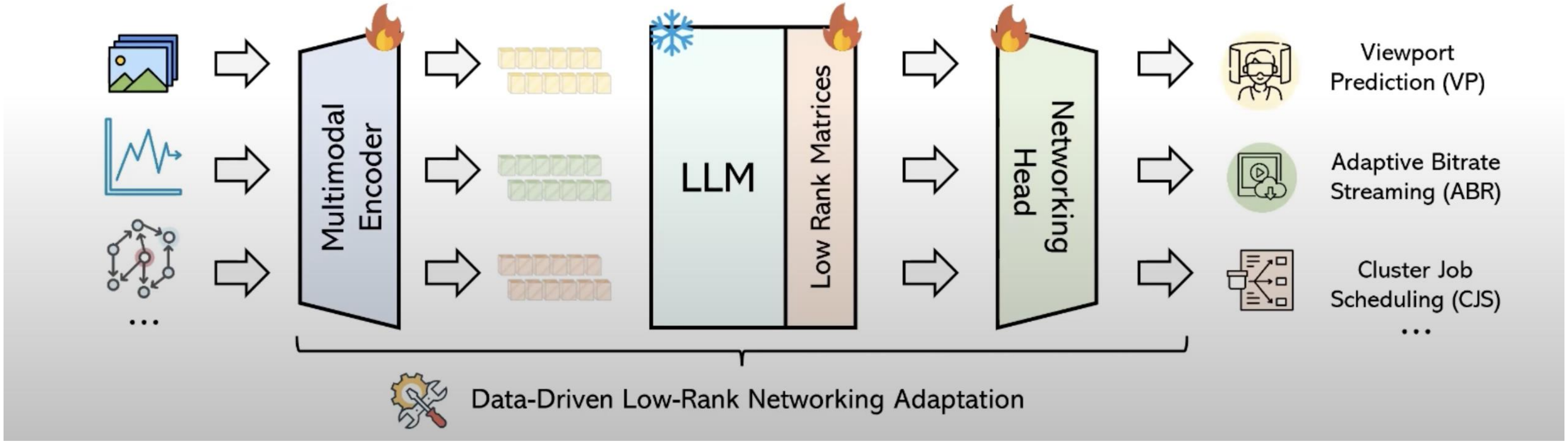
How to build a transformer for network data?

How to Build a Foundation Model for Networks?

Two different paradigms:

- Use a pre-trained large-language model
  - netLLM
- Build a foundation model from scratch*
  - netFound

NetLLM Overview

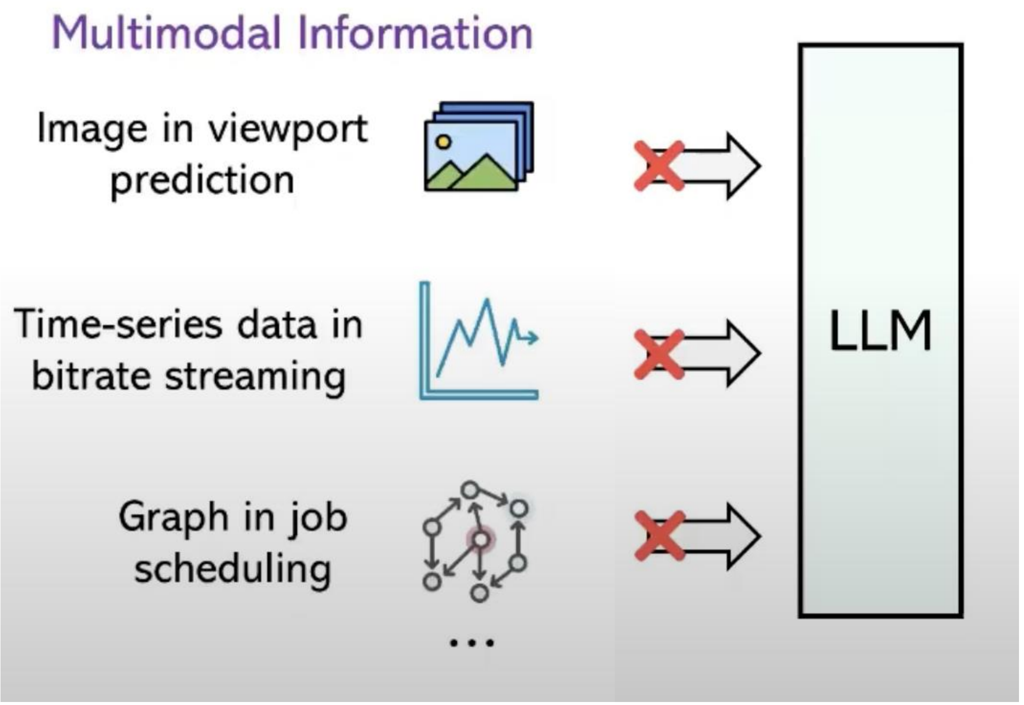


Three major innovations

- Multimodal Encoder
- Networking Head
- Data-driven low-rank networking adaptation

Challenge 1: Network Data is Multimodal

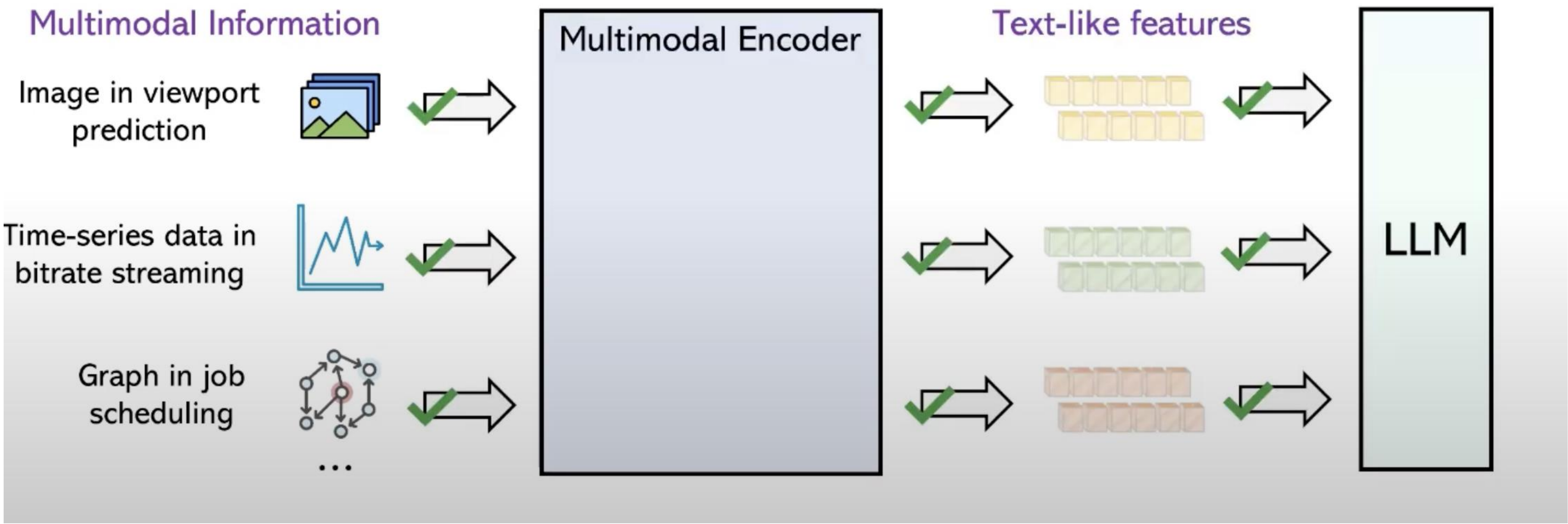
- How to enable the LLM to understand networking information?



Challenge 1: Network Data is Multimodal

- How to enable the LLM to understand networking information?

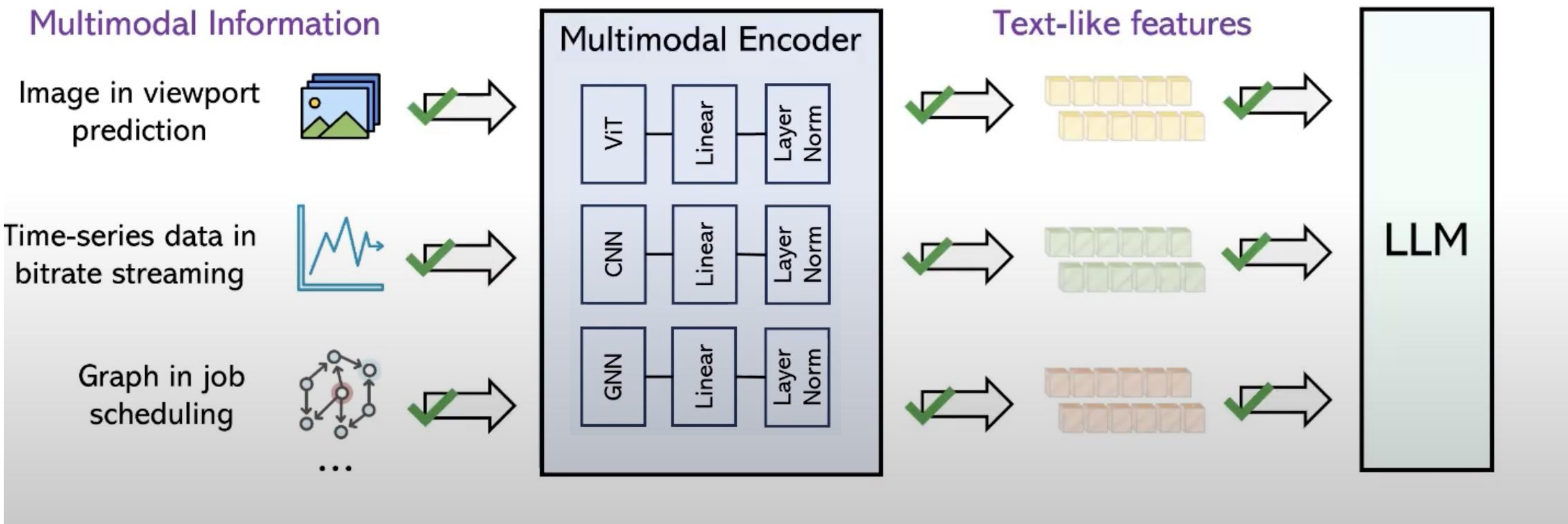
Solution: Project data into the same feature space as texts



Challenge 1: Network Data is Multimodal

- How to enable the LLM to understand networking information?

Solution: Project data into the same feature space as texts

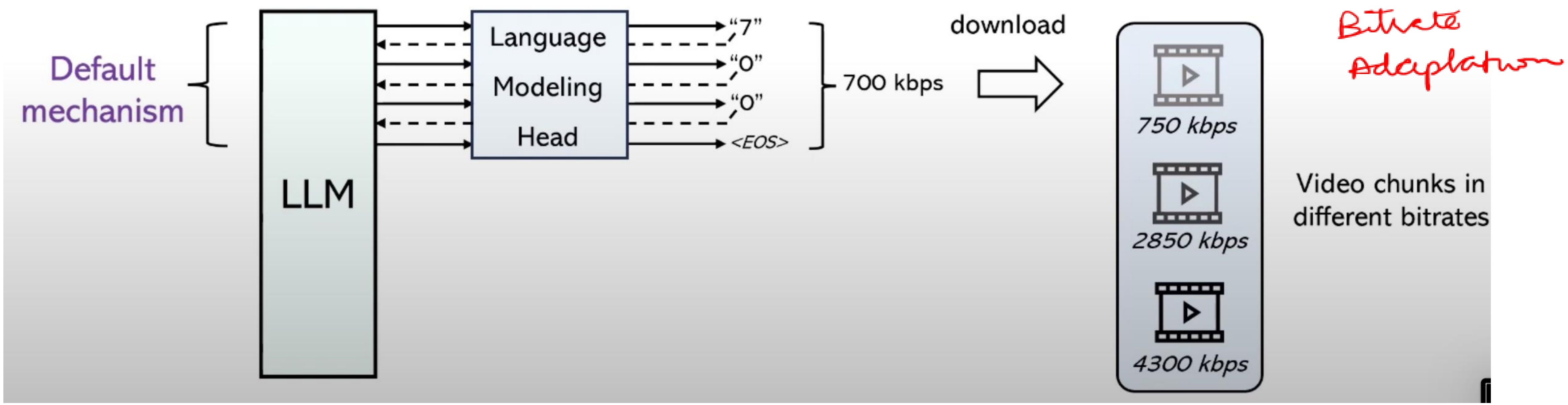


Challenge 2: Generate Output for Network Tasks

Default: Token-based generation with a language modeling output head

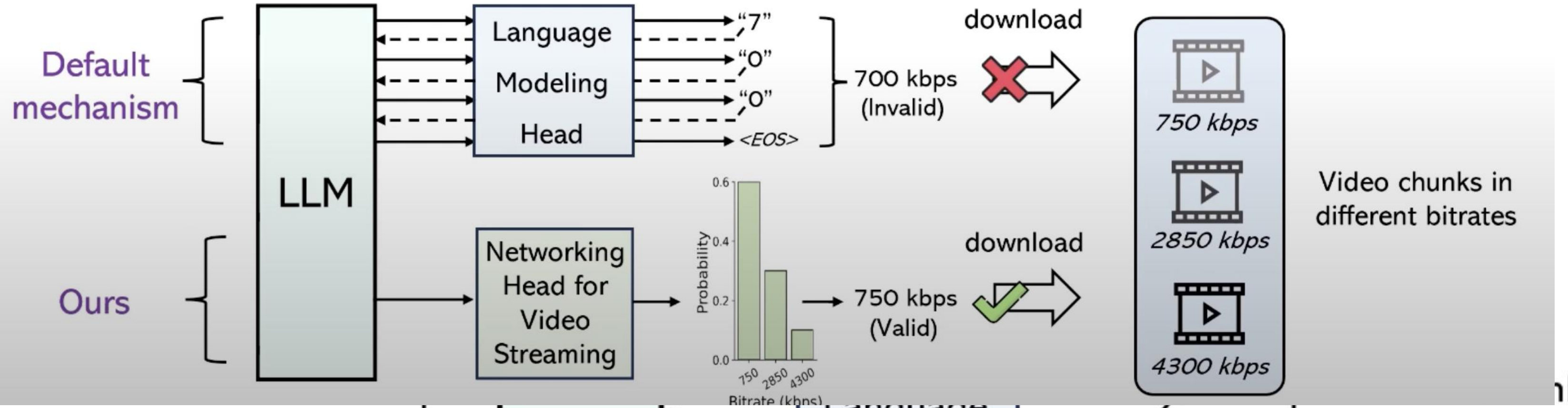
Prediction head

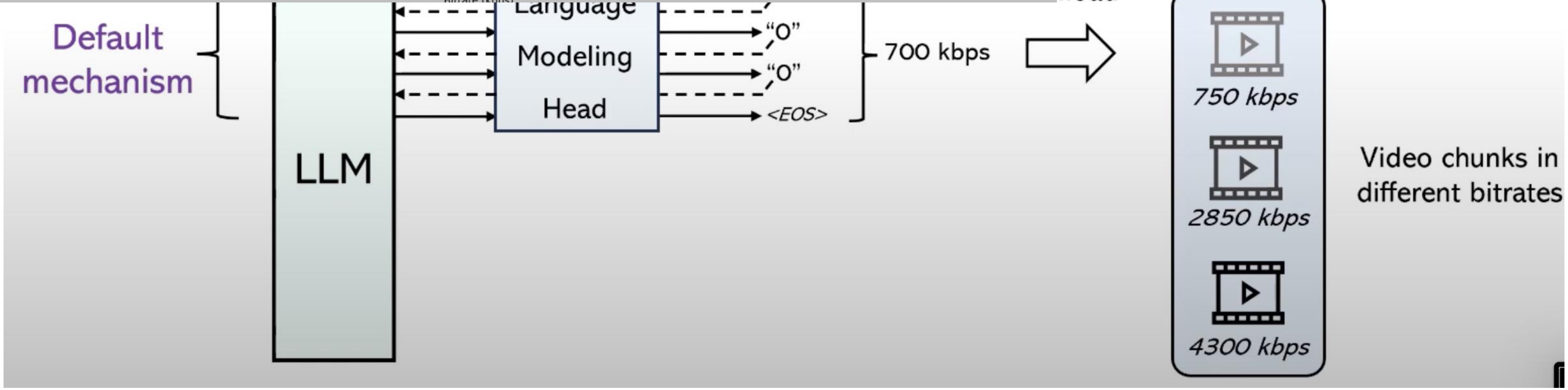
1. High latency, 2. Invalid answers



Challenge 2: Generate Output for Network Tasks

Default: Token-based generation with a language modeling output



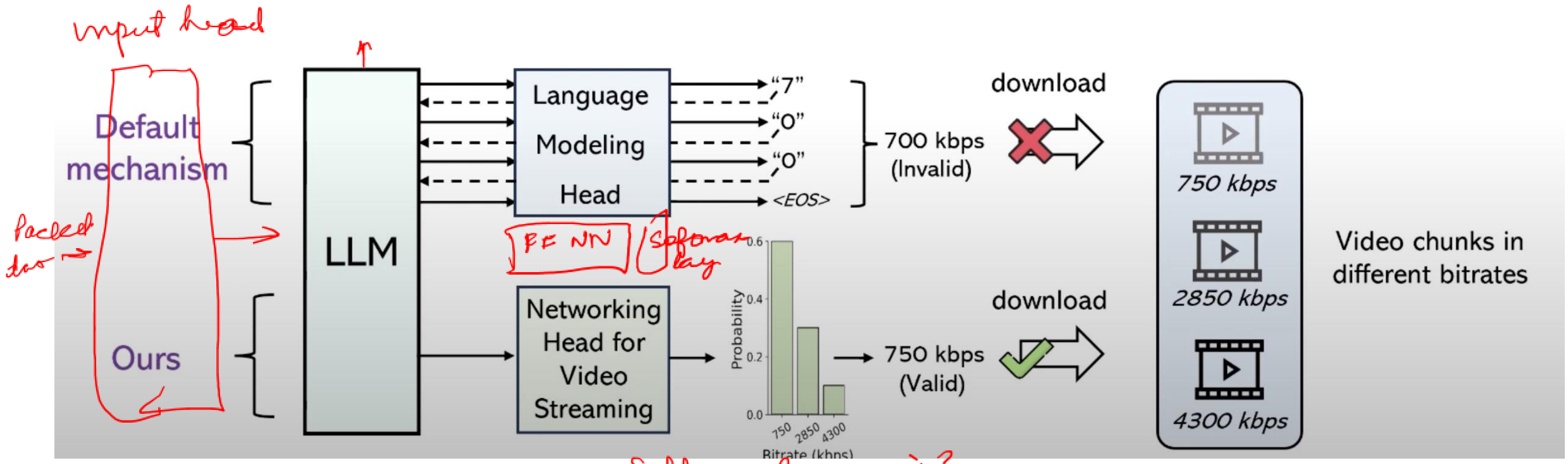


Application classification ~ N Appl

Challenge 2: Generate Output for Network Tasks

Default: Token-based generation with a language modeling output head

Solution: Networking head to generate task-specific answers directly



Softmax layer > 3

$O(dk) \rightarrow O(dr) + O(rk) = O(r(d+k))$ $\quad r << d$

$O(d^2)$

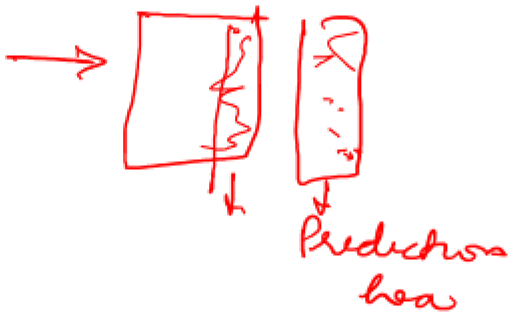
Challenge 3: How to fine-tune the LLMs to learn network knowledge effectively?

How to train RL methods?

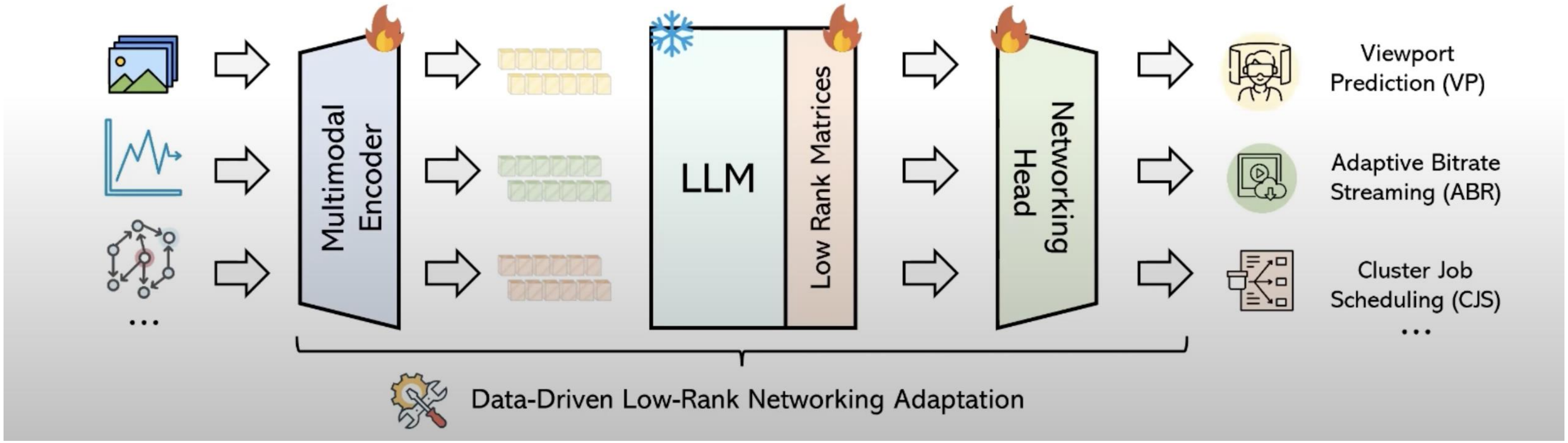
- The cost of fine tuning the LLMs are expensive due to large parameter size
  - E.g., Llama: 7B – 70.6 B depending on the version, model

$\phi' = \phi + \delta$ $\quad AB$ $\quad = W + W' \quad d \times k$

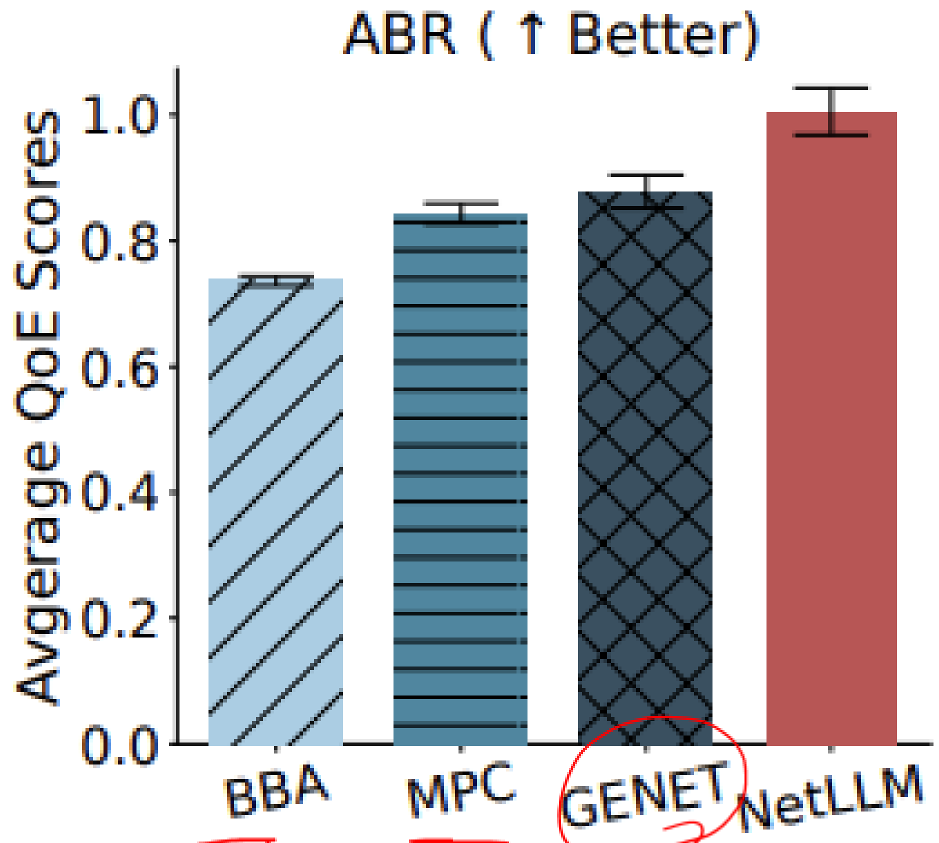
$+ AB \quad d \times r \quad r \times k$



- Solution: low-rank networking adaptation



Evaluation: Bitrate Adaptation



Buffer-based Adaptation

Model Predictive Control – DASH

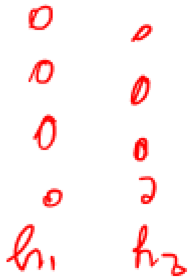
Special Topics: Machine Learning (ML) for Networking

COL867
Holi, 2025

Foundation Model

Tarun Mangla

$O(lk^2)$



$O(h_1 h_2) = O(h_1 r) + O(r h_2)$

Recap

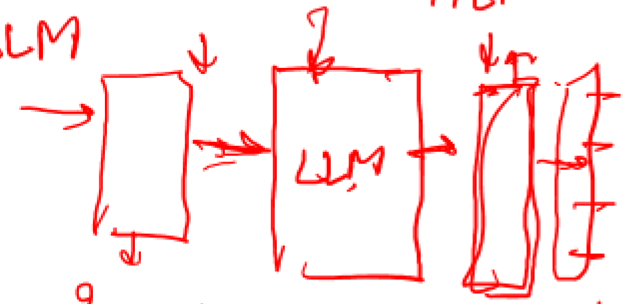
Task agnostic → generalizes
↓
saves effort + cost

- Foundation model for network data. **Why**?

- Two approaches
  - Adapt existing pre-trained large language models — NetLLM
  - Build a foundation model from scratch — NetFound

$W_0 + AB$

MLP



Input head

output head

- NetLLM
  - **Input:** Use existing deep neural network models
  - **Output:** Append a task-specific output head
  - **Training:** Improve fine tuning using low-rank adaptation

① Time series data → 1-D CNNs

② Image data → ViTs / 2-D CNN

③ Graph data → Job scheduling, TE
↓
GNNs

④ Scalar features

How to Build a Foundation Model for Networks?

Two different paradigms:

- Use a pre-trained large-language model
  - netLLM
- **Build a foundation model from scratch***
  - netFound

Building NetFound

(6) What should be the ML model? → Architecture

(1) How to tokenize data?

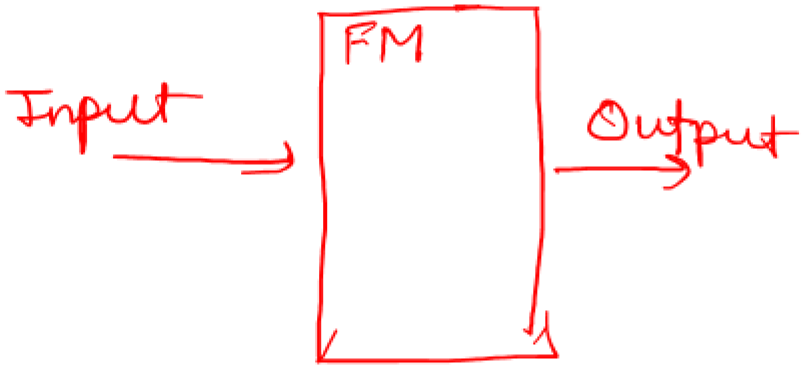
(2). What are the tasks → what is the data.

(3). Incorporate multimodality /

(4). latency of the system

(5) Traing

Architecture



(4) Traing

Input: How to tokenize the data?

- What is the network data?
- How to tokenize packet-level data?
- Challenges:
  - How to handle both content and statistics (multi-modality)
  - How to retain semantic integrity?
  - How to handle variable length input?

Given Seq of Packet [1] [2] [3]

Packet-level data → App classification
→ App performance prediction
→ Security / Anomaly detection

① Protocol semantics 2^{160}

② How to handle variable length.
[Trim the size
1) Truncate longer seq
2) Pad smaller seq

20 by | 20 by | n-bytes → x-bytes (2, 5, 10) ⇓ 2^{16}

| IP | TCP | Payload | + | IP | TCP | Payload |

Token1 Token2 Token3 4 5 6

Missing some information
: IAT
: # Pkts per flow
: Throughput

RTP headers

TLS header

Preserving Semantic Integrity

2 bytes

8-bit field

8 bits

- Use a protocol-aware tokenizer

5

7

- Consider 2-byte token
  - Pad shorter fields

6

Protocol	Fields				
IPv4	HeaderLen	ToS	TotalLen	Flags	TTL
TCP	Flags	WinSize	SeqNum	AckNum	UrgentPtr
UDP	Length	1	2	2	1
ICMP	Type	Code			
Payload	12 bytes				

18 tokens per packet

(1) SNI / DNS

32-bit (→ 2 tokens)

Content ≠ Metadata

Handling multiple modality

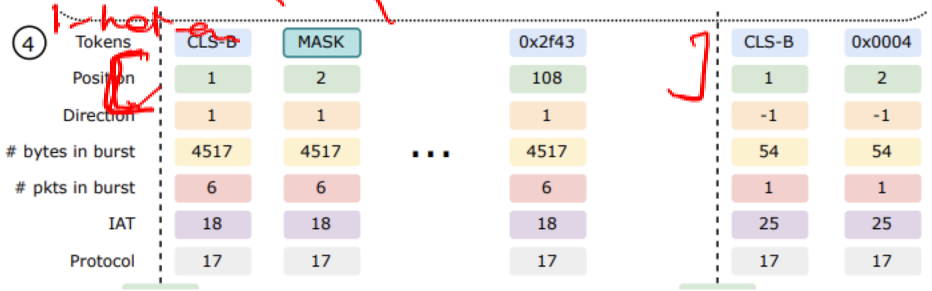
[0, ... Tmax]

- Example modalities:
  - Temporal details (IAT, pkt sizes)
  - Statistical aggregates (throughput)
  - Contextual information (Downstream/up)

Meta data

t1 t2 t3 - - - - -

1-hot-en



Handling Variable Length

- Problem statement → Seq length can change ↓ T

- **Solution**: Data-driven approach to determine the sequence length

Maximum

Avg → Trade-off

Median

Flow-level classification

Tokenize

FM

F : B_1, B_2, B_3, B_4.., B_N

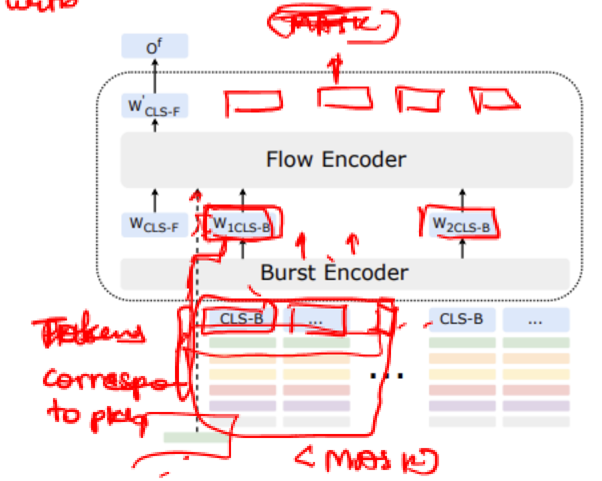
↓ flow encoder

Flow

Burst: group of pkts with IAT < δ

Model

- What should be the underlying model? Transformers

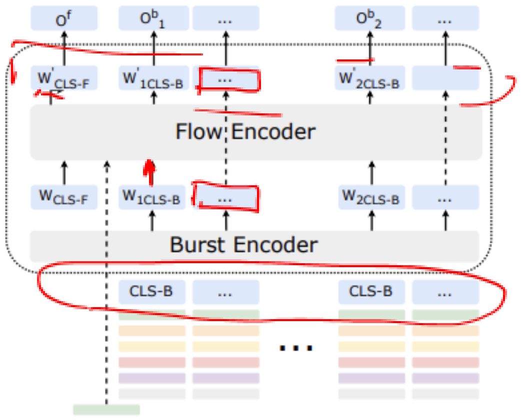


- Insight: Inherent hierarchy in the network data. Can we leverage it?

Packets → Burst → Flow

↓

Subnet ← Device ← Session



Workflow

- Data-preprocessing
  - Group packets into flows
  - Discard all flows with only 1-2 packets
  - Flow: 6 packets per burst, 12 bursts per flow
- Featurization
  - Extract relevant headers and metadata
- Tokenization
  - Maximum number of tokens per packet (why?): 18

Protocol	**Fields**				
IPv4	HeaderLen	ToS	TotalLen	Flags	TTL
TCP	Flags	WinSize	SeqNum	AckNum	UrgentPtr
UDP	Length				
ICMP	Type	Code			
Payload	12 bytes				

Token Embedding

- Packet field token embedding
- Positional embedding
- Metadata embedding

pre-training

$l \times p$ $\quad p \quad$ $l \times q$

$$\begin{bmatrix} 1 & 6 & \cdots \\ 0 & 0 & \\ 1 & 1 & \\ 1 & 0 & \end{bmatrix} \times W \rightarrow \begin{bmatrix} \; \end{bmatrix}$$

$+$

$l \times q$ size

$\rightarrow \begin{bmatrix} \; \end{bmatrix}$

$+$

$l \times 5 \times W_2 \rightarrow l \times q$

$(5 \times q)$

$l \times q \rightarrow$ Transformer

Training

- Self-supervised pre-training

    Mask 30%

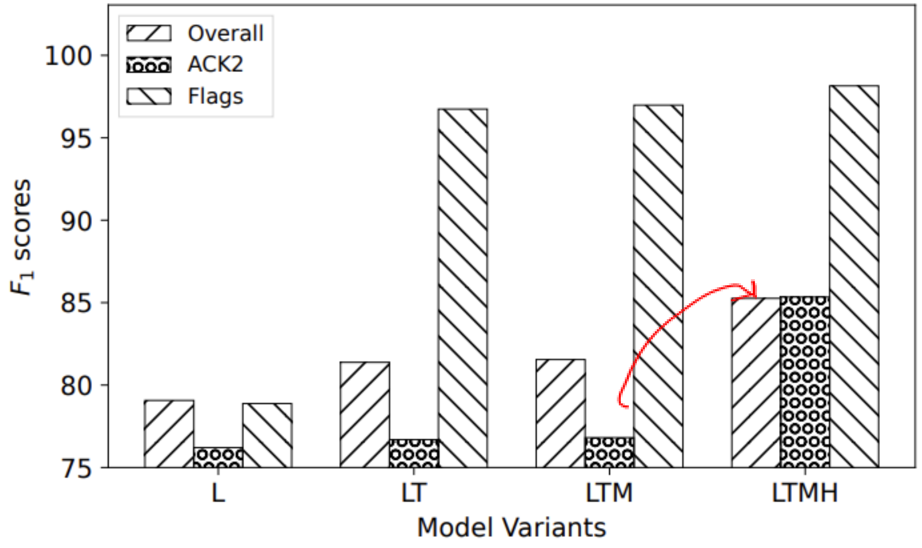
- Fine-tuning
    - Add two-layer MLP
    - Updates weights of pretrained model

Ablation study

Evaluation

Masked Token Prediction

- L: Flat transformer architecture
- LT: Protocol aware tokenizer
- LTM: LT + Metadata
- LTMH: LTM + Hierarchical



Fine-tuning Tasks

Task	Type	Dataset	Curtains (%)	nPrintML (%)	ET-BERT (%)	YaTC (%)	netFound (our) (%)
1	Traffic Classification	Campus dataset	54.53 ± 0.97 $p < 0.001$	87.22 ± 0.12 $p < 0.001$	72.26 ± 0.38 $p < 0.001$	76.54 ± 0.23 $p < 0.001$	**96.08** $\pm$ **0.04** –
2	Application Fingerprinting	Crossmarkets [59] ($Acc@10$)	20.64 ± 0.13 $p < 0.001$	64.83 ± 0.28 $p = 0.098$	35.62 ± 0.39 $p < 0.001$	58.13 ± 0.89 $p = 0.010$	**66.35** $\pm$ **0.99** –
3	Application Fingerprinting	ISCXVPN-2016 [60]	66.85 ± 2.21 $p = 0.003$	84.10 ± 0.41 $p < 0.001$	77.57 ± 1.20 $p < 0.001$	83.84 ± 0.24 $p < 0.001$	**91.02** $\pm$ **0.10** –
4	Intrusion Detection	CICIDS2017 [61]	99.75 ± 0.16 $p = 0.082$	99.93 ± 0.01 $p = 0.012$	99.94 ± 0.01 $p = 0.018$	99.92 ± 0.01 $p = 0.005$	99.99 ± 0.01 –
5	HTTP Bruteforce Detection	netUnicorn [5]	96.82 ± 0.22 $p = 0.006$	98.51 ± 0.02 $p < 0.001$	98.63 ± 0.02 $p < 0.001$	98.73 ± 0.10 $p = 0.030$	**99.01** $\pm$ **0.01** –

Discussion

ML for Networks

Module 1: Case studies of specific network learning tasks

Module 2: Task-agnostic automatic ML pipelines for networks

- Generalized data representation
- Generalized ML model(s)

Module 3: Beyond feature engineering and modeling

Model Architecture

- Want to use transformer

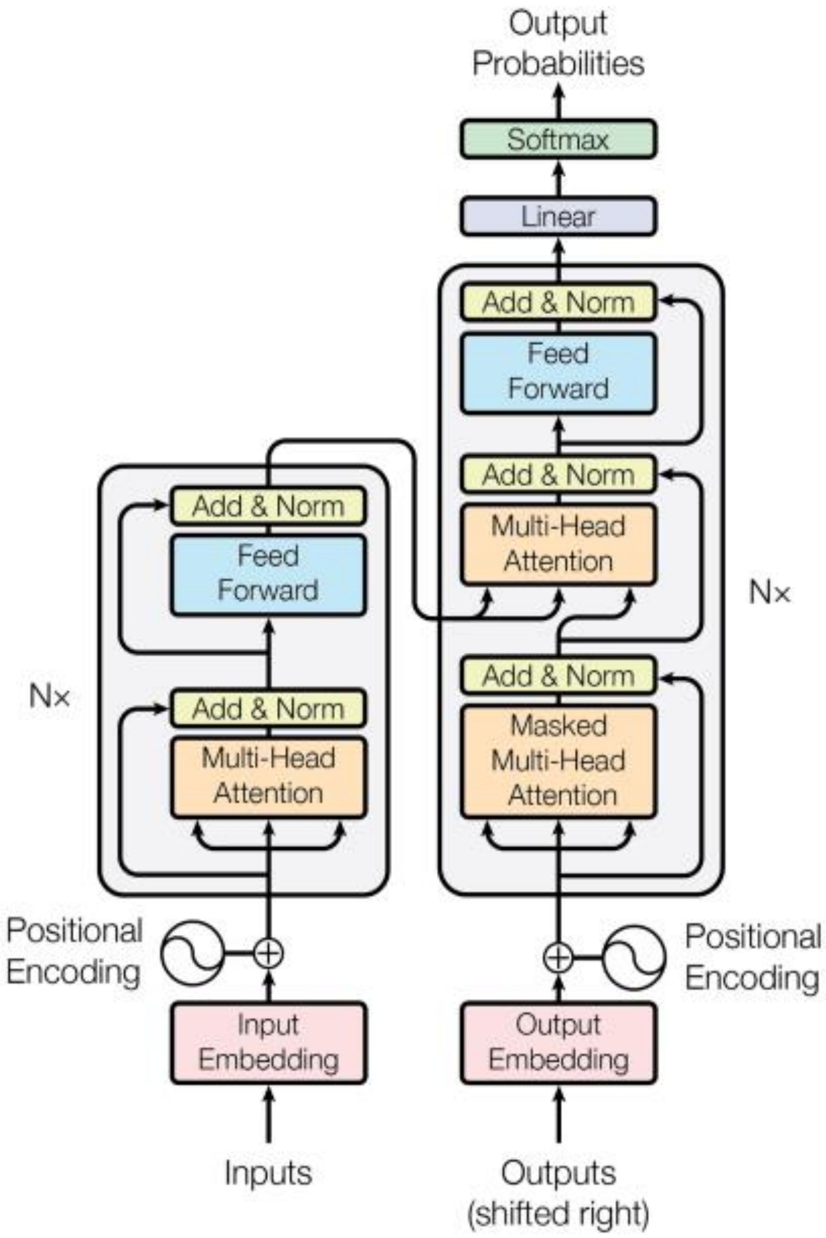
Why Foundation Models for Networking Data?

- Network learning approaches consist of classification, reinforcement learning, anomaly detection, generative
  - Foundation models have been successfully applied to these problems
- Abundant unlabeled sequential data
  - Campus networks
  - Data center networks
  - Transit/ISP networks
- Rich semantic content (like text)
  - Well-defined protocols

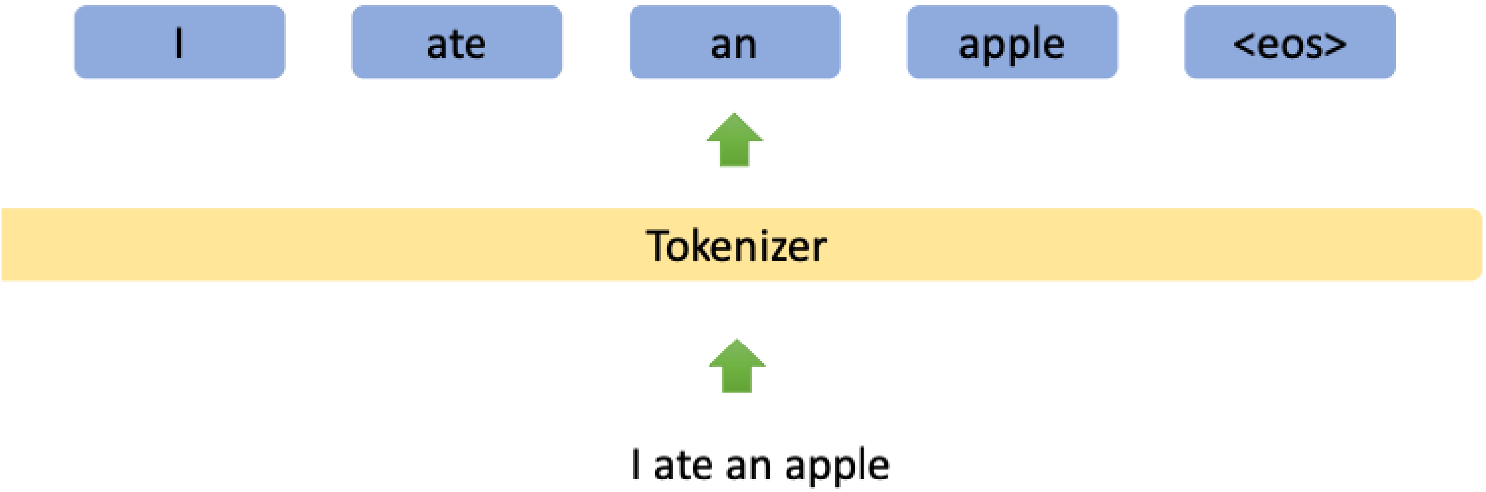
Transformer Architecture

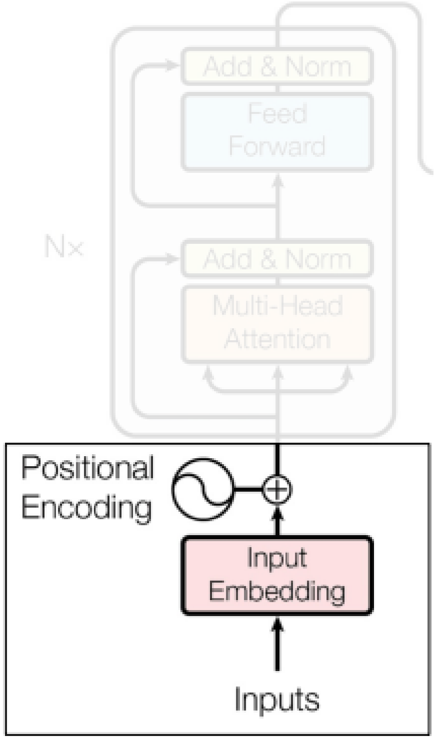
- Tokenizaton
- Input E
- Positior
- Residua
- Query
- Key
- Value
- Add & N
- Encoder
- Decoder

- Attention
- on
- Attention
- ention
- coder Attention
- abilities / Logits
- coder models
- ly models



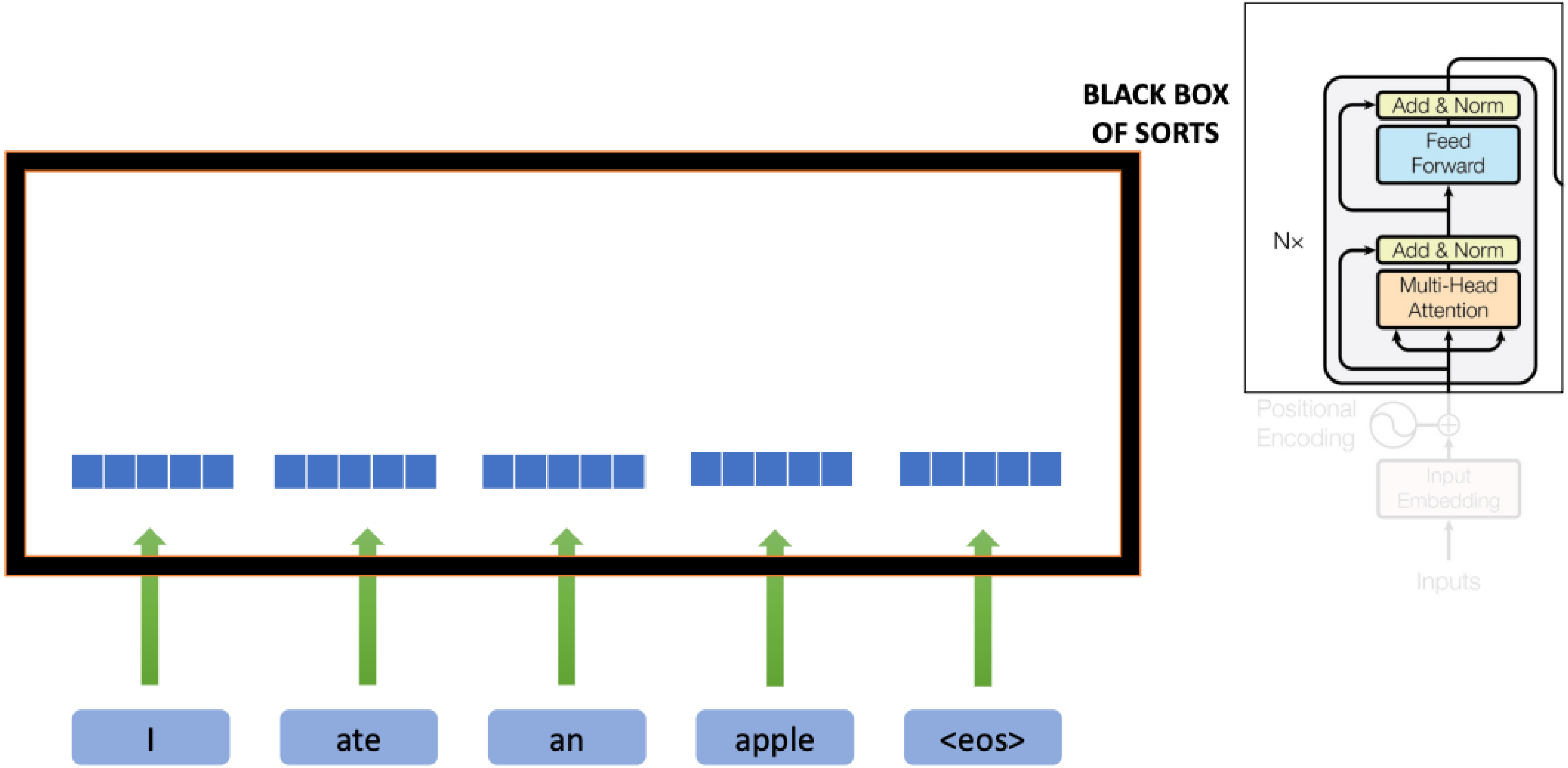
Processing Input





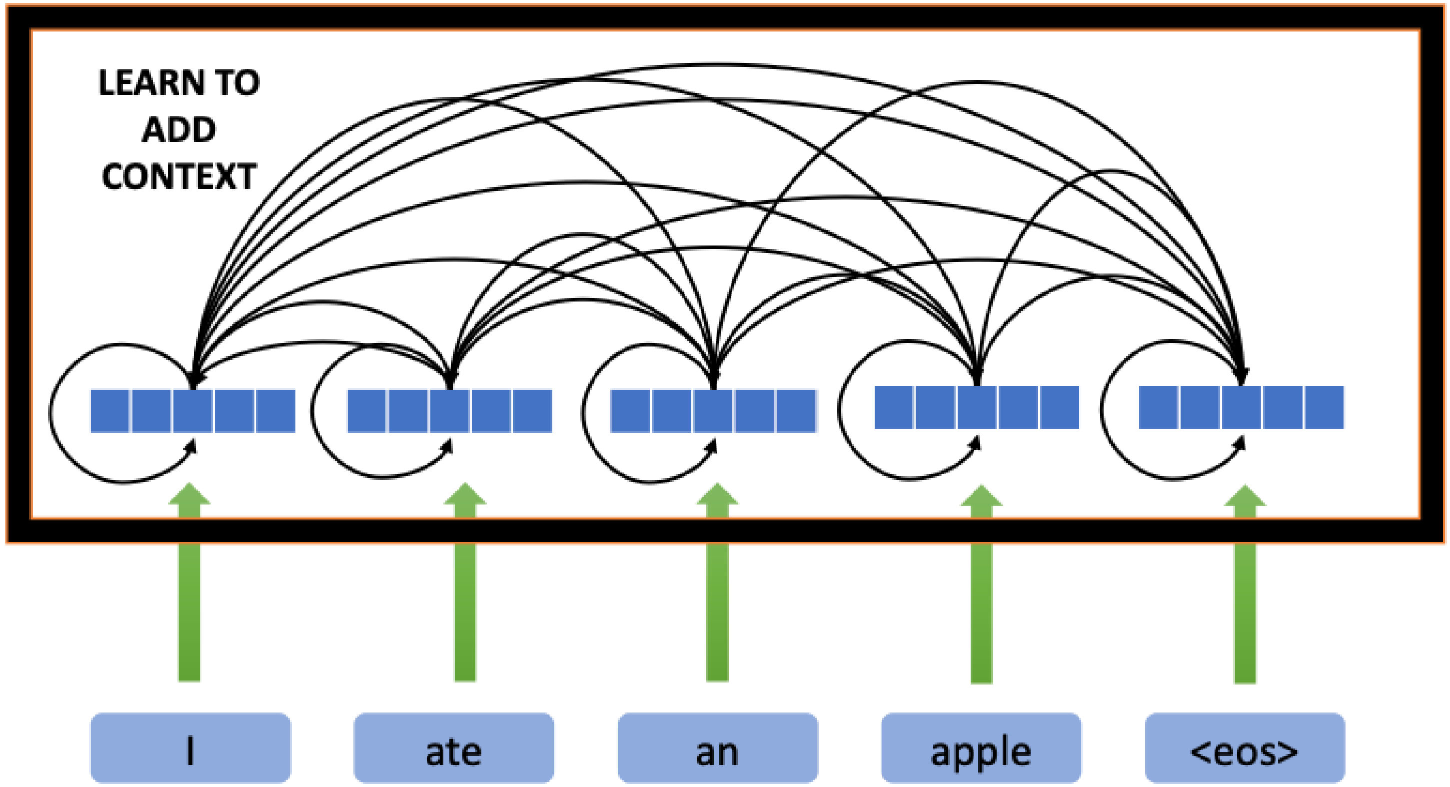
Generate Input Emebeddings

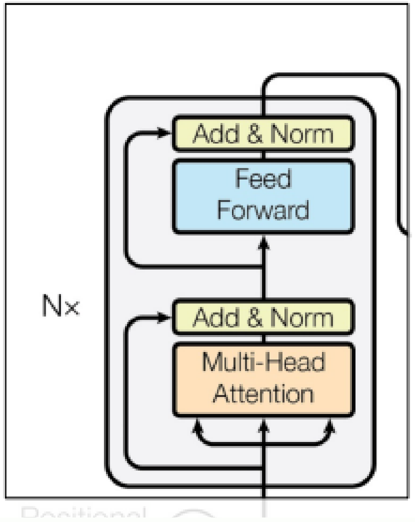
Capturing Context

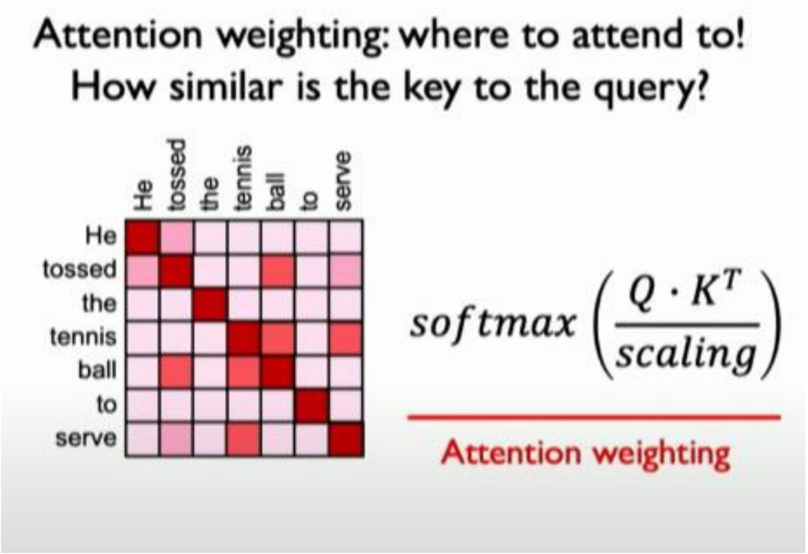


Capturing Context

BLACK BOX OF SORTS

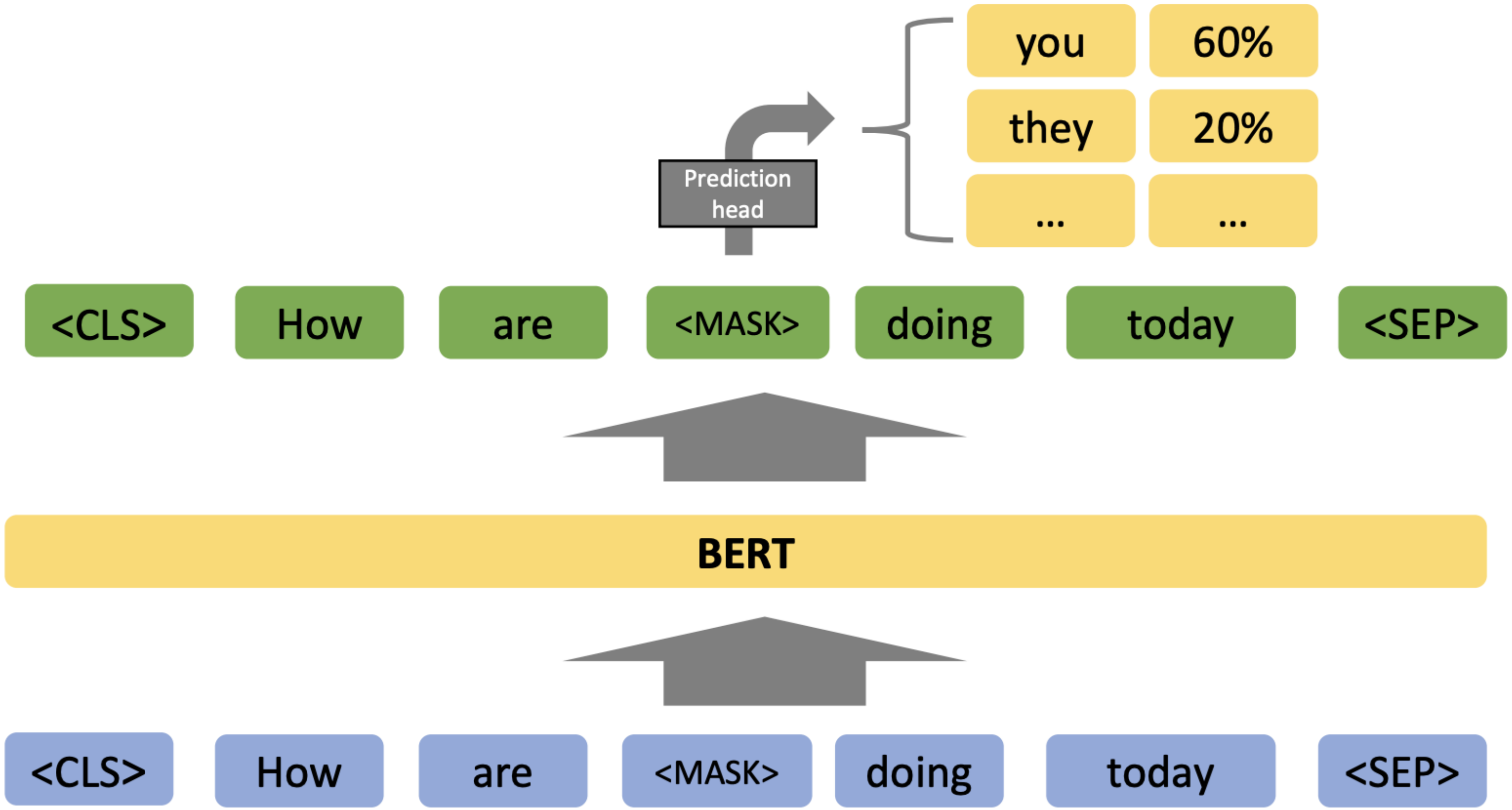


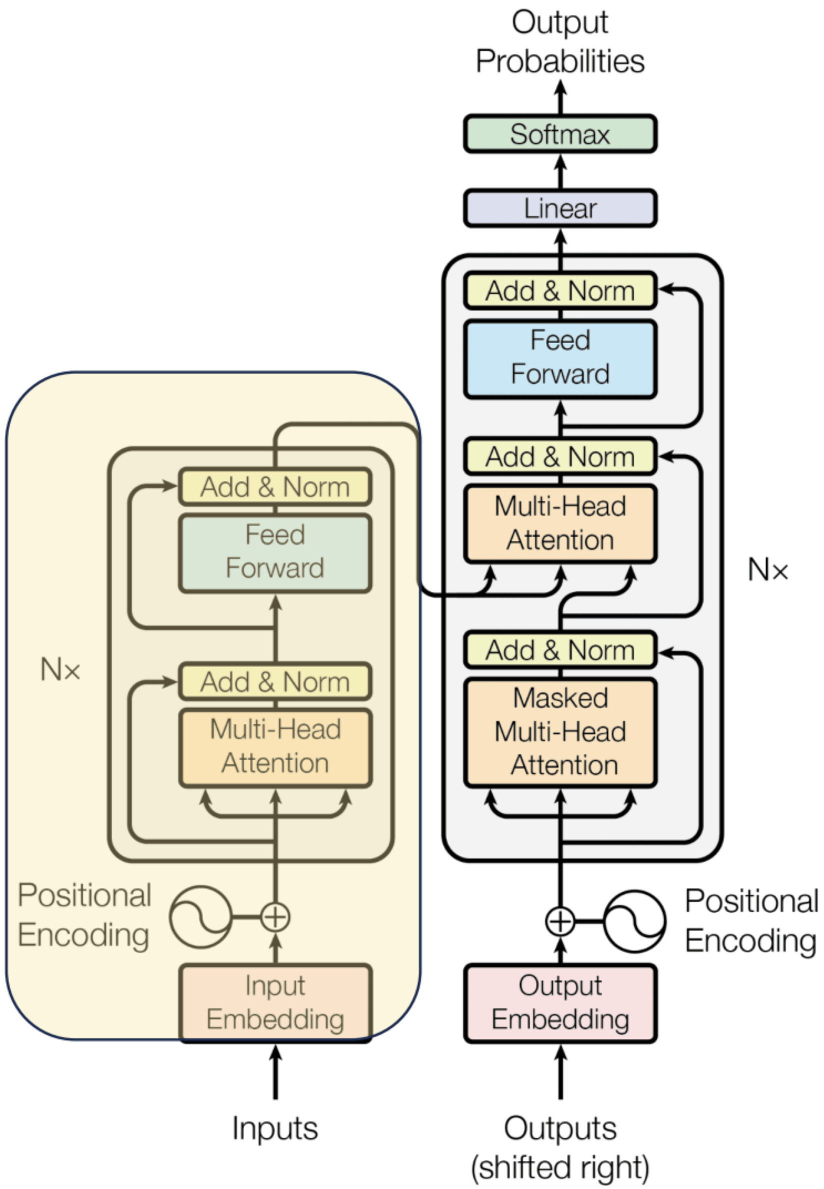




BERT: Bidirectional Encoder Representation

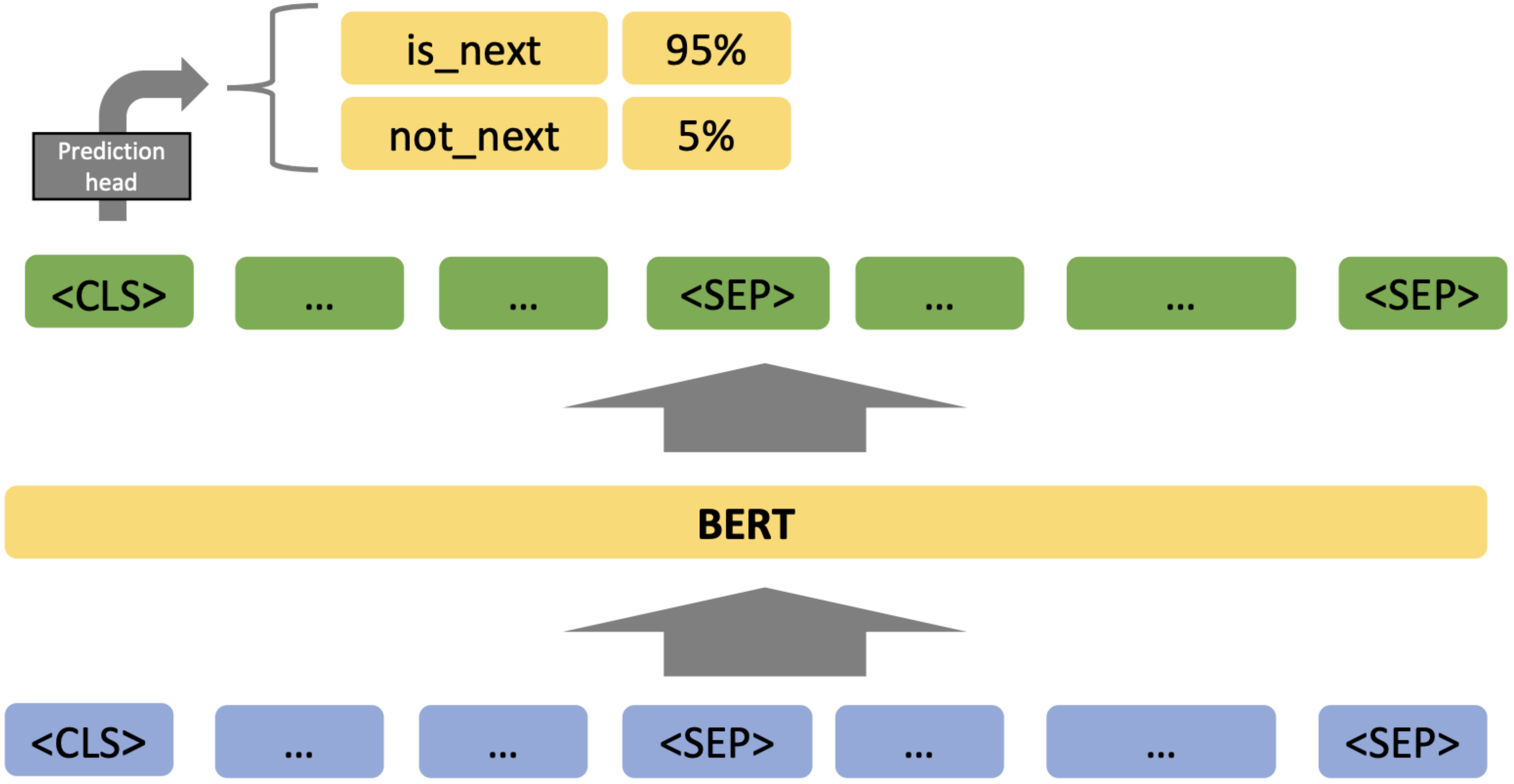
MLM (Masked Language Modeling)

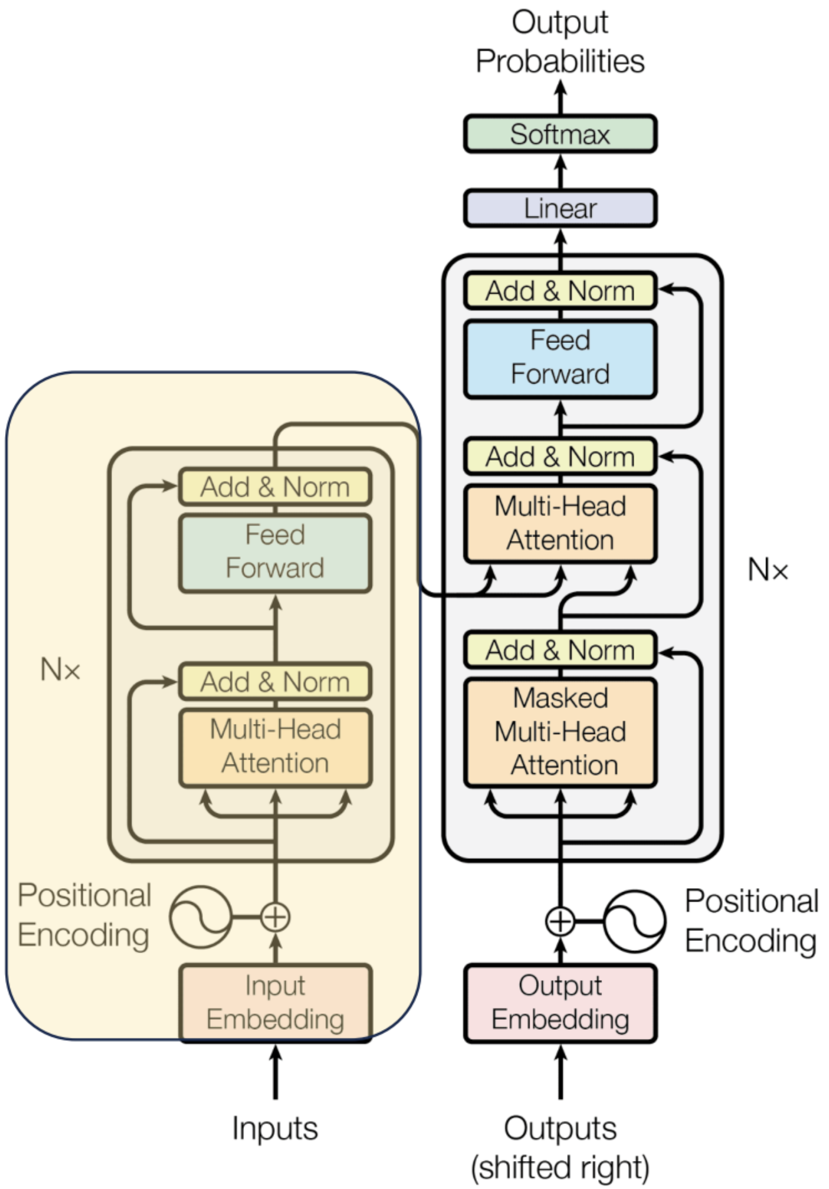




BERT: Bidirectional Encoder Representation

NSP (Next Sentence Prediction)





Why Foundation Models for Networking Data?

- Network learning approaches consist of classification, reinforcement learning, anomaly detection, generative
  - Foundation models have been successfully applied to these problems
- Abundant unlabeled sequential data
  - Campus networks
  - Data center networks
  - Transit/ISP networks
- Rich semantic content (like text)
  - Well-defined protocols

Challenges

- Tokenizer
  - Text: Tokens can be characters or words
  - What is a token for network data?
- Context
  - How do you define context?
  - What are the pre-training tasks?

- Post mid-term: Two papers on design of foundation model for networks
  - netFound
  - netLLM

Resources

- CMU Deep Learning lecture: https://deeplearning.cs.cmu.edu/S24/index.html
- MIT Deep Learning course: https://introtodeeplearning.com/
- Explanation with code: https://nlp.seas.harvard.edu/2018/04/03/attention.html
- Jay Alammar, The illustrated transformer: http://jalammar.github.io/illustrated-transformer/

Motivation

Network data has a unique context

- Multi-modal
  - Data from various contexts and perspectives
  - Cross-layer interactions
  - Network conditions and protocol interactions

- Hierarchical
  - Packets -> bursts -> flows -> session -> devices -> subnet etc.
  - Different learning problems make decisions at different granularity

Can we design a FM keeping in mind this unique context?

Design Decisions

- What kind of model architecture?
- How to represent network data as input to the model?
- How to (pre-)train the model?

Design Decisions

- **What kind of model architecture?**

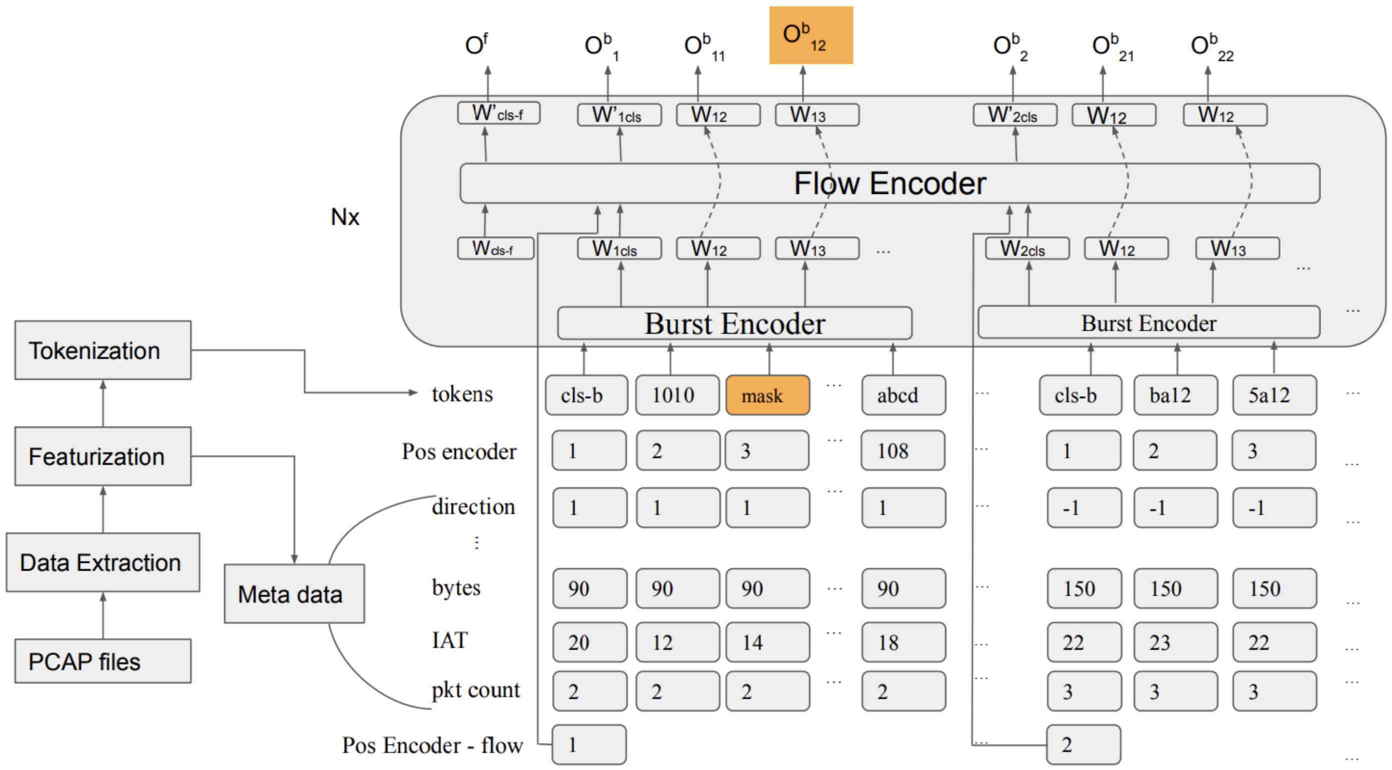
  Transformers

- How to represent network data as input to the model?

- How to (pre-)train the model?

Capturing Multi-modal Inputs

- Raw bytes
  - Similar to ET-BERT, use 2-byte tokens
  - Consider only 12 bytes of payload
- Meta-data
  - Embed other modalities as metadata
  - E.g., direction, time, number of packets in a burst etc.



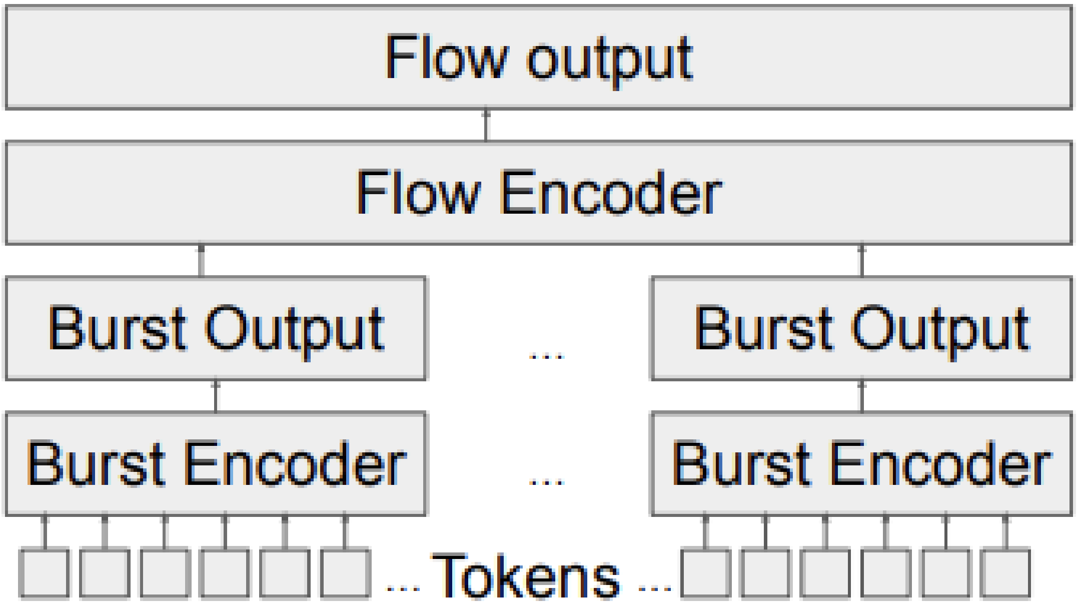
Capturing Hierarchical Structure

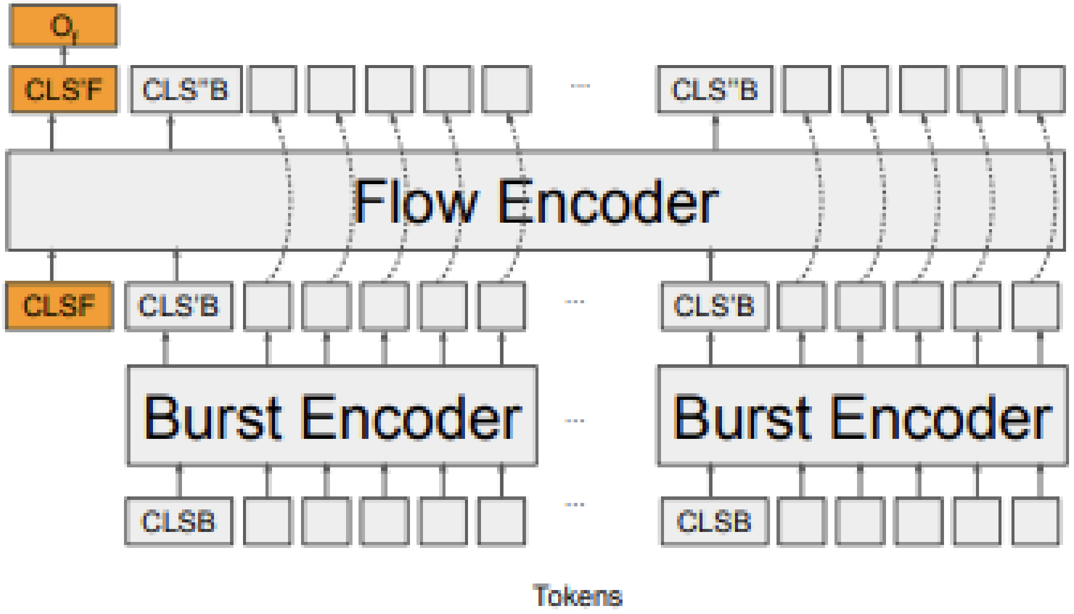
- Hierarchical data: packets -> bursts -> flows etc.

- How to create models to handle hierarchy in data?

- Option 1: Create different models for each level of hierarchy

Capturing Hierarchical Structure





Naive Approach

- Challenging to implement MLM task in this approach

Proposed Approach

- Use skip connections

Design Decisions

- **What kind of model architecture?**

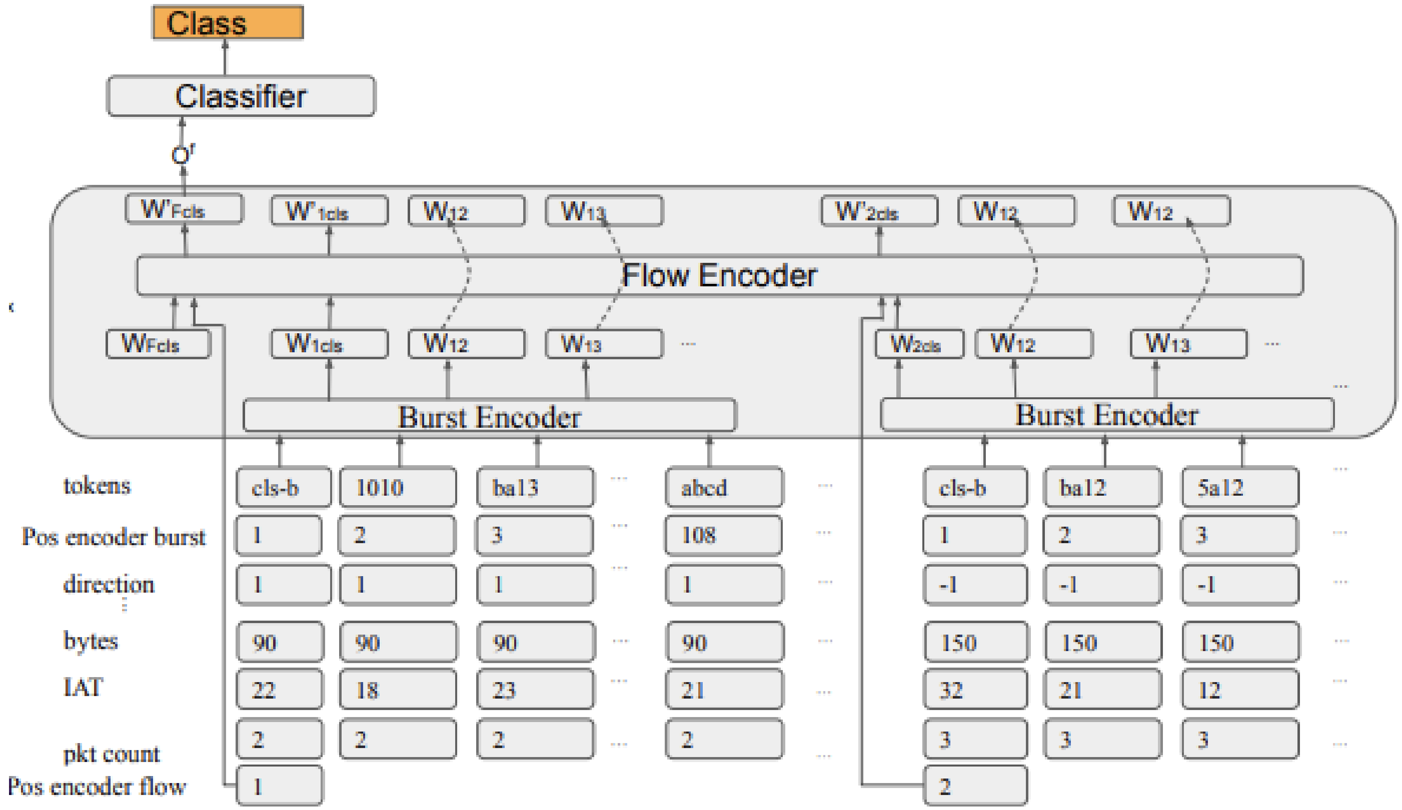
  Transformers

- How to represent network data as input to the model?

- **How to (pre-)train the model?**

  Masked Language Modeling

Putting It All Together



Workflow

- Tokenization
  - 13 packet fields with a 279-bit vector
  - 18 tokens per-packet
  - Special tokens: [PAD], [CLS-B], [CLS-F], [MASK]

- 6 packets per burst, 12 bursts per flow

- **Pretraining:** Only Mask-language Modeling task

Case Study: Mask Prediction

Pkt #	TCP Flag (Masked)	Burst-2	Length	Window	Seq #	Ack #
1	ACK	CLS	52	506	2726740280	2946828322
2	ACK+PUSH	CLS	735	501	2726740950	2946829950
3	ACK+PUSH	CLS	1339	824	2726752145	2946867610
4	ACK+PUSH	CLS	618	501	2726742967	2946832064
5	ACK	CLS	52	501	2726741633	2946830128
6	ACK	CLS	52	497	2726744816	2946848371

Downstream Tasks

Task	Curtains (%)			NprintML(%)			ET-BERT(%)			netFound(%)		
	PR	RE	F_1	PR	RE	F_1	PR	RE	F_1	PR	RE	F_1
Traffic classification	40.48±0.53 $p < 0.001$	43.42±0.62 $p < 0.001$	41.28±0.56 $p < 0.001$	83.86±0.16 $p < 0.001$	80.36±0.28 $p < 0.001$	81.43±0.24 $p < 0.001$	71.04±0.23 $p < 0.001$	69.24±0.17 $p < 0.001$	68.62±0.19 $p < 0.001$	88.79±0.27 -	87.97±0.10 -	88.33±0.14 -

Summary

- Task-agnostic network data representation is useful
- Three approaches:
  - Extract a set of (empirically) effective flow-level features
  - Packet-level bit vector representation
  - Foundation models
- Foundation models are promising. Few open questions
  - What is the best design of an FM for networking?
  - What are these models learning?
  - What are the resource implications of running these models?
  - How do we benchmark these models?
  - …