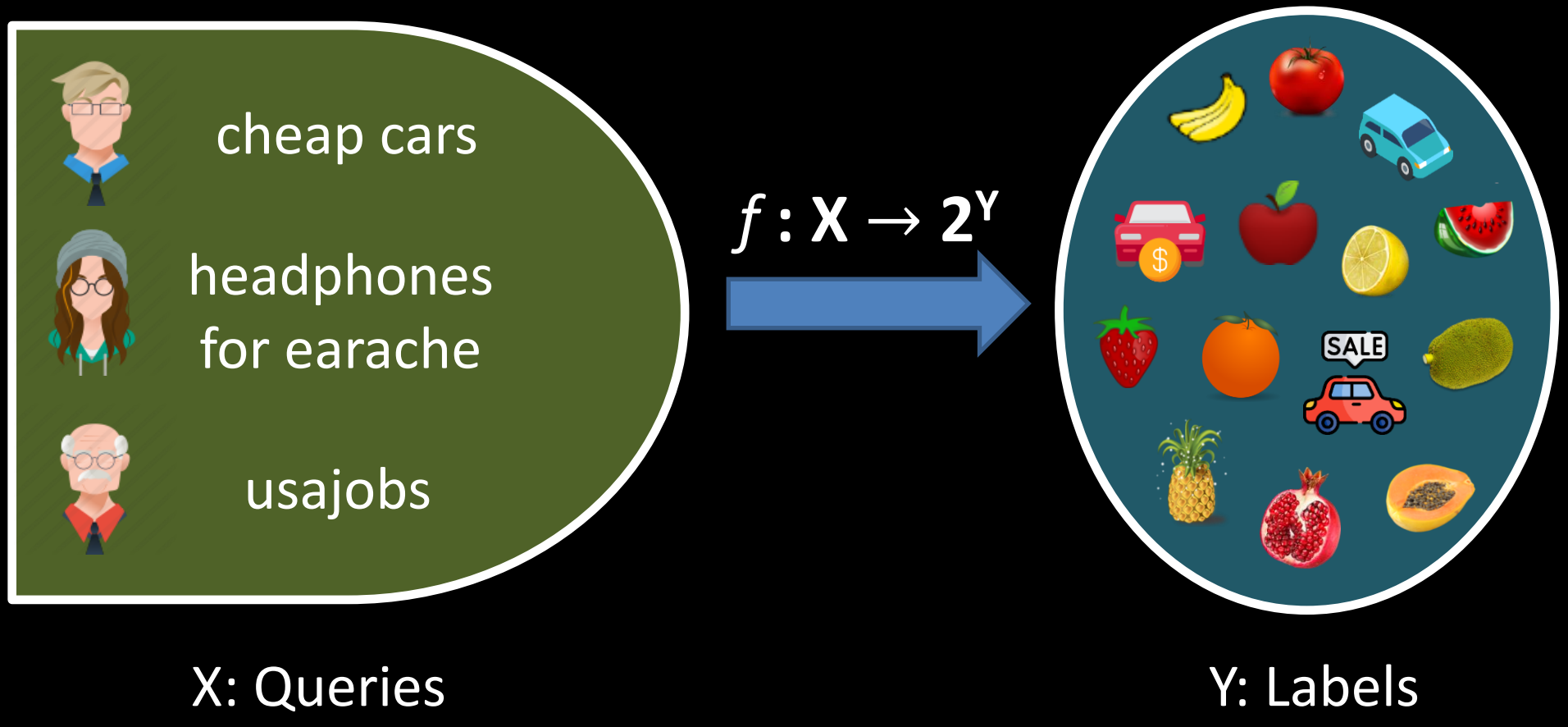
Renee´: END-TO-END TRAINING OF EXTREME CLASSIFICATION MODELS

Microsoft Research India: Vidit Jain, Jatin Prakash, Ramachandran Ramjee, Manik Varma

Microsoft: Deepak Saini, Jian Jiao

Problem setting: Extreme Classification (XC)

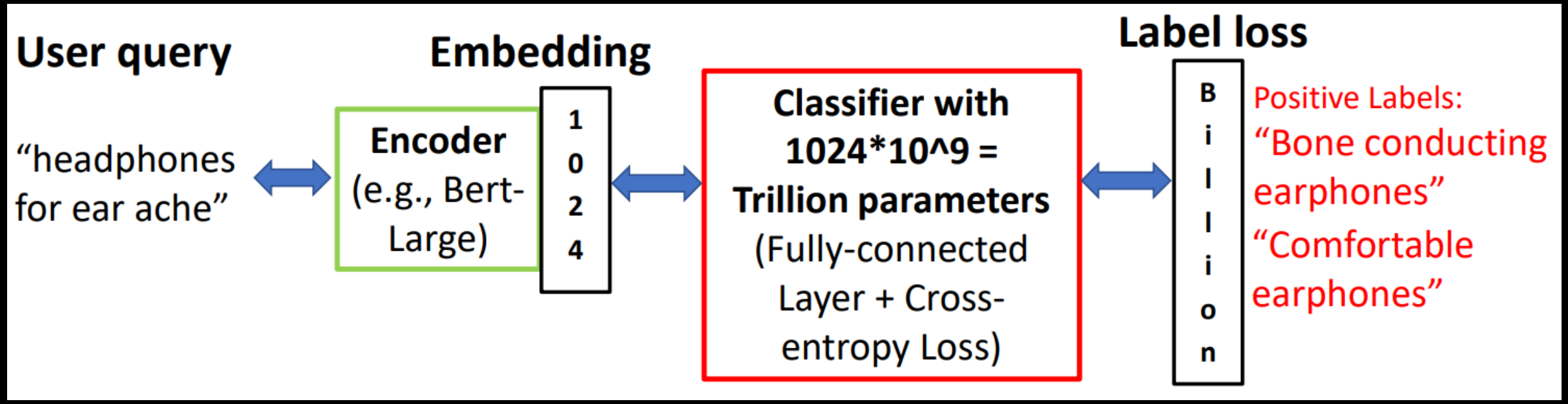
- For a given query, shortlist relevant labels from an extremely large set
- Model label space with free learnable vectors instead of encoder



End-to-End XC Training

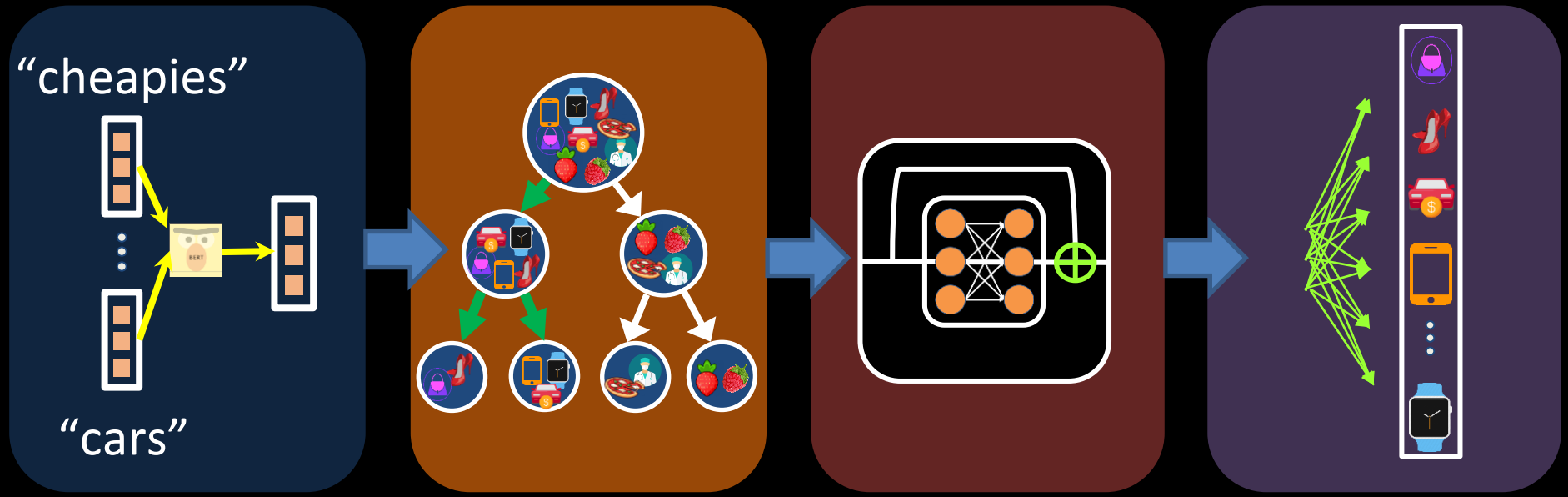
Architecture

- Encoder: Transformer-based model
- Classifier: Learnable free vectors for each label



Prior Work: Modular Training & Negative Mining

- Negative mining: Select small set of irrelevant labels per query
- Decomposed into modular sub-problems with surrogate loss functions for scalability



M1: Learning inter-mediate features

M2: Negative label sub-sampling

M3: Transfers to final features

M4: Joint classifier training

DeepXML [K. Dahiya, D. Saini, A. Mittal, A. Shaw, K. Dave, A. Soni, H. Jain, S. Agarwal & M. Varma, WSDM21]

NGAME [K. Dahiya, N. Gupta, D. Saini, A. Soni, Y. Wang, K. Dave, Gururaj K, P. Dey, A. Singh et al., WSDM23]

End-to-End XC Training

Widely acknowledged that end-to-end training is often better than modular training

Feasible at extreme scale?

Challenges with End-to-End XC Training: Overview

- BERT-Large encoder + 1B output labels
- Compute
  - Encoder: **1 Tflops/example**
  - Classifier: **6 Tflops/example**
- Memory
  - Encoder: **16 GB**
  - Classifier
    - Parameters: **16 TB**
    - Intermediate States: **16 TB**

Focus on reducing classifier memory

Renee: Optimize classifier memory

- Intermediate States: **16 TB → 2 TB**
  - Skip loss computation

- Classifier Parameters: **16 TB → 0.5 TB**
  - Bottleneck layer
  - Split the optimizer for encoder and classifier

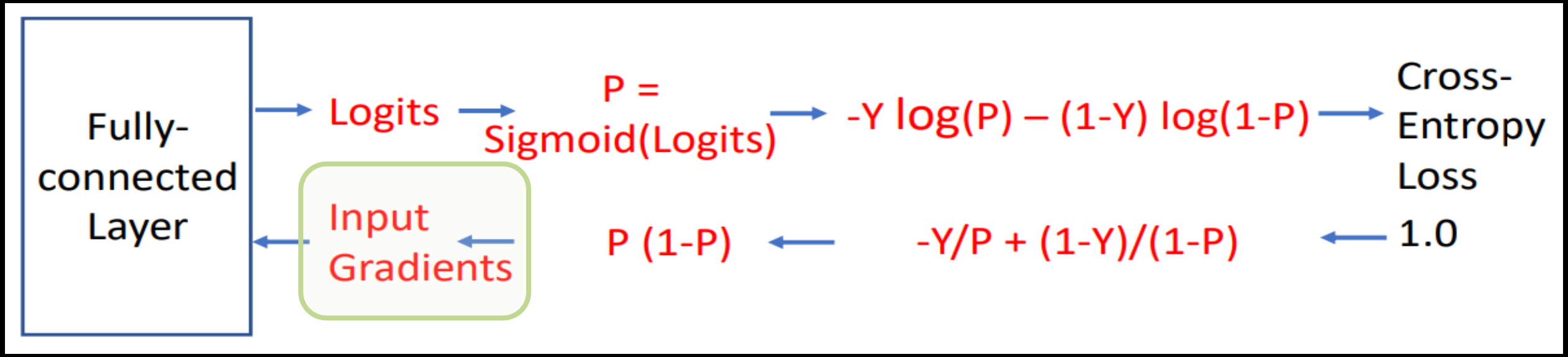
Renee: Optimize classifier memory

- Intermediate States: **16 TB** ➔ **2 TB**
  - Skip loss computation

- Classifier Parameters: **16 TB** ➔ **0.5 TB**
  - Bottleneck layer
  - Split the optimizer for encoder and classifier

Intermediate Memory: Skipping loss computation

- Several intermediate states like *logits*, *P*, *loss* and their derivatives
- Need only **input gradients** to the FC layer



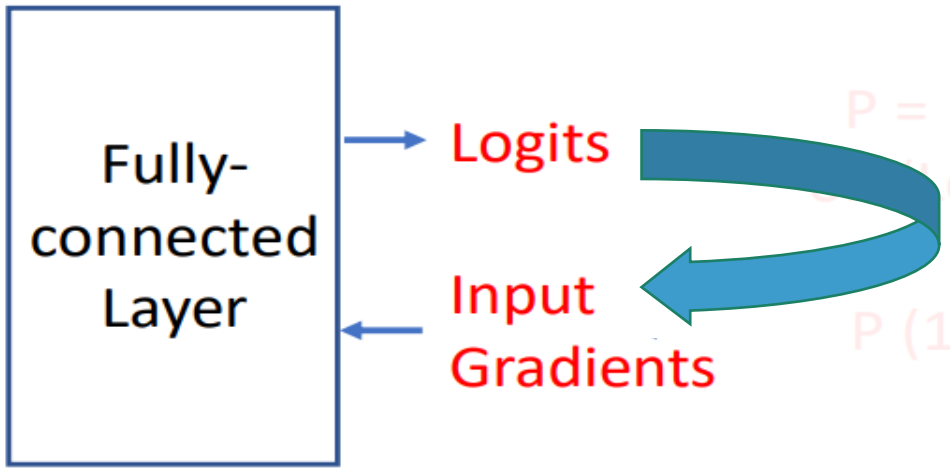
Intermediate Memory: Skipping loss computation

- Several intermediate states like *logits*, *P*, *loss* and their gradients
- Need only **input gradients** to the FC layer

Fuse all derivatives

$$
\begin{aligned}
grad &= P(1-P) * (-Y/P + (1-Y)/(1-P)) * 1 \\
&= -(1-P)Y + P(1-Y) \\
&= P - Y \\
&= 1/(1+e^{-x}) - Y
\end{aligned} \tag{3}
$$

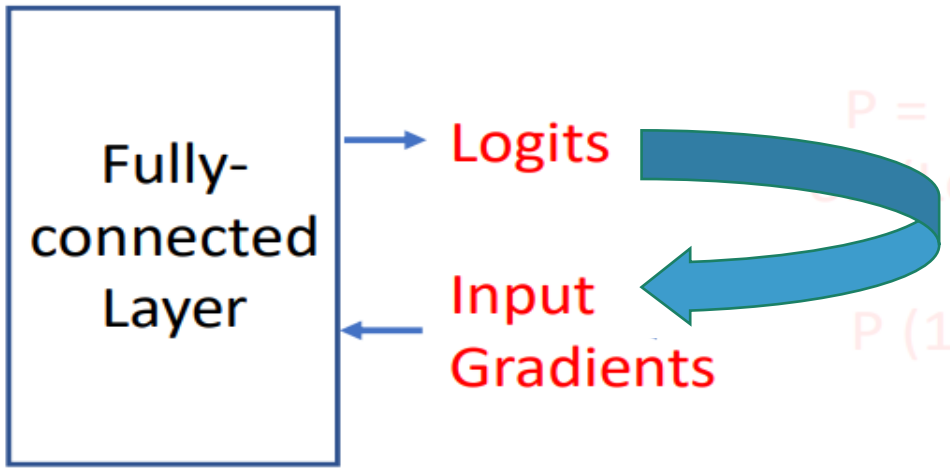
Intermediate Memory: Skipping loss computation



Shortcut to input gradients!!

$$
\begin{aligned}
grad &= P(1-P) * (-Y/P + (1-Y)/(1-P)) * 1 \\
&= -(1-P)Y + P(1-Y) \\
&= P - Y \qquad \text{Use } Y \text{ sparsely} \\
&= 1/(1+e^{-x}) - Y \qquad \text{Stable in fp16} \qquad (3)
\end{aligned}
$$

Intermediate Memory: Skipping loss computation



Memory: 16TB → 2TB

➢ **8x memory reduction**

$$
\begin{aligned}
grad &= P(1-P) * (-Y/P + (1-Y)/(1-P)) * 1 \\
&= -(1-P)Y + P(1-Y) \\
&= P - Y \\
&= 1/(1+e^{-x}) - Y \qquad (3)
\end{aligned}
$$

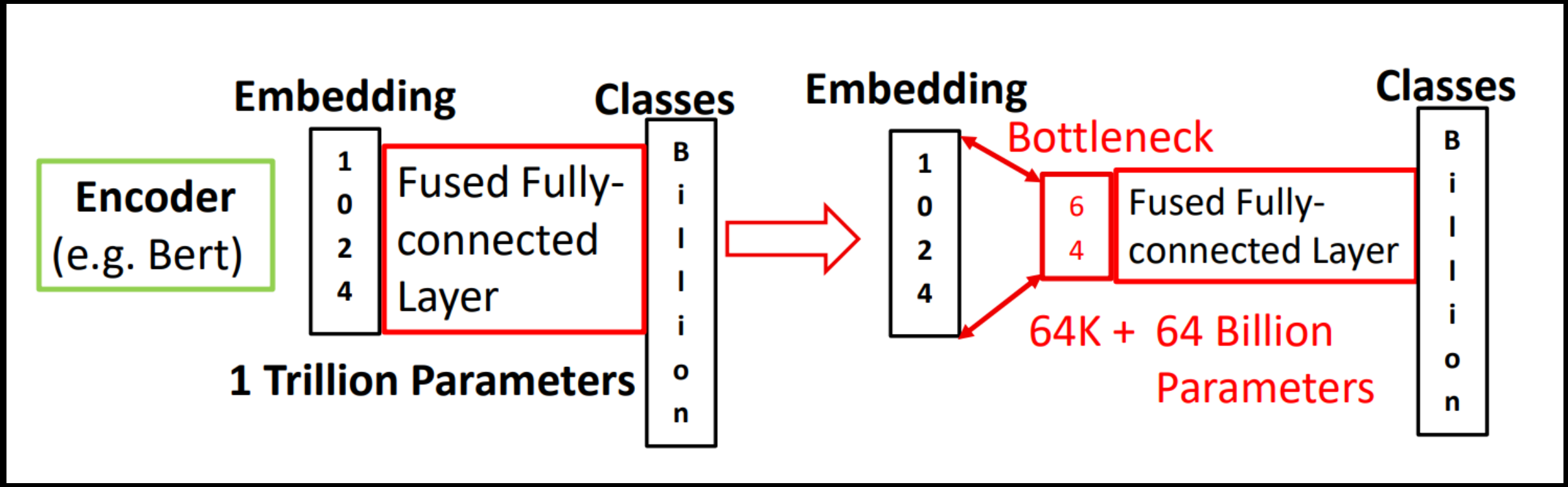
Renee: Optimize classifier memory

- Intermediate States: **16 TB → 2 TB**
  - Skip loss computation

- Classifier Parameters: **16 TB → 0.5 TB**
  - Bottleneck layer
  - Split the optimizer for encoder and classifier

Parameter Memory: Bottleneck Layer

- #(Fully-connected parameters) = *Embedding-size* × *Label-set Cardinality*
- Bottleneck layer: Linear layer mapping from 1024 to 64
- 16x reduction in parameter memory & compute; <1% accuracy hit



Renee: Optimize classifier memory

- Intermediate States: **16 TB → 2 TB**
  - Skip loss computation

- Classifier Parameters: **16 TB → 0.5 TB**
  - Bottleneck layer
  - Split the optimizer for encoder and classifier

Renee: Optimize classifier memory

- Intermediate States: **16 TB → 2 TB**
  - Skip loss computation

- Classifier Parameters: **16 TB → 0.5 TB**
  - Bottleneck layer
  - Split the optimizer for encoder and classifier

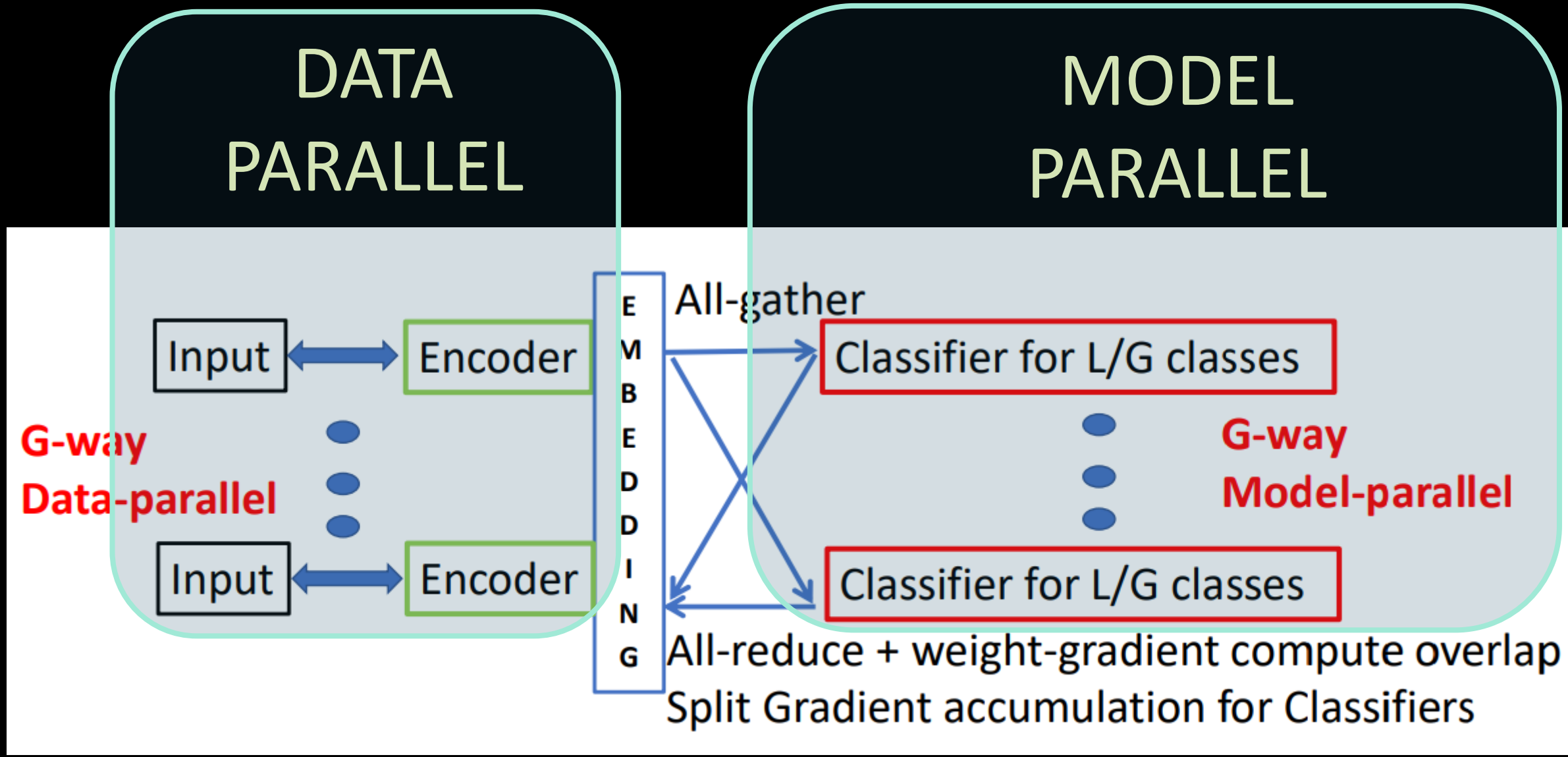
Hybrid Data-Model Parallel Training: Motivation

- Intermediate States: **16 TB → 2 TB**
  - Skip loss computation

- Classifier Parameters: **16 TB → 0.5 TB**
  - Bottleneck layer
  - Split the optimizer for encoder and classifier

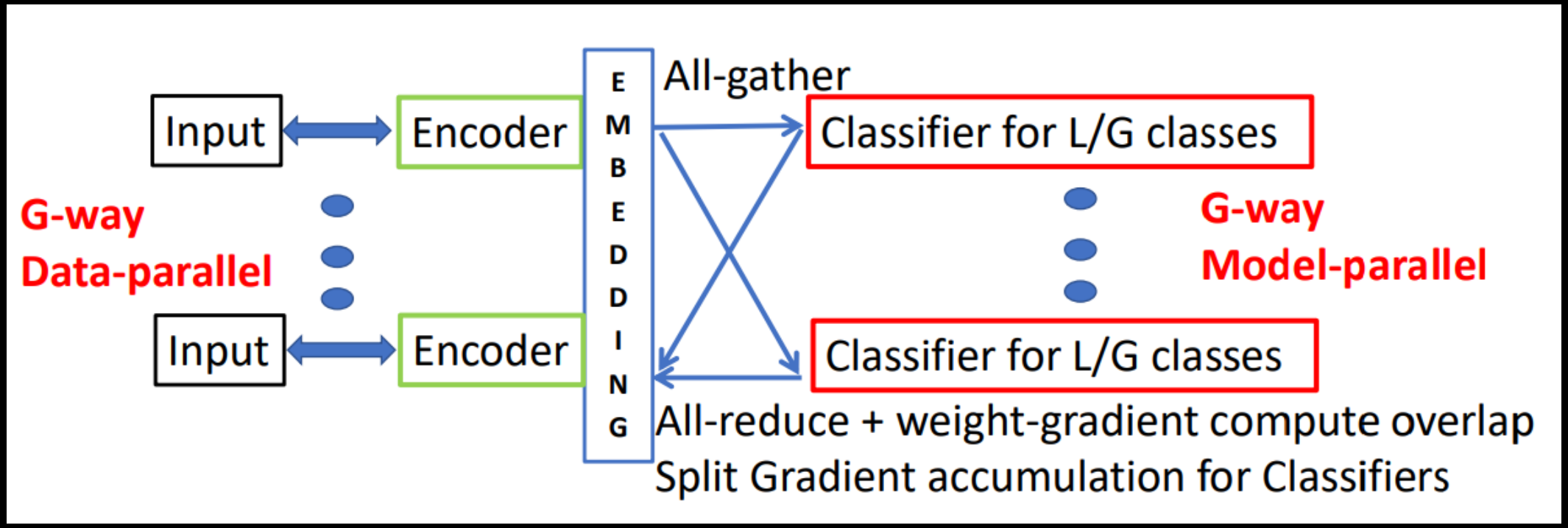
Single GPU still cannot fit 1B labels
Use multiple GPUs

Hybrid Data-Model Parallel Training: Architecture



Hybrid Data-Model Parallel Training: Efficient Scaling

- Backward pass has 2 large matmuls
  - Encoder gradient: Compute first and communicate to other GPUs
  - Classifier gradient: Overlap computation with encoder gradient communication
- Save communication overhead ➔ **Efficient scaling**



Proprietary Dataset: Statistics & Results

- Q2BP-120M: Match user queries to advertiser bid phrases
- **16% gains** in P@3 & P@5

#Labels	120,293,341
#Training Points	370,080,440
#Test Points	92,532,582

Methods	P@1	P@3	P@5
SiameseXML	83.46	36.90	23.64
NGAME	**87.82**	38.39	24.35
Renée (DiskANN)	83.30	55.01	41.06
Renée (Exact Search)	83.78	**55.30**	**41.27**

DiskANN [S.J. Subramanya, F. Devvrit, H.V. Simhadri, R. Krishnaswamy, R. Kadekodi , NeurIPS19]

Results: Accuracy

- Up to **5% gains** in P@1
- State-of-the-art across all five datasets
- Beats ensembles consisting of up to 9 models

Methods	AmazonTitles-670K			Amazon-670K			AmazonTitles-3M			Amazon-3M			Wiki-500K		
	P@1	P@3	P@5	P@1	P@3	P@5	P@1	P@3	P@5	P@1	P@3	P@5	P@1	P@3	P@5
AttentionXML	37.92	33.73	30.57	47.14	42.7	38.99	46	42.81	40.59	50.86	48.04	45.83	76.95	58.42	46.14
LightXML	41.7	37.3	34.2	47.3	42.2	38.5	–Not-Scalable–			–Not-Scalable–			76.19	57.22	44.12
XR-Transformer	41.07	36.66	33.55	49.11	43.8	40	48.72	45.74	43.35	52.6	49.4	46.9	78.1	57.6	45
***Renée* (ours)**	**45.2**	**40.24**	**36.61**	**54.23**	**48.22**	**43.83**	**51.81**	**48.84**	**46.54**	**54.84**	**52.08**	**49.77**	**79.47**	**60.37**	**46.84**
Ensemble Models (Shown for Completeness)															
X-Transformer-9	-	-	-	48.07	42.96	39.12	-	-	-	51.2	47.81	45.07	77.09	57.51	45.28
LightXML-3	43.1	38.7	35.5	49.1	44.17	40.25	–Not-Scalable–			–Not-Scalable–			77.78	58.85	45.57
XR-Transformer-3	41.94	37.44	34.19	50.11	44.56	40.64	50.5	47.41	45	54.2	50.81	48.26	79.4	59.02	46.25

Ablation: Renee Performance

- Compare Renee with standard Pytorch baseline **on Q2BP-120M**
- A **15x reduction** over baseline

Approach	Batch Size	Epoch Time (hrs)
Q2BP-120M, MiniLM-L3		
Baseline (grad accum)	512	204
Renee		13.5

Conclusion & Future Work

- End-to-end training is practical for XC tasks
- Renee achieves state-of-the-art results

- Exploring further accuracy and performance improvements
- Data Augmentation, loss functions, regularization, etc.

Thanks!

- Github link: *https://github.com/microsoft/renee*