

Scaling a Transformer-Powered Recommendation Model for Personalized Online Advertising

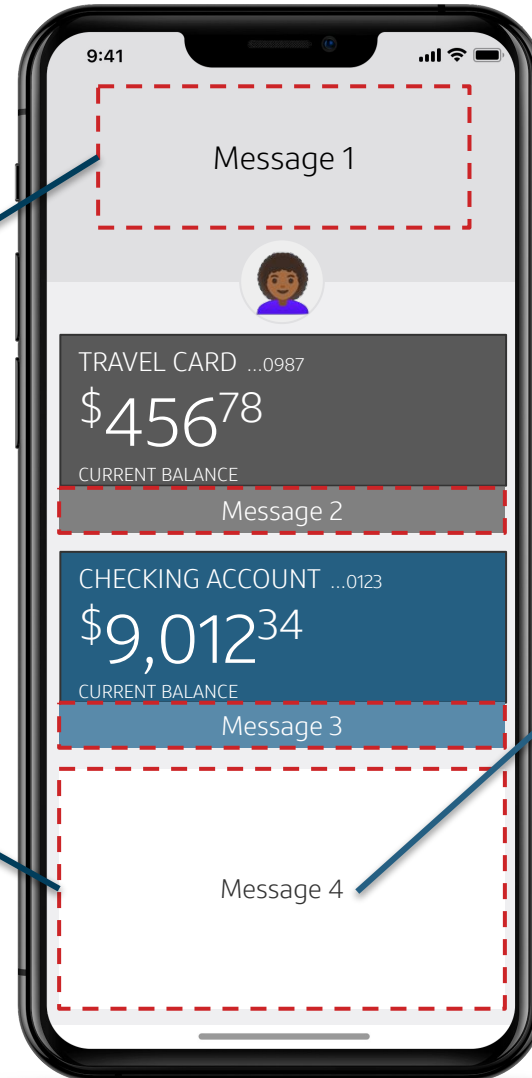
Kalanand Mishra, Snehita Varma, and Benjamin Wu
Capital One Data Science

March 20, 2024

Many mobile & web apps offer unique channels for serving highly personalized messages and product recommendations for authenticated customers

The type, location, and objective of each message can vary

Only one message can be shown in a set location at a given time



A recommender system can be used to determine the best message to serve each user

Car Shopping with **Auto Finance** – Search millions of cars with new ones added daily.



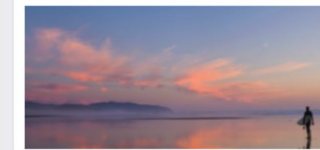
Explore Auto Finance

Savings Account: Earn one of the nation's best savings rates—plus no fees or minimums.



Explore Savings Account

Travel Card has arrived—our new travel rewards card designed to take you further.



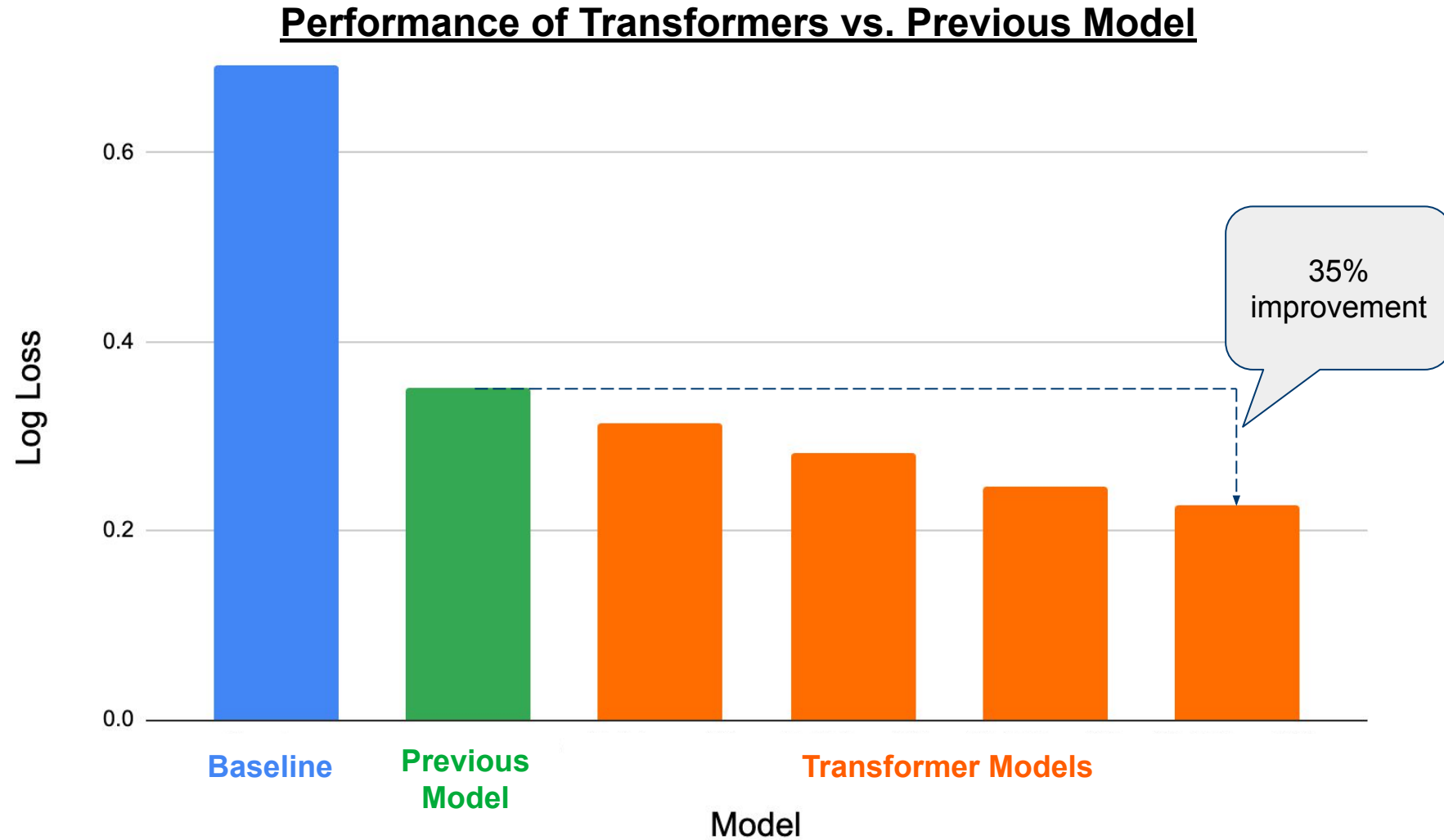
Explore Travel Card

Best Message?

Summary of Presentation

- A multi-step, multi-objective Recommendation Engine personalizes messages shown to a visitor
 - Previous tree-based ensemble models did not take into account sequenced data on user interactions
- Today: Transformer-based Recommender Engine
 - Applies self-attention to learn most relevant interactions from visitor's sequence
 - Utilizes NVIDIA Merlin's NVTabular and Transformers4Rec packages to achieve:
 - 35% gain in prediction quality in the experimentation data
 - Robust scaling solutions for preprocessing and modeling on user data size of hundreds of millions




Experiments with Transformer-powered recommendation systems show 35% improvement in model performance metric for all users



Agenda

- Description of the problem
- Transformer based Recommender System
 - Data preprocessing
 - Model training
 - Evaluation
- Scalability and performance
- Conclusion and future work

Relatively small catalogue sizes, low sparsity interactions, and multi-objective ranking require a recommender system that differs from those in typical e-commerce applications

Benchmark Dataset	# User	# Item	# Interaction	Sparsity	Interaction Type	Time Stamp	User Context	Item Context	Interaction Context
 Amazon* (Books)	8,026,324	2,330,066	22,507,155	99.9999%	Rating [0-5] ☆	Y	-	Y	-
 Steam*	2,567,538	32,135	7,793,069	99.99%	Buy	Y	-	Y	Y
 Netflix* (Prize data)	480,189	17,770	100,480,507	98.82%	Rating [1-5] ☆	Y	-	-	-
Example Financial Institution	~ 100 MM	~ 1000	~ 4 B	~ 70%*	Clicks / conversions / engagement	Y	Y	Y	Y

Relatively smaller catalog size, but with multiple objectives

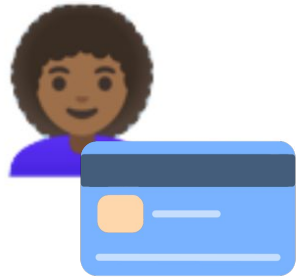
User-item interactions and data sparsity are fundamentally different

Events could be delayed by several days

Rich and accurate user context (KYC)

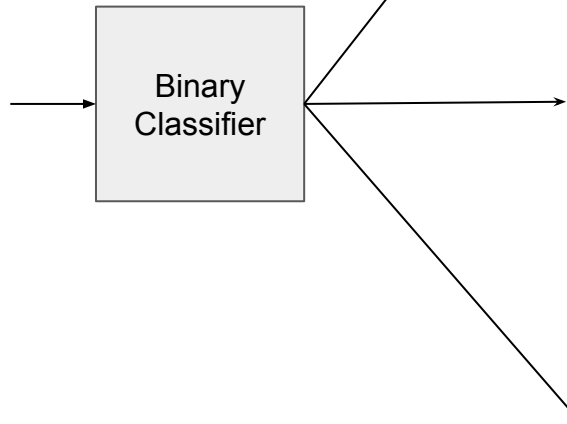
... bringing us from next item prediction into the classification domain

Previous tree-based ensemble models took into account a user's context, but did not leverage sequential user data



Date	Customer	Input Space		Response
		Message Objective	Owns a credit card?	Convert
2/27/2024	Jane	CD	1	0
3/2/2024	Jane	Auto Finance	1	0
3/3/2024	Jane	Savings	1	0
3/15/2024	Jane	Travel Card	1	1 (Savings Account)

Session-Level Data Structure



Travel Card has arrived—our new travel rewards card designed to take you further.

Explore Travel Card

Probability of conversion if Jane sees Message 1 = 0.03

Savings Account: Earn one of the nation's best savings rates—plus no fees or minimums.

Explore Savings Account

Probability of conversion if Jane sees Message 2 = 0.02

... etc.

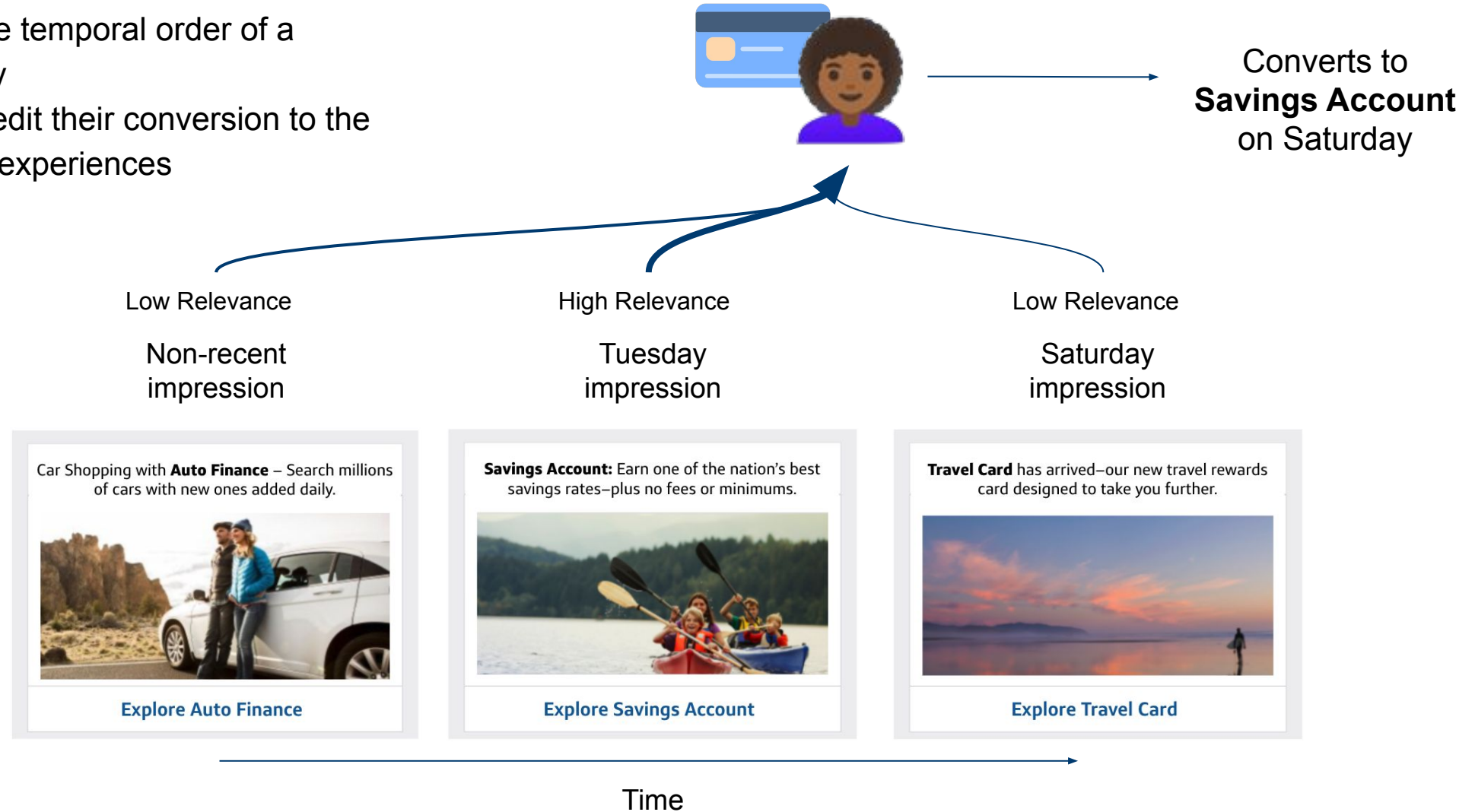
Ranked List of Messages

- Train model to predict conversion based on input configuration
- Score candidate messages based on trained parameters, pick highest scoring message to show next

Sequence modeling utilizes historical sequence data and serves the right message to the right individual more often, stimulating performance lift

Advantages:

- Keeps track of the temporal order of a user's visit history
- Learns how to credit their conversion to the appropriate prior experiences

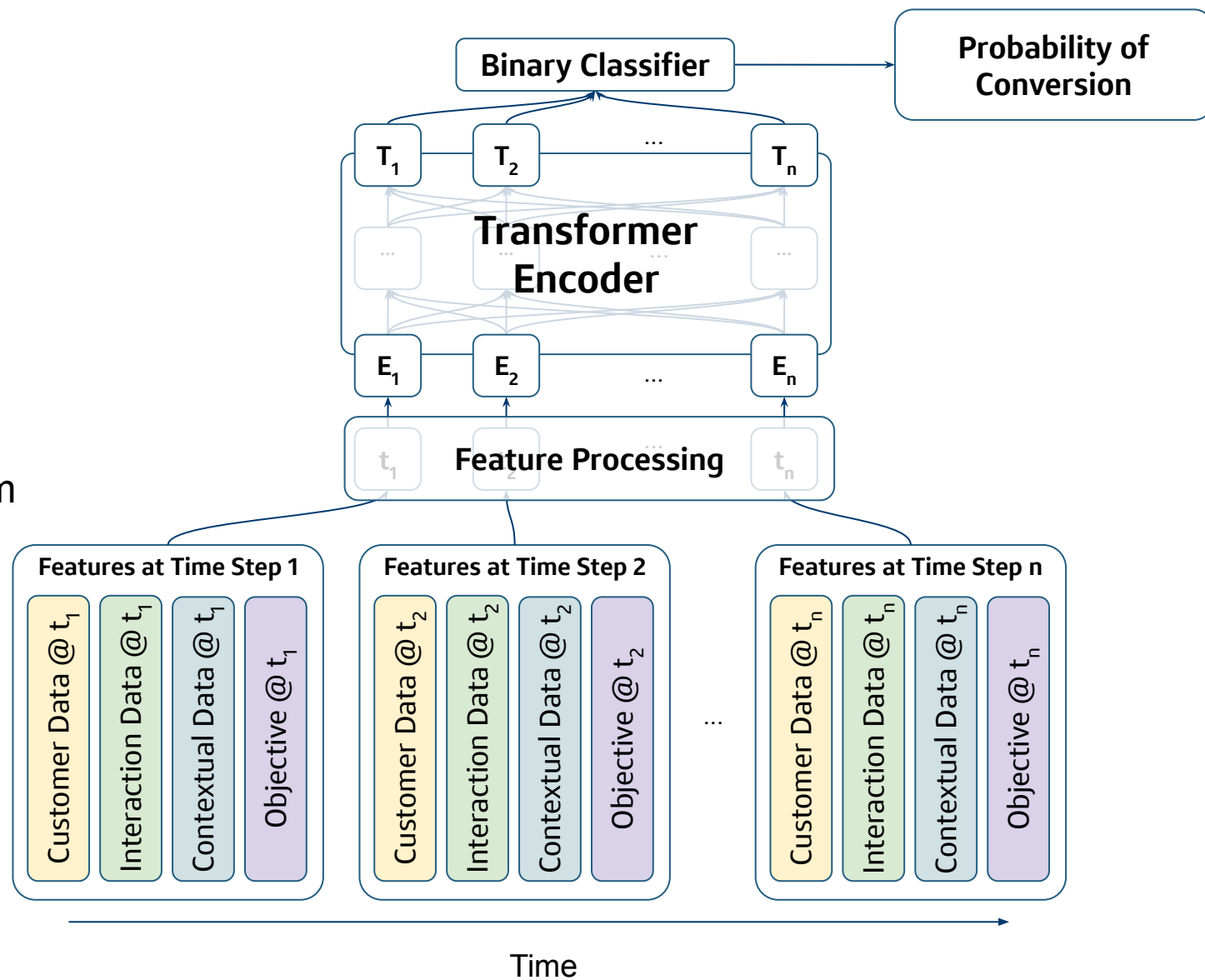


A Transformer-powered architecture can utilize self-attention to learn from user sequential behavior and provide more relevant predictions

A user's journey can be translated into input features at various interaction times

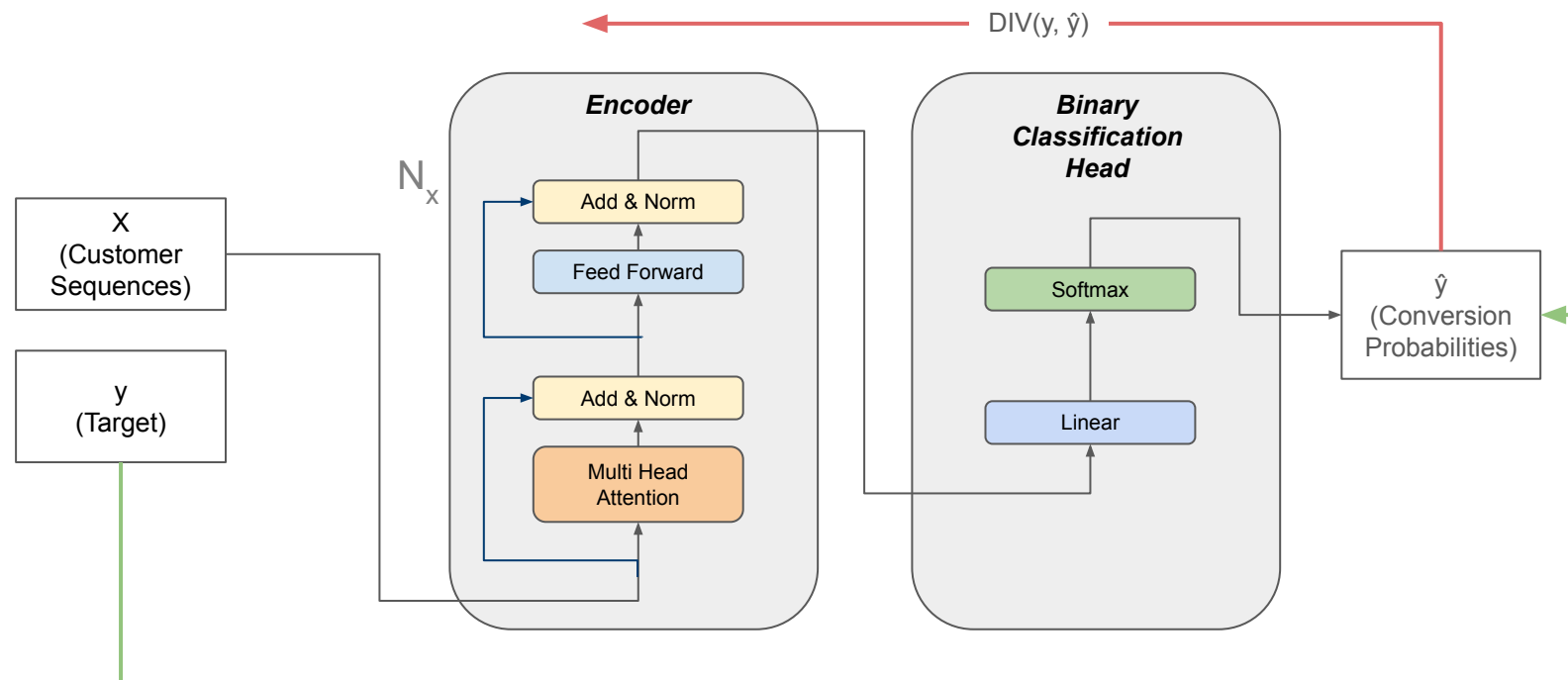
Notable Differences:

- Transformer encoder architecture
- Train from scratch
- Binary classification instead of next-item prediction



This supervised training process differs from how Transformers are typically trained for NLP use cases

- Though Transformers typically have a pretraining component and masking module, this framework has neither
- Binary targets are explicitly passed in and modeled on directly
 - Transformer gets signal from divergence between true target and predicted label, rather than masked target and prediction over mask

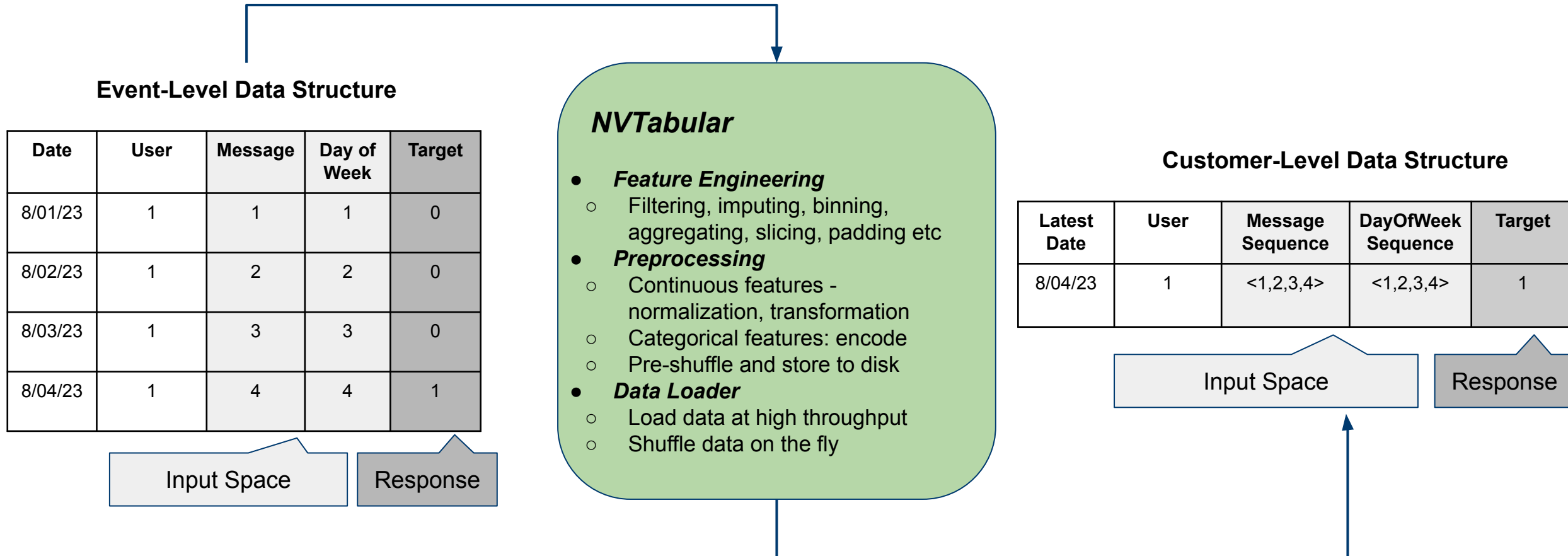


Agenda

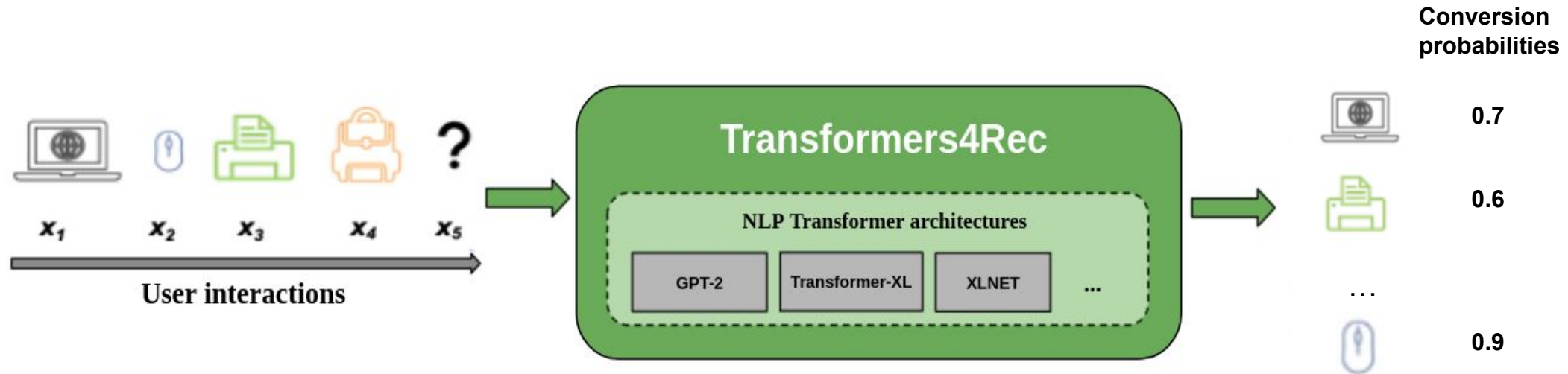
- **Description of the problem**
- **Transformer based Recommender System**
 - **Data preprocessing**
 - **Model training**
 - **Evaluation**
- **Scalability and performance**
- **Conclusion and future work**

For data preprocessing in this framework, we use NVIDIA Merlin's *NVTabular* to make use of GPU powered acceleration

- NVTabular is the ETL part of NVIDIA's Merlin ecosystem that allows for end-to-end GPU-Accelerated and Distributed pipelines.
- It takes us from a tabular representation, like the one we currently model on, to a sequential one.

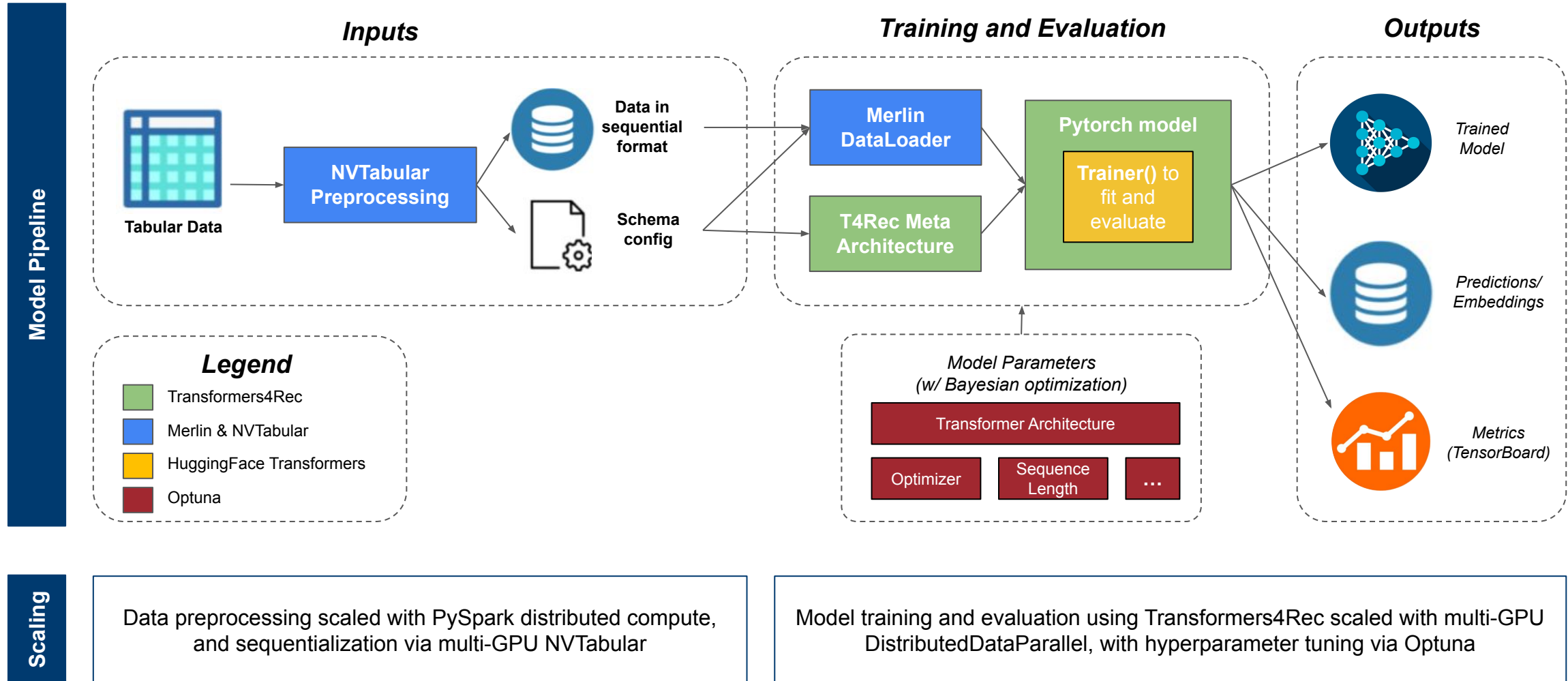


Model Training: NVIDIA Merlin Transformers4Rec library provides GPU-accelerated Deep Learning framework tailored for Recommendation Systems



- Leverages SOTA NLP architectures from the Hugging Face Transformers library, allowing experimentation with many different Transformer architectures
- NVIDIA Merlin Transformers4Rec provides many out-of-the-box capabilities:
 - ✓ Provides a classification framework that allows us to predict probabilities of conversion based on input features — other packages only provide next-item prediction framework
 - ✓ Integrates user feedback and contextual features, enabling any type of sequential tabular data
 - ✓ Modular nature provides flexibility to design ML systems for unique needs

An end-to-end pipeline takes advantage of multi-GPU acceleration and modularity with state of the art open-source transformer models



Typical Evaluation Framework

Dataset	Time Range	Class Balance	# Rows	# Sequences
Training	~1 month	1:1	~ 50 M	~ 2 M
Validation	~3 days	Original ratio (high imbalance)	~ 200 M	~ 3 M
Test	~3 days	Original ratio (high imbalance)	~ 200 M	~ 3 M

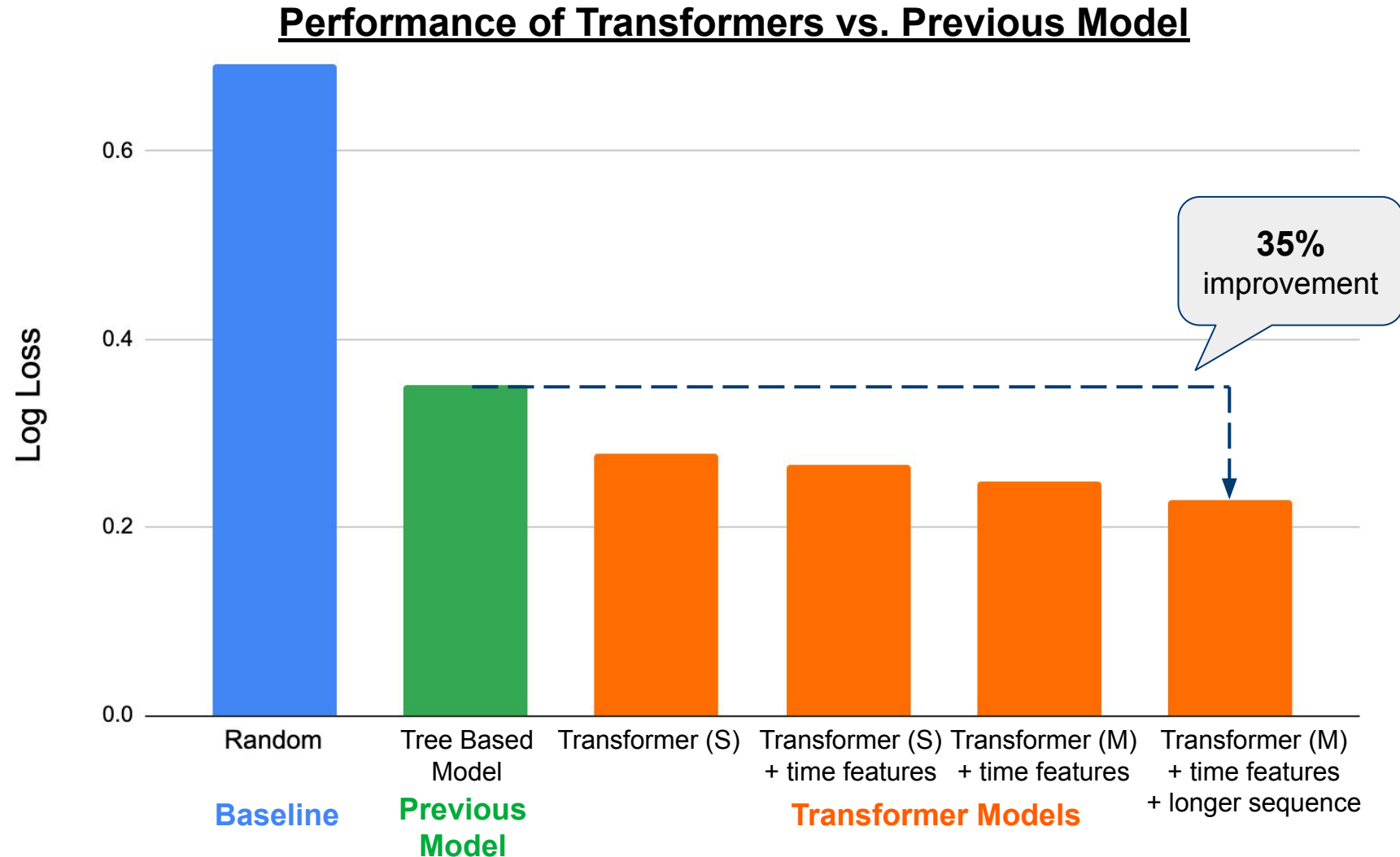
Experimental Methods:

- Transformer Architecture
- Additional Input Features
- Sequence Length
- Performance based on user visit frequency
- Hyperparameter Tuning
- Optimize for Binary Cross Entropy (Log Loss)

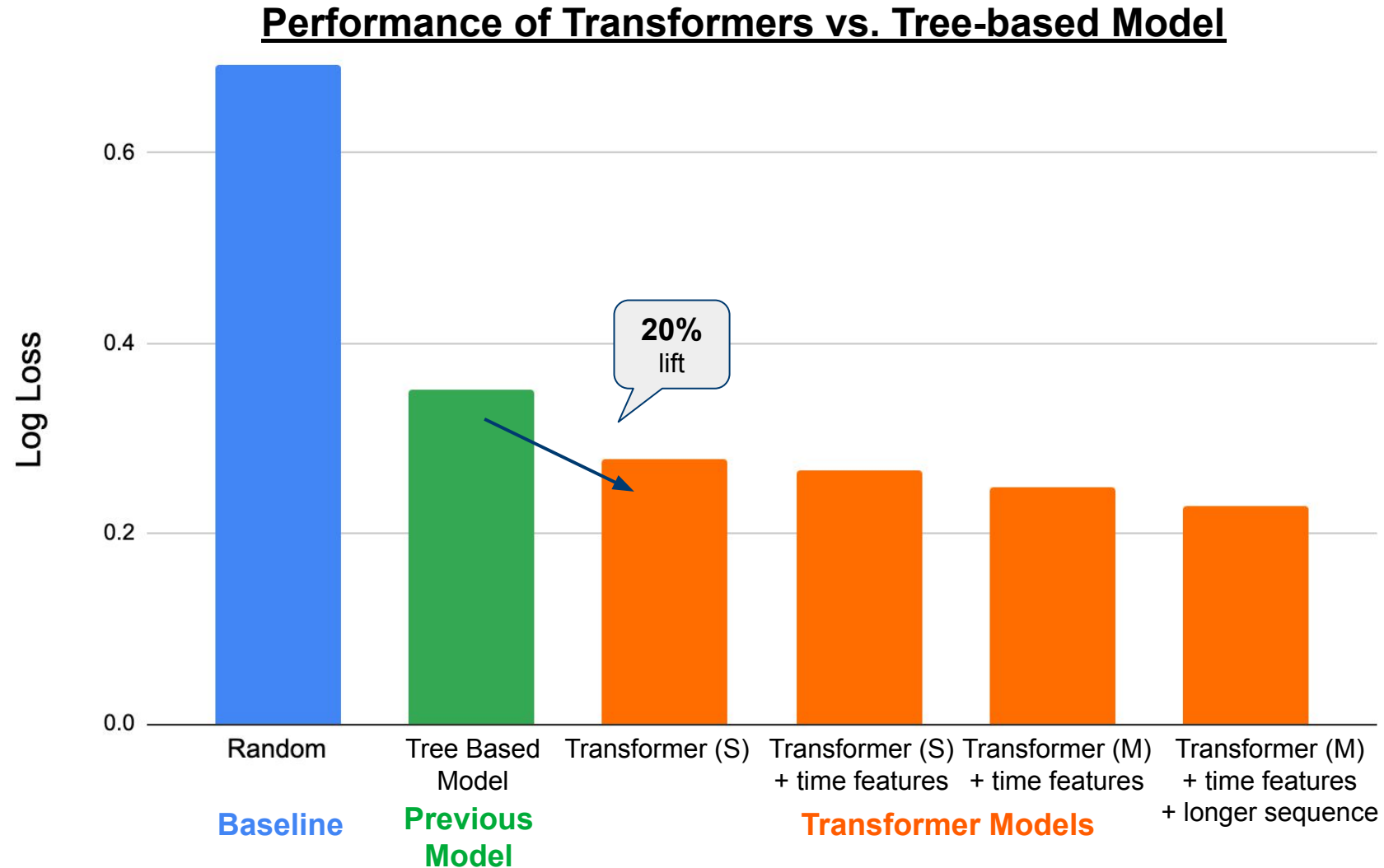
Agenda

- **Description of the problem**
- **Transformer based Recommender System**
 - **Data preprocessing**
 - **Model training**
 - **Evaluation**
- **Scalability and performance**
- **Conclusion and future work**

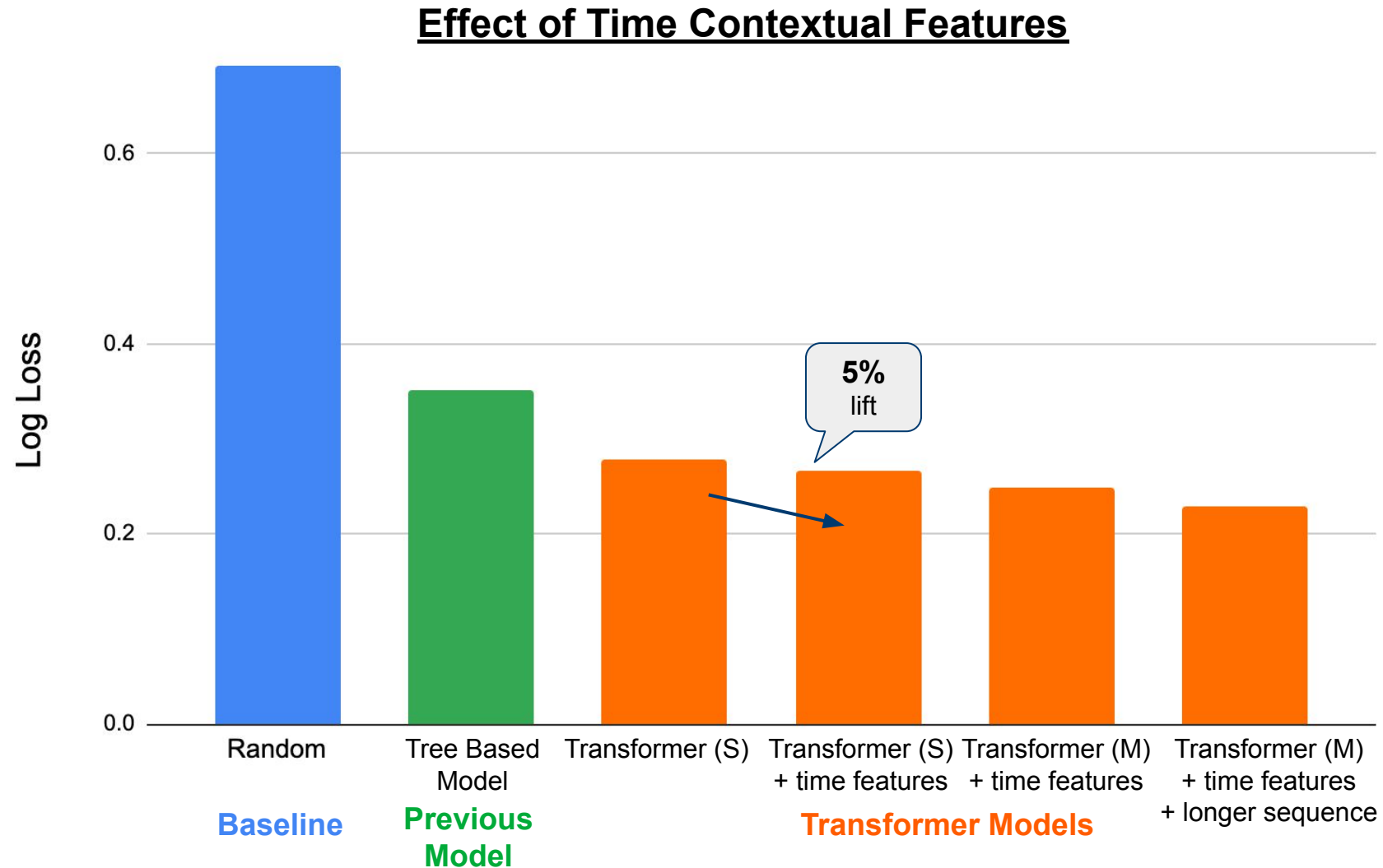
Experimental results show considerable improvement of transformer model performance relative to the previous tree-based classifier, resulting in 35% lift in a key metric



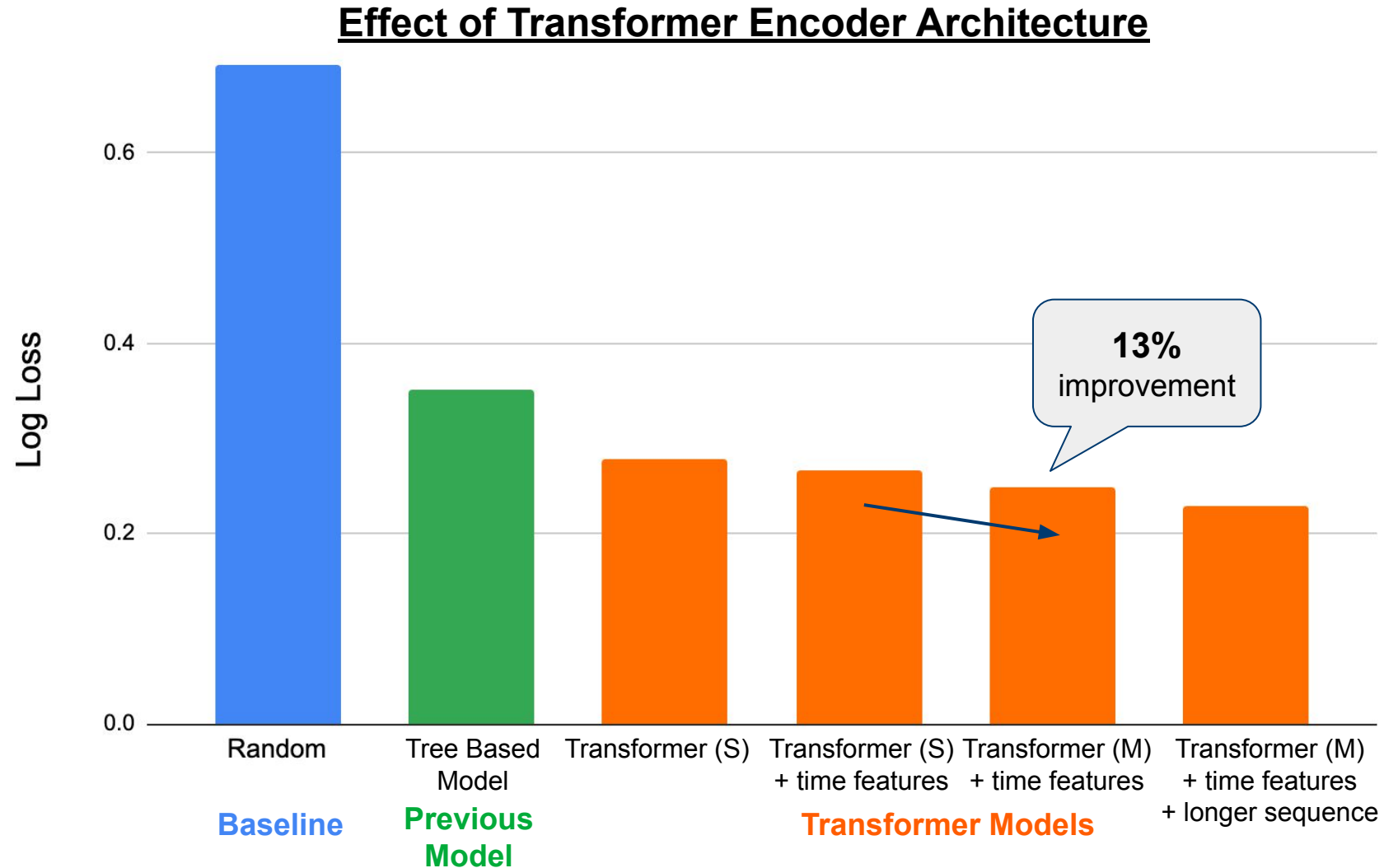
The initial switch to a lightweight transformer encoder architecture offered 20% improvement over the previous tree-based model



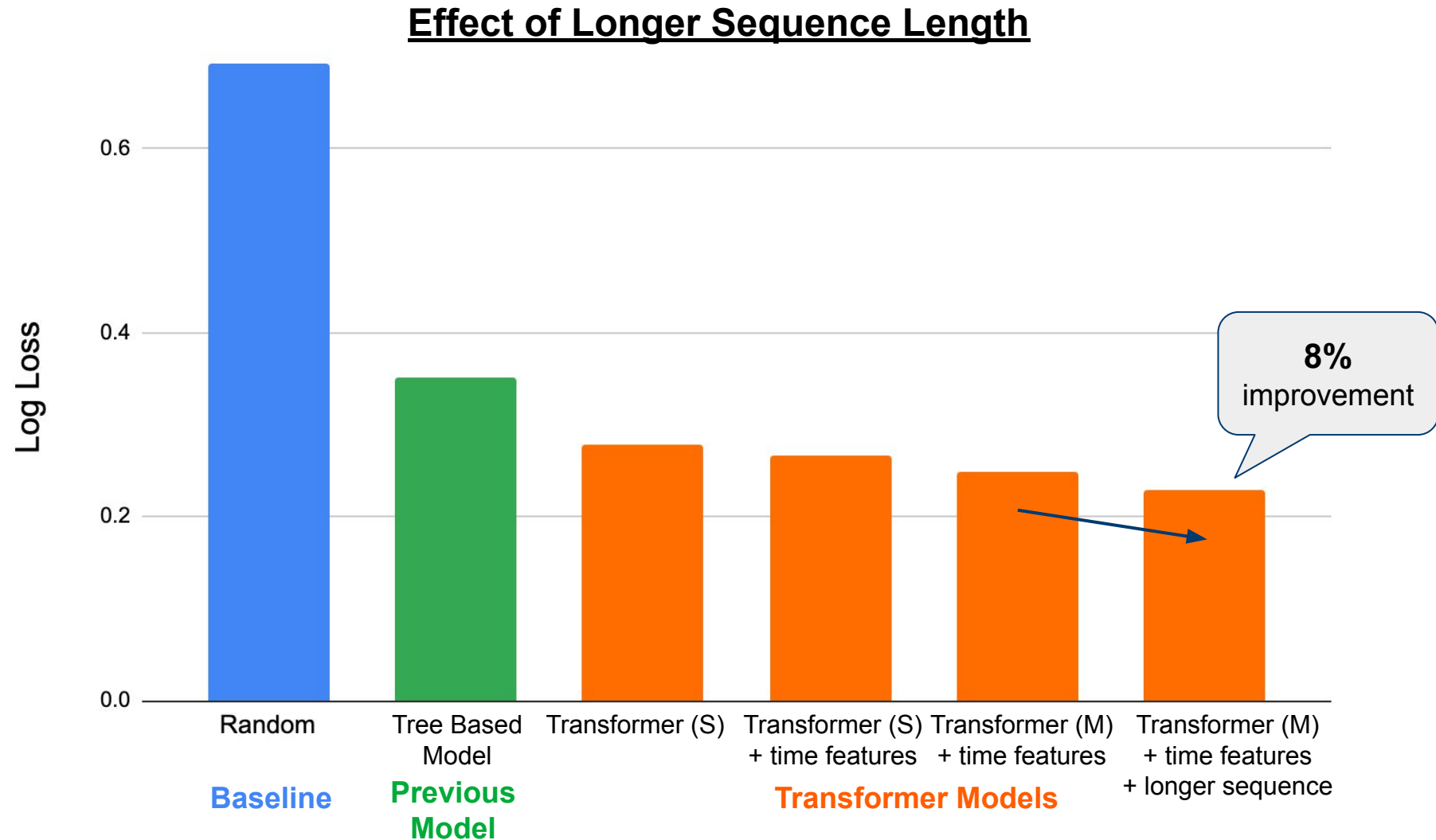
The addition of time contextual features to the base transformer model contributed ~5% in improved performance



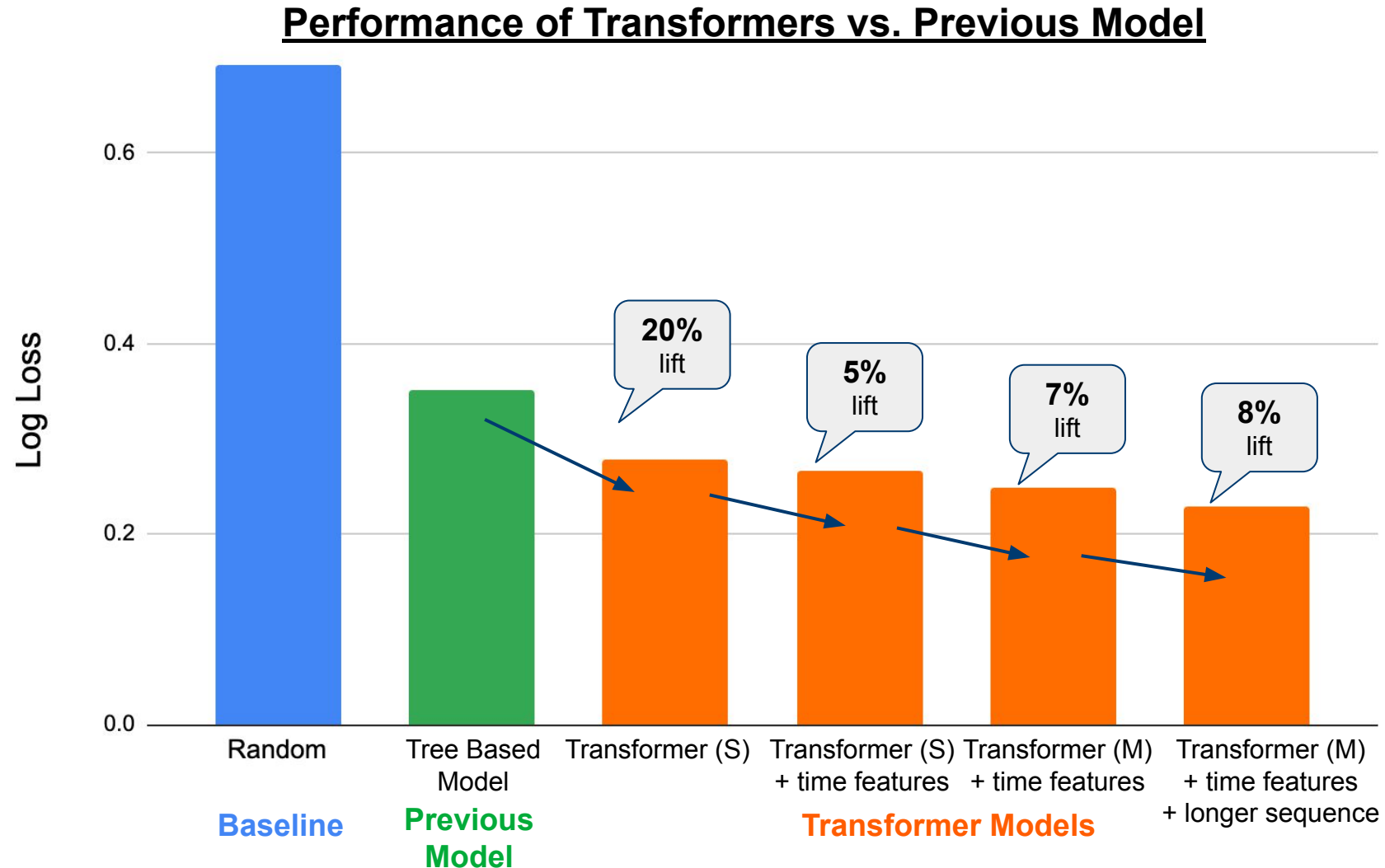
A more sophisticated encoder architecture with ~2x parameter size led to an additional ~13% improvement in performance



Finally, expanding the sequence length $\sim 1.3x$ contributed a further 8% improvement

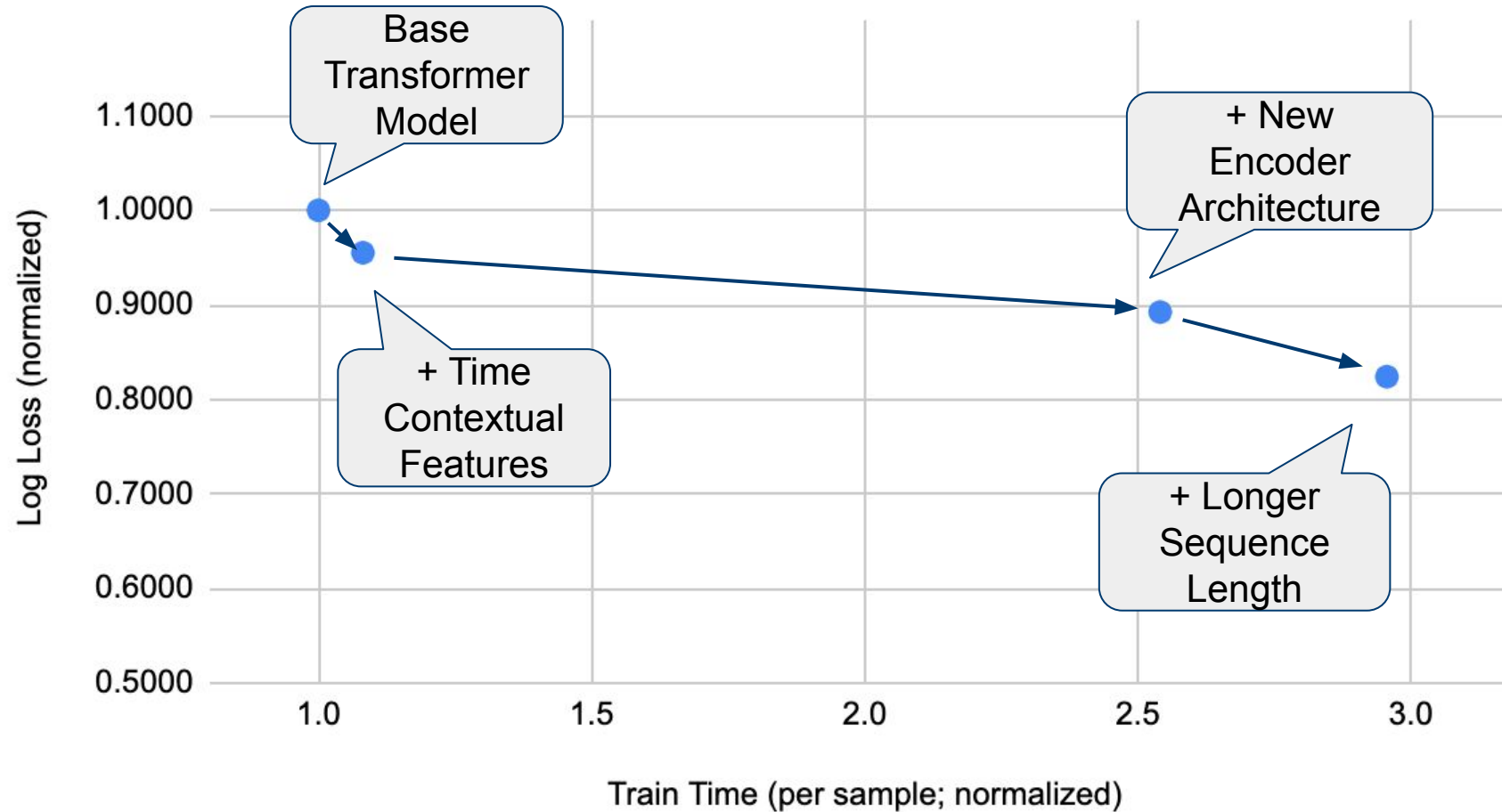


The switch to Transformers, addition of time contextual features, more sophisticated encoder architecture, and longer sequence lengths all contributed significantly to the overall performance improvement of the model



However, each technique leading to model performance improvement also increased model complexity and train/eval time

Model Performance vs. Train Time



Other optimization methods have also proven effective in our efforts to scale up the Transformer-powered recommender model

Method	Experiments	Train Time (↓)	Log Loss (↓)	AUROC (↑)
Architecture	Transformer small → medium	222% ↑	8.0% ↓	1.4% ↑
Feature Addition	Time features	13% ↑	4.5% ↓	~
Max Seq Length	3x longer	274% ↑	7.8% ↓	2.6% ↑
Multi-GPU	1x → 2x A100	20% ↓	~	1.7% ↑
Batch size	2x	47% ↓	~	1.3% ↑
Optimizer choice	SGD / AdamW / AdaFactor	~	~	~
Mixed precision	FP32 → TF32	3% ↓	~	1.2% ↑

Agenda

- **Description of the problem**
- **Transformer based Recommender System**
 - **Data preprocessing**
 - **Model training**
 - **Evaluation**
- **Scalability and performance**
- **Conclusion and future work**

Helpful Takeaways and Actionable Insights

Scaling:

1. Recommend distributed compute and multi-GPU acceleration where possible
 - Speeds up workflows
 - High GPU utilization is important
2. Sequence length vs performance tradeoffs
 - Full self-attention quadratic scaling in computational complexity

Streamlined pipeline:

1. Data-centric methods and model-centric methods both significantly improved performance
 - Scalable data preprocessing
 - Modular/flexible model training frameworks
 - Having both allows for focus on experimentation

Our current and future work includes continued R&D and scaling where appropriate

Architecture

- Test more transformer architectures, optimizers, training schemes
- Apply different positional encoding methods

Features

- Additional input features and embeddings
- Explore best method to add such features
- Explainability and attribution

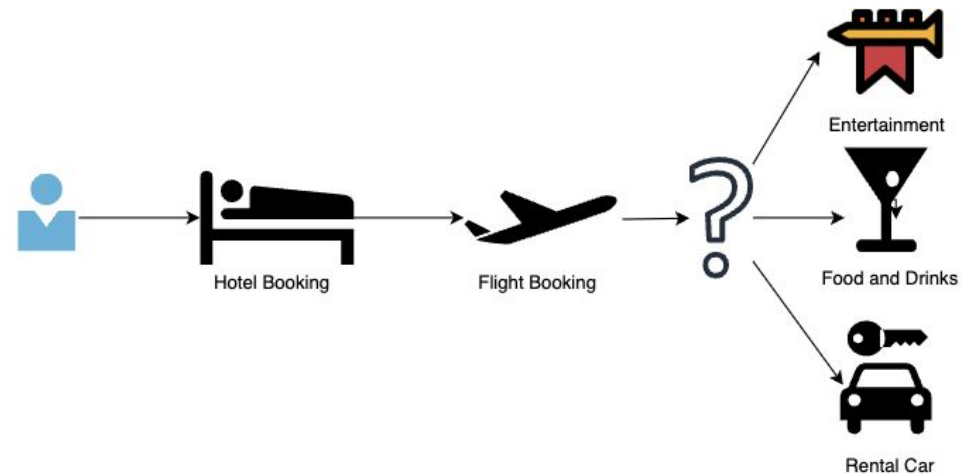
Infrastructure

- Optimize inference throughput and latency
- Real-time applications

Thank you!

Transformer4Rec's popular use case is next-item prediction, but certain financial services applications align better to classification

- NextItemPrediction is the most popular use-case for T4Rec:
 - Show ad of product you think they will book next *based on products booked in recent past*



- Financial Services customers do not typically book multiple products in quick succession
- Instead, we need to classify a sequence of experiences (e.g., impressions) as ending in a conversion or not
 - Then we score ads w/ probability of conversion based on learned parameters, picking highest-scoring ad to show next

The training procedure differs between the general language case and the presented financial services application of Transformers

- In general, language models are *pretrained* (*self-supervised learning*) on a large corpora of text with a specific task
 - Encoder models usually have the task of masked language modeling (MLM)
 - Decoder models usually have the task of casual language modeling (CLM)
- *Pretrained* models are then fine-tuned on downstream tasks, like question answering, sentiment analysis, autocomplete, etc.

Masked Language Modeling: Randomly masking some percentage of the input data and 'filling in the blanks'



Causal Language Modeling: Predicting the next token following a sequence of tokens

