Deep Representation Learning for Prediction of Temporal Event Sets in the Continuous Time Domain – Appendix

Appendix A. Dataset statistics

We provide the dataset statistics in Table 6 corresponding to the datasets mentioned in Section 4.1, which were used in our experiments.

	Synthea	Instacart	MIMIC-III
Total # data-points	55299	110035	1865
Average seq length	7.25	15.65	4.06
Average set length	1.19	7.23	2.79
# i/p event types	211	135	211
# target event types	124	135	124
Dataset type	Synthetic	Real	Real

Table 6: Dataset statistics.

Appendix B. Hyperparameters

We fix the embedding dimensions, $d_{emb} = 100$ throughout all our experiments, i.e., the item embedding dimension, the output dimension of the item-set embedding generator model, and the hidden dimensions of the transformers are all fixed to be 100. We additionally fix the number of transformer encoder layers to 2 and the number of attention-heads to 4.

Contextual Embedding Encoder	
Hidden dimension	100
Learning Rate	0.0005
Dropout	0.1
Num Layers	1
Batch Size	128
Single-step Model	
Hidden dimension	100
Dense Layer dimension	256
Learning Rate	0.003
Dropout	0.1
Num Encoder Layers	2
Batch Size	512
Max Seq Length	500
Dice ϵ	0.1
Loss $\lambda_1, \lambda_2, \lambda_3$	$0.85, 1, 0.2$

Table 7: Hyper-parameter table

Appendix C. Training Details

For the item embedding generator that we used in Section 3.2, we use s single layer of densely connected (feed-forward) neural network as our auxiliary encoder $\mathcal{A}_E$ without any activation at the output (embedding) layer. We use transformers as sequence encoders in the Single-step training approach (Section 3). We use transformers with 2 encoder layers as our sequential encoder while reporting the results. These are not the best possible results, rather, we tried to keep the model's capacity/expressive-power/architecture similar to baselines for a fair comparison.

However, we have experimented with LSTMs as well. It should be noted that due to the bi-directional embeddings in both the Bi-LSTMs and Transformers, a specialized dataset preprocessing is required, which can be skipped if using simple LSTMs. This reduces the training time in LSTMs. However, the inference time remains asymptotically the same.

We use an $80 - 20$ train-test split in our datasets, and within the training split, we further use 10% of the data for validation. We use Nvidia A100 GPUs to run our experiments.

Appendix D. Notations

Table 8: Description of the notations used in the main paper

Notation	Description
$\mathcal{S}$	It defines the input sequence of event sets in continuous time domain
$\mathbf{s}_k$	It denotes the event set in the input sequence $\mathcal{S}$
$\mathcal{I}$	It defines the set of all possible events
$\mathbf{f}_k$	It denotes the set of features associated with $\mathbf{s}_k$ in the input sequence $\mathcal{S}$
$\mathbf{t}_k$	It denotes the timestamp associated with $\mathbf{s}_k$ in the input sequence $\mathcal{S}$
$\mathcal{T}$	It defines the set of target events $\mathcal{T} \subset \mathcal{I}$
$\mathcal{M}$	It denotes the model being trained for a given task
$\mathcal{A}_E$	It denotes the auxiliary encoder model
$\mathbf{v}_{emb}$	It denotes the embedding for an event $\mathbf{i}$ from the auxiliary encoder model $\mathcal{A}_E$
$\mathbf{d}_{emb}$	It denotes the dimension of the event embedding from the auxiliary encoder model $\mathcal{A}_E$
$\mathcal{L}_{aux}$	It defines the auxiliary contextual loss objective
$\mathcal{H}_k$	It denotes all the previous set of events along with their corresponding timestamps and features until $\mathbf{s}_k$
$\hat{[\cdot]}$	A hat over any symbol indicates that it is the model's prediction
$\mathbf{e}_{k+1}$	It denotes the target event set corresponding to the input history $\mathcal{H}_k$
$\mu^j_{\hat{\mathbf{e}}_{k+1}}$	It defines the Gaussian distributional parameter - mean of j^{th} mixture for the target event set $\mathbf{e}_{k+1}$
$\sigma^j_{\hat{\mathbf{e}}_{k+1}}$	It defines the Gaussian distributional parameter - standard deviation of j^{th} mixture for the target event set $\mathbf{e}_{k+1}$
$\alpha^j_{\hat{\mathbf{e}}_{k+1}}$	It defines the mixture coefficient of j^{th} mixture for the target event set $\mathbf{e}_{k+1}$
$\mu^j_{\hat{\mathbf{t}}_{k+1}}$	It defines the Gaussian distributional parameter - mean of j^{th} mixture for the target time $\mathbf{t}_{k+1}$
$\sigma^j_{\hat{\mathbf{t}}_{k+1}}$	It defines the Gaussian distributional parameter - standard deviation of j^{th} mixture for the target time $\mathbf{t}_{k+1}$
$\alpha^j_{\hat{\mathbf{t}}_{k+1}}$	It defines the mixture coefficient of j^{th} mixture for the target time $\mathbf{t}_{k+1}$

Appendix E. Additional Tables

Table 9: **Additional metrics for Temporal Event-set Modeling Results.** We compare our approaches to baselines, and in addition to Table 1 in the main paper, report the F-scores and RMSEs.

Training method	Synthea		Instacart	
	Event-set pred (F-score)	Time pred (RMSE)	Event-set pred (F-score)	Time pred (RMSE)
Baselines:				
Neural Hawkes Process	0.02	6.68	0.27	0.18
Transformer Hawkes Process	0.08	5.85	0.34	0.17
Hierarchical Model	0.10	6.10	0.35	0.12
Ours:				
TESET	0.32	4.73	0.45	0.06
TESET + Contextual Embeddings	**0.47**	**4.28**	**0.64**	**0.04**

Table 10: **Additional Fine-tuning results.** In addition to Table 2 in the manuscript, we report additional metrics: F-score and RMSE, and compare fine-tuning vs training from scratch results for our method vs the baselines.

FT?	Training method	Synthea		Instacart	
		Event-set given time (F-score)	Time given event (RMSE)	Event-set given time (F-score)	Time given event (RMSE)
Trained from scratch	Neural Hawkes Process	0.25	8.66	0.48	4.39
	Transformer Hawkes Process	0.27	7.08	0.45	3.88
	Hierarchical Model	0.24	7.81	0.44	4.50
	TESET (Ours)	*0.32*	*6.97*	*0.52*	*3.12*
Fine-tuned	Neural Hawkes Process	0.09	9.40	0.41	5.03
	Transformer Hawkes Process	0.18	7.55	0.45	4.75
	Hierarchical Model	0.16	8.13	0.48	4.99
	TESET (Ours)	***0.44***	***6.27***	***0.60***	***2.25***

Appendix F. Additional Figures

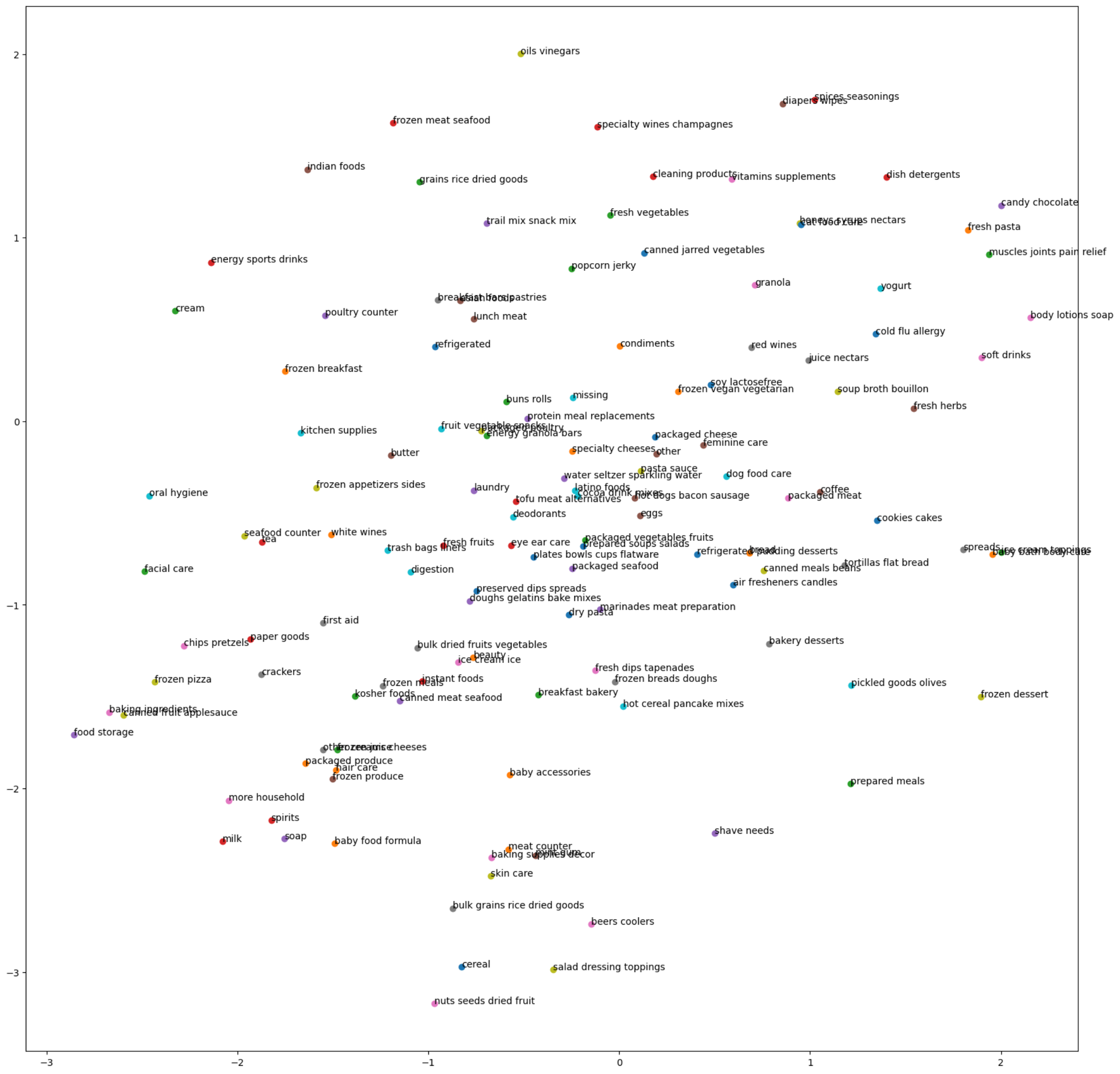

Figure 7: 2D t-SNE of the embedding space after the first step of training (learning the contextual representations) for the Instacart Dataset. It can be observed that the clusters formed in the embedding space are valid. For instance, frozen meals and instant noodles form a cluster, soy, and vegan items are in the same cluster, and bakery and dough are also in close proximity to each other.

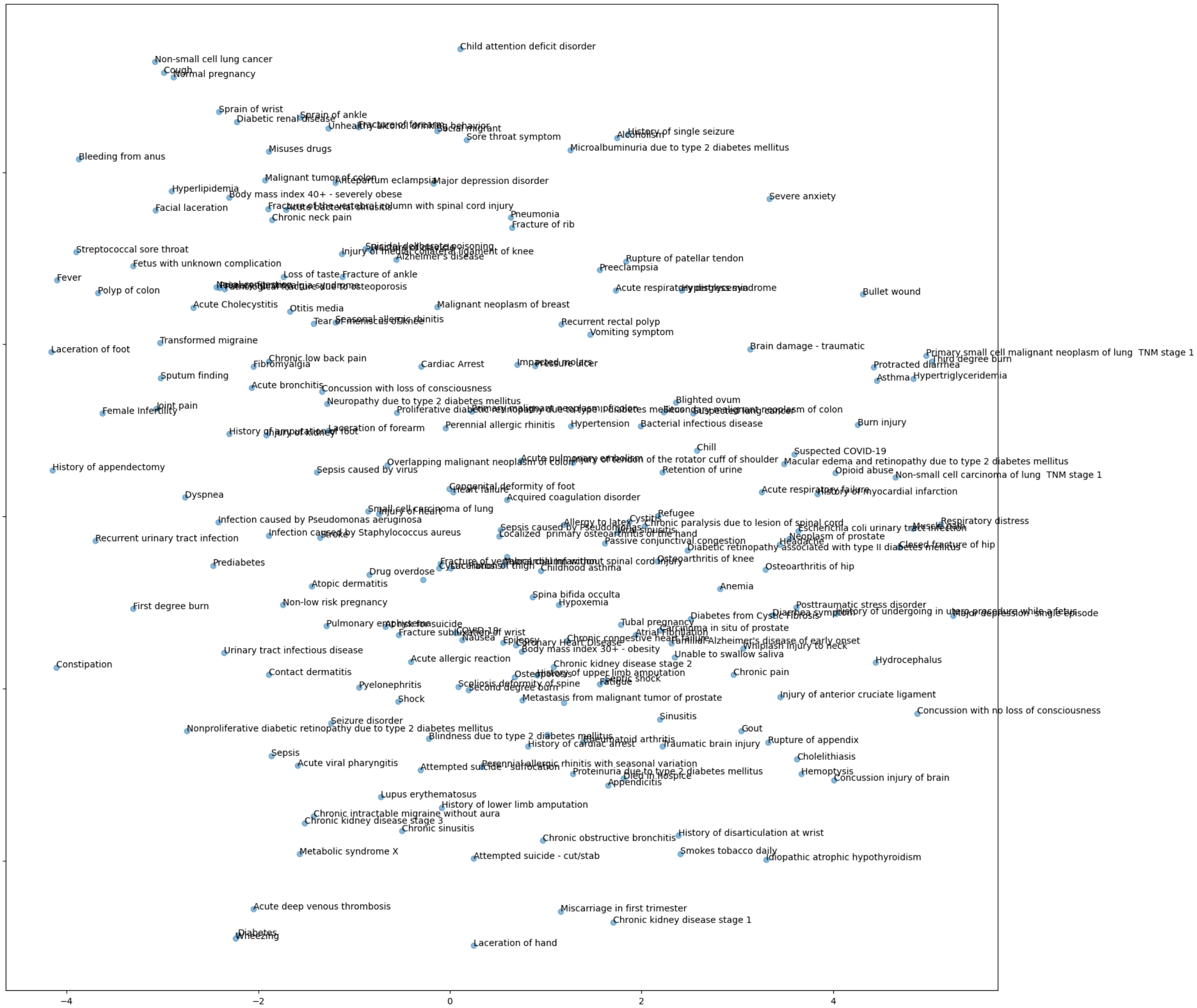

Figure 8: 2D t-SNE (full) of the embedding space after the first step of training (learning the contextual representations) for the Synthea Dataset.