References

Alibaba. User behavior data from taobao for recommendation, 2018.

Bakhtin, A., Gross, S., Ott, M., Deng, Y., Ranzato, M., and Szlam, A. Real or fake? learning to discriminate machine from human generated text. 2019.

Boyd, A., Bamler, R., Mandt, S., and Smyth, P. User-dependent neural sequence models for continuous-time event data. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

Brakel, P., Stroobandt, D., and Schrauwen, B. Training energy-based models for time-series imputation. *J. Mach. Learn. Res.*, 14(1):2771–2797, jan 2013. ISSN 1532-4435.

Daley, D. J. and Vere-Jones, D. *An Introduction to the Theory of Point Processes, Volume II: General Theory and Structure*. Springer, 2007.

Deng, Y., Bakhtin, A., Ott, M., Szlam, A., and Ranzato, M. Residual energy-based models for text generation. In *International Conference on Learning Representations*, 2020.

Deshpande, P., Marathe, K., De, A., and Sarawagi, S. Long horizon forecasting with temporal point processes. In *Proceedings of the 14th ACM International Conference on Web Search and Data Mining*. ACM, mar 2021.

Du, N., Dai, H., Trivedi, R., Upadhyay, U., Gomez-Rodriguez, M., and Song, L. Recurrent marked temporal point processes: Embedding event history to vector. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2016.

Du, Y. and Mordatch, I. Implicit generation and modeling with energy based models. In Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

Enguehard, J., Busbridge, D., Bozson, A., Woodcock, C., and Hammerla, N. Neural temporal point processes [for] modelling electronic health records. In *Proceedings of Machine Learning Research*, volume 136, pp. 85–113, 2020. NeurIPS 2020 Workshop on Machine Learning for Health (ML4H).

Goyal, K. *Characterizing and Overcoming the Limitations of Neural Autoregressive Models*. PhD thesis, Carnegie Mellon University, 2021.

Guan, J., Mao, X., Fan, C., Liu, Z., Ding, W., and Huang, M. Long text generation by modeling sentence-level and discourse-level coherence. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, 2021.

Guo, J., Lu, S., Cai, H., Zhang, W., Yu, Y., and Wang, J. Long text generation via adversarial training with leaked information. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2018.

Gutmann, M. and Hyvärinen, A. Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. In *Proceedings of the International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2010.

Hawkes, A. G. Spectra of some self-exciting and mutually exciting point processes. *Biometrika*, 1971.

Hinton, G. E. Training products of experts by minimizing contrastive divergence. *Neural Comput.*, 14(8):1771–1800, aug 2002. ISSN 0899-7667.

Hochreiter, S. and Schmidhuber, J. Long short-term memory. *Neural Computation*, 1997.

Hopfield, J. Neural networks and physical systems with emergent collective computationalabilities. *National Academy of Sciences of the USA*, 79, 1982.

Kingma, D. and Ba, J. Adam: A method for stochastic optimization. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2015.

Le Guen, V. and Thome, N. Shape and time distortion loss for training deep time series forecasting models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

LeCun, Y., Chopra, S., Hadsell, R., Ranzato, M., and Huang, F.-J. A tutorial on energy-based learning. 2006.

Leskovec, J. and Krevl, A. SNAP Datasets: Stanford large network dataset collection, 2014.

Lewis, P. A. and Shedler, G. S. Simulation of nonhomogeneous Poisson processes by thinning. *Naval Research Logistics Quarterly*, 1979.

Lin, C.-C. and McCarthy, A. D. On the uncomputability of partition functions in energy-based sequence models. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2022.

Lin, C.-C., Jaech, A., Li, X., Gormley, M., and Eisner, J. Limitations of autoregressive models and their alternatives. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*, 2021.

Liniger, T. J. *Multivariate Hawkes processes*. Diss., Eidgenössische Technische Hochschule ETH Zürich, Nr. 18403, 2009.

Ma, Z. and Collins, M. Noise-contrastive estimation and negative sampling for conditional models: Consistency and statistical efficiency. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2018.

Mei, H. and Eisner, J. The neural Hawkes process: A neurally self-modulating multivariate point process. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.

Mei, H., Qin, G., and Eisner, J. Imputing missing events in continuous-time event streams. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2019.

Mei, H., Qin, G., Xu, M., and Eisner, J. Neural Datalog through time: Informed temporal modeling via logical specification. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2020a.

Mei, H., Wan, T., and Eisner, J. Noise-contrastive estimation for multivariate point processes. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020b.

Mnih, A. and Teh, Y. W. A fast and simple algorithm for training neural probabilistic language models. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2012.

Ngiam, J., Chen, Z., Koh, P. W., and Ng, A. Y. Learning deep energy models. ICML'11, pp. 1105–1112, Madison, WI, USA, 2011. Omnipress. ISBN 9781450306195.

O'Connor, J. and Andreas, J. What context features can transformer language models use? In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, 2021.

Omi, T., Ueda, N., and Aihara, K. Fully neural network based model for general temporal point processes. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

Oord, A. v. d., Dieleman, S., Zen, H., Simonyan, K., Vinyals, O., Graves, A., Kalchbrenner, N., Senior, A., and Kavukcuoglu, K. Wavenet: A generative model for raw audio. *arXiv preprint arXiv:1609.03499*, 2016.

Pang, B., Zhao, T., Xie, X., and Wu, Y. Trajectory prediction with latent belief energy-based model. *IEEE Conference on Computer Vision and Pattern Recognition*, 04 2021.

Paszke, A., Gross, S., Chintala, S., Chanan, G., Yang, E., DeVito, Z., Lin, Z., Desmaison, A., Antiga, L., and Lerer, A. Automatic differentiation in PyTorch. 2017.

Pérez, J., Marinković, J., and Barceló, P. On the turing completeness of modern neural network architectures. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2019.

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., and Sutskever, I. Language models are unsupervised multitask learners. 2019.

Ranzato, M., Boureau, Y.-L., Chopra, S., and LeCun, Y. A unified energy-based framework for unsupervised learning. In Meila, M. and Shen, X. (eds.), *Proceedings of the Eleventh International Conference on Artificial Intelligence and Statistics*, volume 2 of *Proceedings of Machine Learning Research*, pp. 371–379, San Juan, Puerto Rico, 21–24 Mar 2007. PMLR.

Ranzato, M., Chopra, S., Auli, M., and Zaremba, W. Sequence level training with recurrent neural networks. In *International Conference on Learning Representation*, 2016.

Sharma, K., Zhang, Y., Ferrara, E., and Liu, Y. Identifying coordinated accounts on social media through hidden influence and group behaviours. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2021.

Shchur, O., Biloš, M., and Günnemann, S. Intensity-free learning of temporal point processes. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2020.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. Attention is all you need. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.

Whong, C. FOILing NYC's taxi trip data, 2014.

Xiao, S., Yan, J., Farajtabar, M., Song, L., Yang, X., and Zha, H. Joint modeling of event sequence and time series with attentional twin recurrent neural networks. *arXiv preprint arXiv:1703.08524*, 2017a.

Xiao, S., Yan, J., Yang, X., Zha, H., and Chu, S. Modeling the intensity function of point process via recurrent neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2017b.

Xie, J., Zhu, S. C., and Wu, Y. N. Learning energy-based spatial-temporal generative convnets for dynamic patterns. *IEEE transactions on pattern analysis and machine intelligence*, 2019.

Yang, C., Mei, H., and Eisner, J. Transformer embeddings of irregularly spaced events and their participants. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2022.

Yu, R., Zheng, S., Anandkumar, A., and Yue, Y. Long-term forecasting using higher order tensor rnns. 2019.

Zhang, Q., Lipani, A., Kirnap, O., and Yilmaz, E. Self-attentive Hawkes process. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2020.

Zhu, S., Zhang, M., Ding, R., and Xie, Y. Deep Fourier kernel for self-attentive point processes. In *International Conference on Artificial Intelligence and Statistics*, 2021.

Zuo, S., Jiang, H., Li, Z., Zhao, T., and Zha, H. Transformer Hawkes process. In *International Conference on Machine Learning*, pp. 11692–11702. PMLR, 2020.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes] See section 7
   (c) Did you discuss any potential negative societal impacts of your work? [Yes] See section 7.
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes] See sections 2 and 3 and Appendix A.1.
   (b) Did you include complete proofs of all theoretical results? [Yes] See Appendix A.1.
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] See the supplemental material.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] See Appendix B.1 and Appendix B.3.
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes] See section 5.
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] See Appendix B.3.
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes] See Appendix B.3
   (b) Did you mention the license of the assets? [Yes] See Appendix B.2.
   (c) Did you include any new assets either in the supplemental material or as a URL? [Yes] We include our code in the supplementary material.
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A] We use existing datasets that are publicly available and already anonymized.
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A] We use existing datasets that are publicly available and already anonymized.
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]

Appendices

A Method Details

A.1 Derivation of NCE Objectives

Given a prefix $x_{[0,T]}$, we have the true continuation $x^{(0)}_{(T,T']}$ and N noise samples $x^{(1)}_{(T,T']}, \ldots, x^{(N)}_{(T,T']}$. By concatenating the prefix and each (true or noise) continuation, we obtain $N+1$ completed sequences $x^{(0)}_{[0,T']}, x^{(1)}_{[0,T']}, \ldots, x^{(N)}_{[0,T']}$.

Binary-NCE Objective. For each completed sequence $x^{(n)}_{[0,T']}$, we learn to classify whether it is real data or noise data. The unnormalized probability for each case is:

$$\tilde{p}\,(\text{it is real data}) = p_{\text{HYPRO}}\left(x^{(n)}_{(T,T']} \mid x_{[0,T]}\right) = p_{\text{auto}}\left(x^{(n)}_{(T,T']} \mid x_{[0,T]}\right) \frac{\exp\left(-E_\theta(x^{(n)}_{[0,T']})\right)}{Z_\theta\left(x_{[0,T]}\right)} \tag{7}$$

$$\tilde{p}\,(\text{it is noise data}) = p_{\text{auto}}\left(x^{(n)}_{(T,T']} \mid x_{[0,T]}\right) \tag{8}$$

Then the normalized probabilities are:

$$p\,(\text{it is real data}) = \frac{\tilde{p}(\text{it is real data})}{\tilde{p}(\text{it is real data})+\tilde{p}(\text{it is noise data})} = \frac{\exp\left(-E_\theta(x^{(n)}_{[0,T']})\right)}{Z_\theta\left(x_{[0,T]}\right)+\exp\left(-E_\theta(x^{(n)}_{[0,T']})\right)} \tag{9}$$

$$p\,(\text{it is noise data}) = \frac{\tilde{p}(\text{it is noise data})}{\tilde{p}(\text{it is real data})+\tilde{p}(\text{it is noise data})} = \frac{Z_\theta\left(x_{[0,T]}\right)}{Z_\theta\left(x_{[0,T]}\right)+\exp\left(-E_\theta(x^{(n)}_{[0,T']})\right)} \tag{10}$$

Following previous work (Mnih & Teh, 2012), we assume that the model is self-normalized, i.e., $Z_\theta\left(x_{[0,T]}\right) = 1$. Then the normalized probabilities become

$$p\,(\text{it is real data}) = \frac{\exp\left(-E_\theta(x^{(n)}_{[0,T']})\right)}{1+\exp\left(-E_\theta(x^{(n)}_{[0,T']})\right)} = \sigma\left(-E_\theta(x^{(n)}_{[0,T']})\right) \tag{11}$$

$$p\,(\text{it is noise data}) = \frac{1}{1+\exp\left(-E_\theta(x^{(n)}_{[0,T']})\right)} = \sigma\left(E_\theta(x^{(n)}_{[0,T']}))\right) \tag{12}$$

where σ is the sigmoid function.

For the true completed sequence $x^{(0)}_{[0,T']}$, we maximize the log probability that it is real data, i.e., $\log p\,(\text{it is real data})$; for each noise sequence $x^{(n)}_{[0,T']}$, we maximize the log probability that it is noise data, i.e., $\log p\,(\text{it is noise data})$. The Binary-NCE objective turns out to be equation (3), i.e.,

$$J_{\text{binary}} = \log\sigma\left(-E_\theta(x^{(0)}_{[0,T']})\right) + \sum_{n=1}^{N}\log\sigma\left(E_\theta(x^{(n)}_{[0,T']}))\right)$$

Multi-NCE Objective. For these $N+1$ sequences, we learn to discriminate the true sequence against the noise sequences. For each of them $x^{(n)}_{[0,T']}$, the following is the unnormalized probability that it is real data but all others are noise:

$$\tilde{p}\left(x^{(n)}_{[0,T']} \text{ is real, others are noise}\right) = p_{\text{HYPRO}}\left(x^{(n)}_{(T,T']} \mid x_{[0,T]}\right)\prod_{n'\neq n} p_{\text{auto}}\left(x^{(n')}_{(T,T']} \mid x_{[0,T]}\right) \tag{13}$$

where can be rearranged to be

$$\tilde{p}\left(x^{(n)}_{[0,T']} \text{ is real, others are noise}\right) = \frac{\exp\left(-E_\theta(x^{(n)}_{[0,T']})\right)}{Z_\theta\left(x_{[0,T]}\right)}\prod_{n=0}^{N} p_{\text{auto}}\left(x^{(n)}_{(T,T']} \mid x_{[0,T]}\right) \tag{14}$$

Note that $\frac{1}{Z}\prod_{n=0}^{N} p_{\text{auto}}$ is constant with respect to n. So we can ignore that term and obtain

$$\tilde{p}\left(x^{(n)}_{[0,T']} \text{ is real, others are noise}\right) \propto \exp\left(-E_\theta(x^{(n)}_{[0,T']})\right) \tag{15}$$

Therefore, we can obtain the normalized probability that $x^{(0)}_{[0,T']}$ is real data as below

$$p\left(x^{(0)}_{[0,T']} \text{ is real data}\right) = \frac{\exp\left(-E_\theta(x^{(0)}_{[0,T']}))\right)}{\sum_{n=0}^{N}\exp\left(-E_\theta(x^{(n)}_{[0,T']}))\right)} \tag{16}$$

Algorithm 3 Thinning Algorithm.

Input: an event sequence $x_{[0,T]}$ over the given interval $[0,T]$ and an interval $(T,T']$ of interest; trained autoregressive model p_{auto}
Output: a sampled continuation $x_{(T,T']}$

procedure THINNING($x_{[0,T]}, T', p_{\text{auto}}$)
  initialize $x_{(T,T']}$ as empty
  ▷ *use the thinning algorithm to draw each noise sequences from the autoregressive model p_{auto}*
  $t_0 \leftarrow T$; $i \leftarrow 1$; $\mathcal{H} \leftarrow x_{[0,T]}$
  while $t_0 < T'$:   ▷ *draw next event if we haven't exceeded the time boundary T' yet*
    ▷ *upper bound λ^* can be found for NHP and AttNHP.*
    ▷ *technical details can be found in Mei & Eisner (2017) and Yang et al. (2022).*
    find upper bound $\lambda^* \geq \sum_{k=1}^{K} \lambda_k(t \mid \mathcal{H})$ for all $t \in (t_0, \infty)$ ▷ *compute sampling intensity*
    repeat
      draw $\Delta \sim \text{Exp}(\lambda^*)$; t_0 += Δ ▷ *time of next proposed noise event*
      $u \sim \text{Unif}(0,1)$
    until $u\lambda^* \leq \sum_{k=1}^{K} \lambda_k(t_0 \mid \mathcal{H})$ ▷ *accept proposed next noise event with prob $\sum_{k=1}^{K} \lambda_k / \lambda^*$*
    if $t_0 > T'$: **break**
    draw $k \in \{1, \ldots, K\}$ where probability of k is $\propto \lambda_k(t_0 \mid \mathcal{H})$
    append (t_0, k) to both $\mathcal{H}$ and $x_{(T,T']}$
  return $x_{(T,T']}$

Note that the normalizing constant Z doesn't show up in the normalized probability since it has been cancelled out as a part of the $\frac{1}{Z}\prod_{n=0}^{N} p_{\text{auto}}$ constant. That is, unlike the Binary-NCE case, we do not need to assume self-normalization in this Multi-NCE case.

We maximize the log probability that $x^{(0)}_{[0,T']}$ is real data, i.e., $\log p\left(x^{(0)}_{[0,T']} \text{ is real data}\right)$; the Multi-NCE objective turns out to be equation (4), i.e.,

$$J_{\text{multi}} = -E_\theta(x^{(0)}_{[0,T']}) - \log \sum_{n=0}^{N} \exp\left(-E_\theta(x^{(n)}_{[0,T']})\right)$$

A.2 Sampling Algorithm Details

In section 3.2, we described a sampling method to approximately draw $x_{(T,T']}$ from p_{HYPRO}. It calls the thinning algorithm, which we describe in Algorithm 3.

B Experimental Details

B.1 Dataset Details

Taobao (Alibaba, 2018). This dataset contains time-stamped user click behaviors on Taobao shopping pages from November 25 to December 03, 2017. Each user has a sequence of item click events with each event containing the timestamp and the category of the item. The categories of all items are first ranked by frequencies and the top 16 are kept while the rests are merged into one category, with each category corresponding to an event type. We work on a subset of 2000 most active users with average sequence length 58 and then end up with $K = 17$ event types. We randomly sampled disjoint train, dev and test sets with 1300, 200 and 500 sequences from the dataset. Given the average inter-arrival time 0.06 (time unit is 3 hours), we choose the prediction horizon as 1.5 that approximately has 20 event tokens per sequence.

Taxi (Whong, 2014). This dataset contains time-stamped taxi pickup and drop off events with zone location ids in New York city in 2013 . Following the processing recipe of previous work (Mei et al., 2019), each event type is defined as a tuple of (location, action). The location is one of the 5 boroughs {Manhattan, Brooklyn, Queens, The Bronx, Staten Island}. The action can be either pick-up or drop-off. Thus, there are $K = 5 \times 2 = 10$ event types in total. We work on a subset of 2000 sequences of taxi pickup events with average length 39 and then end up with $K = 10$ event types. We randomly sampled disjoint train, dev and test sets with 1400, 200 and 400 sequences from the dataset. Given the average inter-arrival time 0.22 (time unit is 1 hour), we choose the prediction horizon as 4.5 that approximately has 20 event tokens per sequence.

DATASET	K	# OF EVENT TOKENS			SEQUENCE LENGTH		
		TRAIN	DEV	TEST	MIN	MEAN	MAX
TAOBAO	17	75000	12000	30000	58	59	59
TAXI	10	56000	10000	16000	38	39	39
STACKOVERFLOW	22	91000	26000	27000	41	65	101

Table 1: Statistics of each dataset.

StackOverflow (Leskovec & Krevl, 2014). This dataset has two years of user awards on a question-answering website: each user received a sequence of badges and there are $K = 22$ different kinds of badges in total. We randomly sampled disjoint train, dev and test sets with $1400, 400$ and 400 sequences from the dataset. The time unit is 11 days; the average inter-arrival time is 0.95 and we set the prediction horizon to be 20 that approximately covers 20 event tokens.

Table 1 shows statistics about each dataset mentioned above.

B.2 Implementation Details

All models are implemented using the PyTorch framework (Paszke et al., 2017).

For the implementation of NHP, AttNHP, and thinning algorithm, we used the code from the public Github repository at `https://github.com/yangalan123/anhp-andtt` (Yang et al., 2022) with MIT License.

For DualTPP, we used the code from the public Github repository at `https://github.com/pratham16cse/DualTPP` (Deshpande et al., 2021) with no license specified.

For the optimal transport distance, we used the code from the public Github repository at `https://github.com/hongyuanmei/neural-hawkes-particle-smoothing` (Mei et al., 2019) with BSD 3-Clause License.

Our code can be found at `https://github.com/alipay/hypro_tpp` and `https://github.com/iLampard/hypro_tpp`.

B.3 Training and Testing Details

Training Generators. For AttNHP, the main hyperparameters to tune are the hidden dimension D of the neural network and the number of layers L of the attention structure. In practice, the optimal D for a model was usually 32 or 64; the optimal L was usually $1, 2, 3, 4$. In the experiment, we set $D = 32, L = 2$ for AttNHP and $D = 32, L = 4$ for AttNHP-LG. To train the parameters for a given generator, we performed early stopping based on log-likelihood on the held-out dev set.

Training Energy Function. The energy function is built on NHP or AttNHP with 3 MLP layers to project the hidden states into a scalar energy value. AttNHP is set to have the same structure as the base generator 'Att'. NHP is set to have $D = 36$ so that the joint model have the comparable number of parameters with other competitors. During training, each pair of training sample contains 1 positive sample and 5 negative samples ($N = 5$ in equation 3 and 4), generated from generators. Regarding the regularization term in equation 5, we choose $\beta = 1.0$.

All models are optimized using Adam (Kingma & Ba, 2015).

Testing. During testing, for efficiency, we generates 20 samples ($M = 20$ in Algorithm 2) per test prefix and select the one with the highest weight as the prediction. Increasing M could possibly improves the prediction performance.

Computation Cost. All the experiments were conducted on a server with 256G RAM, a 64 logical cores CPU (Intel(R) Xeon(R) Platinum 8163 CPU @ 2.50GHz) and one NVIDIA Tesla P100 GPU for acceleration. On all the datasets, the training time of HYPRO-A and HYPRO-N is 0.005 seconds per positive sequence.

For training, our batch size is 32. For Taobao and Taxi dataset, training the baseline NHP, NHP-lg, AttNHP, AttNHP-lg approximately takes 1 hour, 1.3 hour, 2 hours, and 3 hours, respectively (12, 16, 25, 38 milliseconds per sequence), training the continuous-time LSTM energy function and continuous-time Transformer energy function takes 20 minutes and 35 minutes (4 and 7 milliseconds per sequence pair) respectively.

MODEL	DESCRIPTION	VALUE USED		
		TAOBAO	TAXI	STACKOVERFLOW
DUALTPP	RNN HIDDEN SIZE	76	76	76
	TEMPORAL EMBEDDING SIZE	32	32	32
NHP	RNN HIDDEN SIZE	36	36	36
NHP-LG	RNN HIDDEN SIZE	52	52	52
ATTNHP	TEMPORAL EMBEDDING SIZE	64	64	64
	ENCODER/DECODER HIDDEN SIZE	32	32	32
	LAYERS NUMBER	2	2	2
ATTNHP-LG	TEMPORAL EMBEDDING SIZE	64	64	64
	ENCODER/DECODER HIDDEN SIZE	32	32	32
	LAYERS NUMBER	4	4	4
HYPRO-N-B	RNN HIDDEN SIZE IN NHP	32	32	32
HYPRO-N-M	RNN HIDDEN SIZE IN NHP	32	32	32
HYPRO-A-B	ENERGY FUNCTION IS A CLONE OF ATTNHP	NA	NA	NA
HYPRO-A-M	ENERGY FUNCTION IS A CLONE OF ATTNHP	NA	NA	NA

Table 2: Descriptions and values of hyperparameters used for models trained on the two datasets.

MODEL	# OF PARAMETERS		
	TAOBAO	TAXI	STACKOVERFLOW
DUALTPP	40.0K	40.1K	40.3K
NHP	19.6K	19.3K	20.0K
NHP-LG	40.0K	39.3K	40.6K
ATTNHP	19.7K	19.3K	20.1K
ATTNHP-LG	38.3K	37.9K	38.7K
HYPRO-A-B	40.0K	40.5K	41.0K
HYPRO-A-M	40.0K	40.5K	41.0K

Table 3: Total number of parameters for models trained on the three datasets.

For inference, inference with energy functions takes roughly 2 to 4 milliseconds. It takes 0.2 seconds to draw a sequence from the autoregressive base model. Our implementation can draw multiple sequences at a time in parallel: it takes only about 0.4 seconds to draw 20 sequences—only twice as drawing a single sequence. We have released this implementation.

B.4 More OTD Results

The optimal transport distance (OTD) depends on the hyperparameter C_{del}, which is the cost of deleting or adding an event token of any type. In our experiments, we used a range of values of $C_{\text{del}} \in \{0.05, 0.5, 1, 1.5, 2, 3, 4\}$, and report the averaged OTD in Figure 1.

In this section, we show the OTD for each specific C_{del} in Figure 7. As we can see, for all the values of C_{del}, our HYPRO method consistently outperforms the other methods.

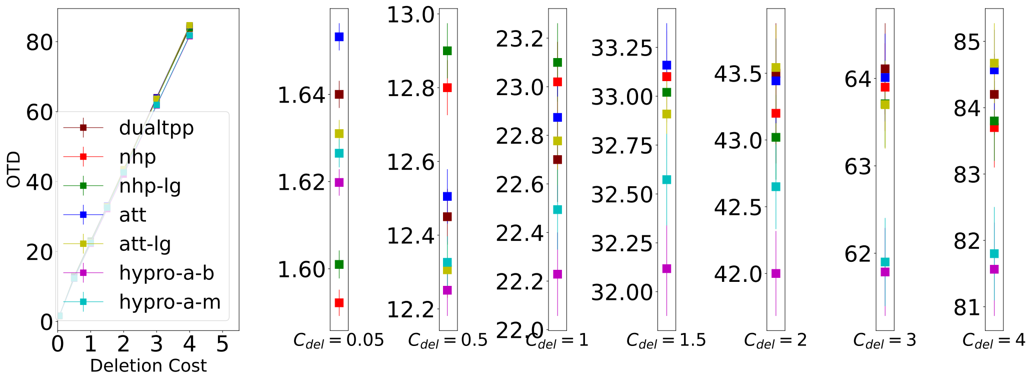
B.5 Analysis Details: Baseline That Ranks Sequences by the Base Model

To further verify the usefulness of the energy function in our model, we developed an extra baseline method that ranks the completed sequences based on their probabilities under the base model, from which the continuations were drawn. This baseline is similar to our proposed HYPRO framework but its scorer is the base model itself.

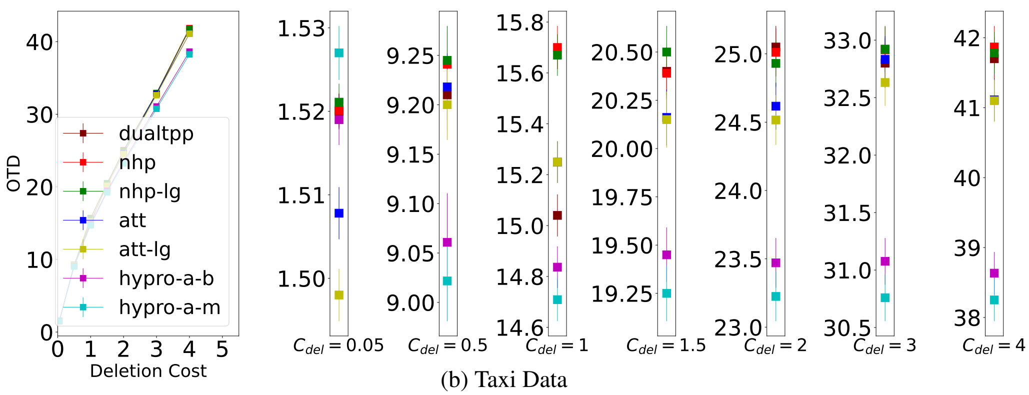
We evaluated this baseline on the Taobao dataset. The results are in Figure 8. As we can see, this new baseline method is no better than our method in terms of the OTD metric but much worse than all the other methods in terms of the RMSE metric.

B.6 Analysis Details: Statistical Significance

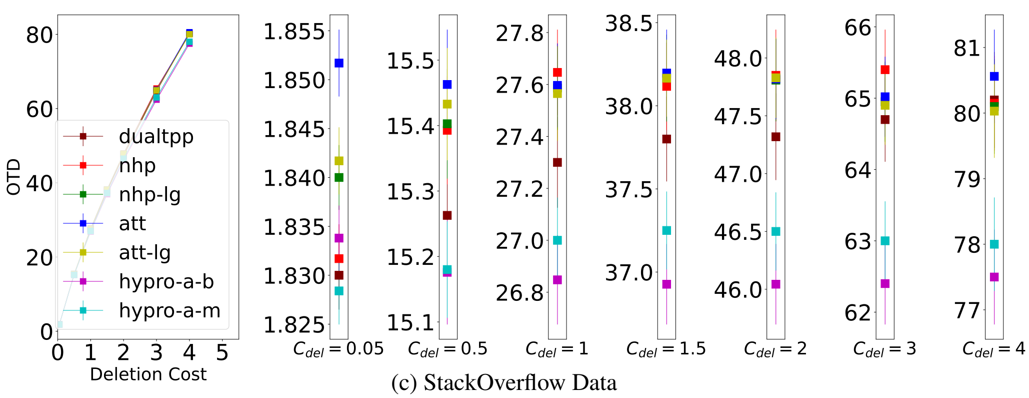
We performed the paired permutation test to validate the significance of our proposed regularization technique. Particularly, for each model variant (hypro-a-b or hypro-a-m), we split the test data into ten folds and collected the paired test results with and without the regularization technique for



(a) Taobao Data



(b) Taxi Data



(c) StackOverflow Data

Figure 7: OTD for each specific deletion/addition cost C_{del}.

each fold. Then we performed the test and computed the p-value following the recipe at `https://axon.cs.byu.edu/Dan/478/assignments/permutation_test.php`.

The results are in Figure 9. It turns out that the performance differences are strongly significant for hypro-a-b (p-value < 0.05) but not significant for hypro-a-m (p-value ≈ 0.1). This is consistent with the findings in Figure 2.

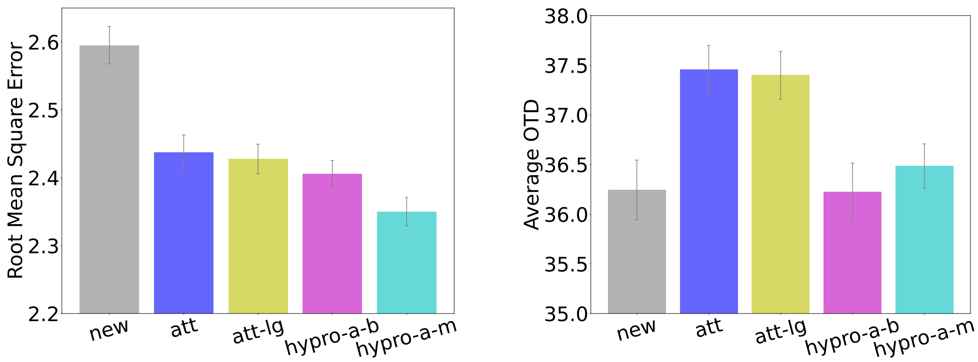
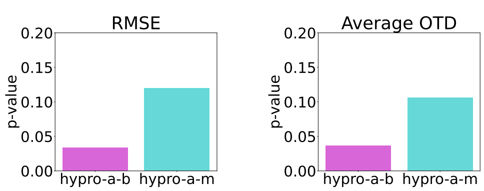
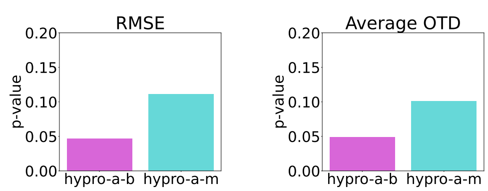


Figure 8: Evaluation of new baseline on Taobao dataset. The base model is AttNHP. The performances of the other methods are copied from Figure 1a.



(a) Taobao dataset.



(b) Taxi dataset.

Figure 9: Statistical significance of our regularization on the Taobao and Taxi datasets.