Challenges and opportunities in applying Neural Temporal Point Processes to large scale industry data

Dominykas Šeputis
dominykas.seputis@mif.stud.vu.lt
Vilnius University
Vilnius, Lithuania

Jevgenij Gamper
jevgenij.gamper@vinted.com
Vinted
Vilnius, Lithuania

Remigijus Paulavičius
remigijus.paulavicius@mif.vu.lt
Vilnius University
Vilnius, Lithuania

ABSTRACT

In this work, we identify open research opportunities in applying Neural Temporal Point Process (NTPP) models to industry scale customer behavior data by carefully reproducing NTPP models published up to date on known literature benchmarks as well as applying NTPP models to a novel, real world consumer behavior dataset that is twice as large as the largest publicly available NTPP benchmark. We identify the following challenges. First, NTPP models, albeit their generative nature, remain vulnerable to dataset imbalances and cannot forecast rare events. Second, NTPP models based on stochastic differential equations, despite their theoretical appeal and leading performance on literature benchmarks, do not scale easily to large industry-scale data. The former is in light of previously made observations on deep generative models. Additionally, to combat a cold-start problem, we explore a novel addition to NTPP models - a parametrization based on static user features.

KEYWORDS

generative modeling;neural temporal point process;user behavior modeling;

1 INTRODUCTION

Considering online platforms, most use cases of generative modeling applications are motivated by the need to capture user behavior. When in need to answer questions such as "How long do we need to wait for the next user's visit?" or "What is the optimal time of recommending winter coats?" prediction type tasks where the goal is to predict times and marks of the subsequent events are formed. With the rapid growth of online platforms, it is crucial to estimate how its users will behave in the near future.

Commonly referred to as Buy-till-You-Die models, Pareto/NBD [24], BG/NBD [9], and other generative models alike [15] might be too simplistic and lack flexibility in modeling heterogeneity of customer behavior. Nonetheless, the generative approach towards customer behavior modeling described by [8] might be useful, albeit in a different form.

For example, in Pareto/NBD model, the number of purchases that an i'th customer makes in their lifetime is described by a Poisson distribution parameterized by λ_i. Given the inferred parameters for Poisson and Exponential distributions representing a customer's purchase number and lifetime, one can estimate the probability of a customer being "alive." Buy-till-You-Die models assume a fixed parameter λ that does not depend on the time, and the models are static. A customer who had a positive experience with a service is more likely to come back and keep returning if the experience continues to be positive and vice versa - these models do not capture such behavior.

An alternative would be to use a self-exciting point process like Hawkes, which implies that λ parameter is a function of time, positively influenced by past events. Lately, a new class of temporal point processes (TPPs) has emerged - Neural Temporal Point Process (NTPP) [3, 20, 28]. The NTPPs are based on the self-exciting nature of the Hawkes process but are more versatile and can capture more diverse user behavior. In this work, we identify open research opportunities in applying neural temporal point process models to industry scale customer behavior data.

We explore and compare different approaches to NTPP: Neural Ordinary Differential Equations [3] and Attention [26] based temporal point processes [7, 28]. We apply these models to various synthetic and real-life datasets. To further test these models, we introduce online second-hand marketplace users behavior dataset. The dataset consists of 4,886,657 events and is 10 times larger than the commonly used Stack Overflow dataset [5, 7, 16, 19]. We attempt to use selected models to describe users' actions in the marketplace and explore the benefits of parameterizing Self-Attention based NTPP models with static features. Finally, we discuss explored models' scalability and their ability to capture rare events.

This paper is organized as follows. In section 2, we explore different ways temporal point processes are used in the industry. In section 3, we give a formal overview of traditional and neural temporal point processes. In section 4, we describe the data and our experimental approach. Section 5 reports the main results of the document, and the final section concludes.

2 RELATED WORK

With the rapid growth of online platforms, it is crucial to estimate how its users will behave in the near future. Multivariate point processes like FastPoint [25] prove that optimal results are achieved by combining neural networks with traditional generative temporal point process models. The approach scales to millions of correlated marks with superior predictive accuracy by fusing deep recurrent neural networks with Hawkes processes. Combining traditional temporal point process models with neural networks enables the industry to solve diverse problems: from predicting when customers will leave an online platform to simulating A/B tests.

Churn prediction. Customer churn prediction is one of the most crucial problems to solve in subscription and non-subscription-based online platforms. Traditionally churn prediction is handled as a supervised learning problem [21, 22, 27] where the goal is to predict if a user will be leaving the platform after a fixed time period. However, platform usage habits can differ from one user to another, and a decrease in an activity does not always is a sign of churn. [17] is one of the solutions that tackle churn prediction via generative modeling. The solution interprets user-generated events

in time scope of sessions and proposes using user return times and session durations to predict user churn.

Time-Sensitive Recommendations. When considering item recommendations for a user, we are thinking about the optimal suggestion and what time of the year we should suggest. [6] introduces a novel convex formulation of the problems by establishing an under-explored connection between self-exciting point processes and low-rank models. The proposed solution expands its applicability by offering time-sensitive recommendations and users' returning time predictions that can determine previously discussed churn or optimize marketing strategies. The item recommendation part is accomplished by calculating intensity $\lambda_{u,i}$ for each item i and user u. After, items are sorted by descending order of calculated intensity, and top-k items are returned.

Interactive Systems Simulation. It is common to see modern systems using numerous different machine learning models interacting with each other. As a result, user experience is often defined by various machine learning systems layered iteratively atop each other. The previously discussed item-recommendation problem is one example of such complex system [10]. When there is a need to change one or multiple models in the recommendation system, one might ask themselves what effect a particular change might have on the overall system performance. Usually, A/B testing answers these types of questions. Although when different system modules are changing rapidly and systems are getting more complex, it becomes more practical to simulate the effect than test it using traditional methods. [18] presents one of many ways to simulate interactive systems. The paper proposes Accordion, a fully trainable simulator for interactive systems based on inhomogeneous Poisson processes. While combining multiple intensity functions, the Accordion enables a comparison between realistic simulation settings and their effect on the total number of visits, positive interactions, impressions, and any other empirical quantity derived from a sampled dataset.

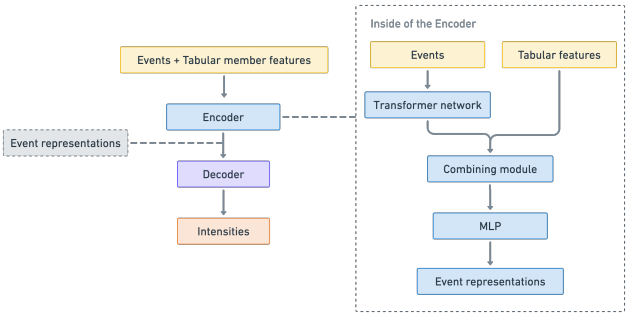


Figure 1: Architecture of parameterized Self Attention encoder.

3 METHODS

The base of our explored generative user behavior modeling methods is Poisson process - a continuous-time version of a Bernoulli process where event arrivals are happening at any point of the time [2]. The homogeneous Poisson process describes intensity of event arrival by $\lambda = \lim_{\delta \to 0} = \frac{P(1,\delta)}{\delta}$, where δ is a minimal interval between events time. If needed to model events that are influenced by the time dimension, we can use non-homogeneous version of the Poisson process. Simply instead of using a static intensity function λ to represent the intensity of the events, we transform it into a function of time $\lambda = \lambda(t)$ [23].

To capture more complex behavior, for example item recommendation, where one successful transaction mostly reinforces members to buy more items, self-exciting processes are used. The self-exciting process is a point process with a property that the arrival of the event causes the conditional intensity function to increase. One of the most known self-exciting processes is Hawkes process [12]. The key feature of the process is a conditional intensity function, influenced by the events in the past $\lambda(t|\mathcal{H}_t) = \lambda_0(t) + \sum_{i:t>T_j} \phi(t - T_j)$, where $\mathcal{H}_t$ is the history of the events that happened before time t. From the defined Hawkes function we can see that there is two major parts in it: base intensity function λ_0 and a summation of *kernel* $\phi(t - T_i)$. The base intensity function λ_0 keeps the independence property from the Poisson process, and past events do not influence it. The influence of the past arrivals in the Hawkes is defined with a kernel function $\phi(\cdot)$, which usually takes form of exponential function $\phi(x) = \alpha e^{-\beta x}$, where $\alpha \geq 0, \beta > 0$ and $\alpha < \beta$. The kernel function ensures that more recent events would have a greater influence on the current intensity than events in the past.

3.1 Neural Hawkes Process

While Hawkes process brings a significant improvement in capturing the self-exciting type of events, the process is not fully sufficient at representing a more complex type of events. Hawkes process's intensity function dictates additive and decaying influence of the past events. While this behavior can be true for some events, event streams like item recommendations do not benefit from the Hawkes process's additive nature. The more similar items are shown, the less influence they bring and even can discourage people from buying the product. Neural Hawkes Process (NHP) [20] is better suited to battle more complex TPP problems. The Neural Hawkes Process introduces methods that can define processes where a past event's influence: is not always additive, is not always positive, and does not always decay. The Neural Hawkes process brings improvements in two parts. First, NHP introduces non-linear *transfer function* $f : \mathbb{R} \to \mathbb{R}^+$, which usually takes form of non-linear *softplus* function $f(x) = \alpha \log(1 + e^{x/\alpha})$. Second, rather than predicting *Hawkes* $\lambda(t)$ as a simple summation, NHP uses recurrent neural network (RNN) [13] to determine the intensity function $\lambda = f(h(x))$, where h is a hidden state of the RNN. This change allows learning a complex dependence of the intensities on past events' number, order, and timing. This architecture lets to improve the base Hawkes process shortcomings: inability to represent events where past event's influence is not always additive is not always positive, and does not always decay; however, the neural Hawkes process has a downside - weaknesses of RNNs are inherited; namely the model's inability to capture long-term dependencies (also called as "forgetting") and is difficult to train [1].

3.2 Self-Attention Hawkes process

Despite not being directly applicable to model point process, recently, Transformer architecture was successfully generalized to continuous-time domain [28, 29]. Self-Attentive Hawkes Process (SHAP) is an self-attention based approach to a Hawkes process proposed by [28]. SHAP employs self-attention to measure the influence of historical events to the next event. The SHAP approach to the temporal point processing field also brings a time-shifted positional embedding method where time intervals act as phase shifts of sinusoidal functions. Transformers Hawkes Process (THP) presented by [29] is another self-attention based approach to the Hawkes process. THP improves recurrent neural network-based point process models that fail to capture short-term and long-term temporal dependencies [14]. The key ingredient of the proposed THP model is the self-attention module. Different from RNNs, the attention mechanism discards recurrent structures. However, the THP model still needs to be aware of the temporal information of inputs, such as timestamps. To achieve this temporal encoding procedure is proposed that is used to compute the positional embedding.

3.3 Neural Jump Stochastic Differential Equations

Sometimes user's interest is built up continuously over time but may also be interrupted by stochastic events. To simultaneously model these continuous and discrete dynamics, we can use the Neural Jump Stochastic Differential Equations (NJSDEs) [16]. The NJSDE uses latent vector $z(t)$ to encode the system's state, which continuously flows over time until an event at random, introducing an abrupt jump and changing its trajectory. The event conditional intensity function and the influence of the jump are parameterized with neural networks as a function of $z(t)$, while the continuous flow is described by Neural Ordinary Differential Equations [3]. The advantage of Neural NJSDEs is that they can be used to model a diverse selection of marked point processes, where a discrete value can compliment events (for example, a class of purchased item) or a vector of real-valued features (e.g., spatial locations).

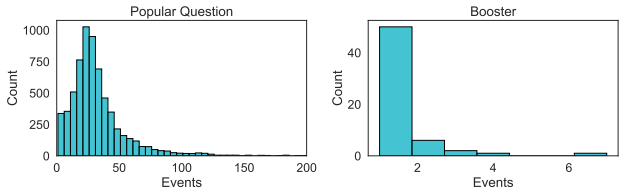
3.4 Parameterised Self-Attention Hawkes process

To combat a cold-start problem, we explore a novel addition to an encoder-decoder architecture based NTPP models - a parametrization based on static user features. Specifically, we experiment with self-attention type encoders. We concatenate outputs of the Transformer model m with processed static user features p and pass them thought multilayer perceptron (MLP): $f(X) = MLP(m||p)$. The change differs from the approach proposed by [7], where outputs of the Transformer model are directly passed to the MLP. We present the illustrated architecture of our approach in Figure 1.

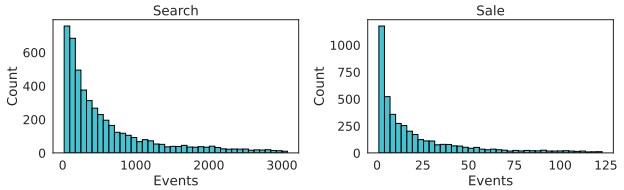
4 EXPERIMENTAL SETUP

4.1 Data

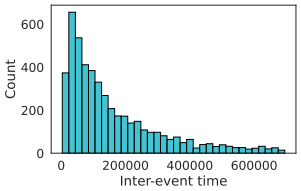
The first dataset we use is a simulated Hawkes process. The synthetic Hawkes data serves as a starting point while evaluating selected models' performance. As the data is generated, we can have unlimited samples. Also, as the intensity values at any point in time are known, we are able to compare them with values provided by the trained models. We designed the synthetic dataset consisting of two independent processes.



(a) Comparison between common (left) and rare (right) event type distribution within Stack Overflow platform.



(b) Comparison between common (left) and rare (right) event type distribution within Vinted platform.



(c) Vinted inter-event time distribution.

Figure 2: Different properties of Stack Overflow and Vinted datasets.

The second and the third datasets are real user behavior datasets. Stack Overflow is a question-answering website that awards users various badges based on their activity on the website. The website's users activity is commonly used when benchmarking various temporal point process models [5, 7, 16, 19]. Novel customer behavior dataset comes from a second hand marketplace Vinted, where users sell and buy various goods.

We collect Vinted users' activity data by sampling all of the recorded activity (from user's registration to the time of collecting the data) of 5,082 Vinted users. All of the selected users were active in the last year (2021/01/01 - 2022/01/01) and completed at least 40 actions. Inside the dataset, four different types of events can be found: Purchase, Listing, Search, and Sale. Frequent event occurrences and long recorded activity periods result in the dataset being ten times larger than Stack Overflow (4,886,667 vs. 480,414 events). While inspecting the event count distributions (see Figure 2), we notice that users' activities within Vinted platform are similar to Stack Overflow, where common events' distributions are shaped similarly. Events collected from Vinted platform as well as from the Stack Overflow website suffer from data imbalance. Events

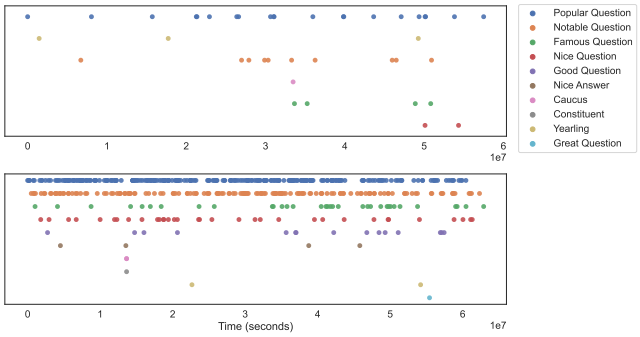


Figure 3: Comparison between passive (top) and active (bottom) Stack Overflow user's activity within the website. Passive website user tends to acquire less diverse badges' set than an active one.

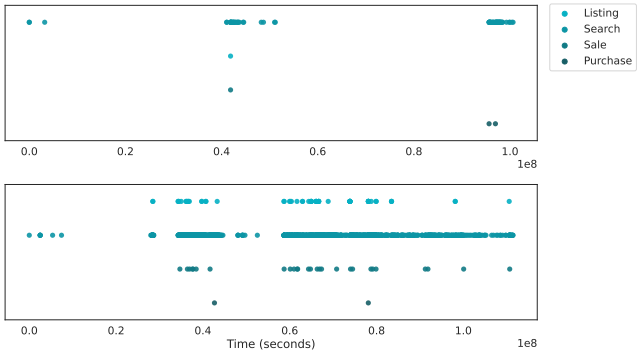


Figure 4: Comparison between passive (top) and active (bottom) Vinted user's activity within the platform. Active Vinted user completes same actions as the passive user, only in a more frequent manner.

like Publicist, Populist, or Booster are given to the website users only when he accomplishes unique action. These actions can be tricky to capture for temporal point process models, as the data imbalance is quite significant within the dataset. The most common events are Search and Listing. Purchases and Sales are happening much rarer - as they are influenced not only by users' behavior but also by external factors such as the listed item's appeal or Vinted's recommendation system's quality.

Where the differences between Vinted and Stack Overflow platforms begins is the measured inter-event times (see Figure 2). Vinted users are returning to the platform more frequently than Stack Overflow ones. This difference is caused by the fact that Stack Overflow browsing activity is not recorded. Activity tendencies between active and passive platform users are also different between mentioned platforms. Active Vinted user completes the same actions as the passive user, only in a more systematic manner (see Figure 4).

Besides user activity records, we collect static Vinted users' features. We use the features to explore a solution to the cold start problem in modeling user behavior. We parameterize temporal point process models in the hope of more accurate event type classification results.

We split each dataset into training, test and validation parts. Dataset statistics are summarized in Table 1.

4.2 Training

We base our model selection on the model's performance in capturing the type and time of the next event based on the time it occurred. Also, the accessibility of a source code is a significant deciding factor. We select [16] and [7] as the primary source of models used for our experimentation. [7] demonstrates that models based on Encoder-decoder architectures are effective not only for Natural Language Processing [4, 26] but also work well in a temporal point processing field. In the paper, the usage of Self-Attention as a building block of the model's architecture proves to be beneficial while modeling healthcare records. Where [16] showcases a Neural Jump Stochastic Differential Equations - model that captures temporal point processes with a piecewise-continuous latent trajectory. The model demonstrates its predictive capabilities on various synthetic and real-world marked point process datasets.

We conclude the final list of models from six different ones. Five of them are selected from [7]: **SA-COND-POISSON**, **SA-LNM**, **SA-MLP-MC**, **SA-RMTPP-POISSON**, **SA-SA-MC**. Where the beginning of the model's name "*SA-*" means the type of encoder (Self-Attention), and the rest of the name marks the type of encoder. The sixth model used in experimentation is Neural Jump Stochastic Differential Equations (**NJSDE**).

For all models, we use the hyperparameters specified in the original literature, with an exception of batch size and training epochs. Experiments on Vinted data use the same training configuration as experiments on Stack Overflow users' activity.

Furthermore, we explore different solutions to the cold start problem in modeling behavior of new Vinted users, we experiment with parameterizing self-attention based NTPPs. Similarly to [11], we join processed tabular user features to the output of Encoder's transformer network. We train all models on a workstation with a *Intel 32 core 3rd gen Xeon CPU and 120 GB memory*. The complete implementation of our algorithms and experiments are available at https://github.com/dqmis/ntpps.

4.3 Model evaluation

All trained models are trained and evaluated using a five-fold cross-validation strategy. As in [16], we use a weighted F1 score as one of the primary evaluation metrics. This allows us to verify model's capabilities of predicting the type of the next event given the time it occurred. As a secondary metric, we use classification accuracy, mainly to compare our results with [7]. Despite its popularity in quantifying the predictive performance of TPPs, we do not use MAE/MSE metric in this paper. As the metric compares actual intensity values with the model predicted ones, the lack of such data (only synthetic Hawkes dataset intensity values are known) is the main reason that determines our choice. Additionally, we assess each model's predictions via a classification report. As the data imbalance problem is a concern, the metric enables us to validate model's performance in predicting low-samples label.

Table 1: Properties of datasets used for experimentation.

					Size			
Dataset	# classes	Task type	# events	Avg. length	Train	Valid	Test	Batch
Hawkes	2	Multi-class	350,281	14	233,537	58,451	58,293	512
Stack Overflow	22	Multi-class	480,414	72	335,870	47,966	96,578	32
Vinted	4	Multi-class	4,886,657	962	3,428,380	482,512	975,765	4

5 RESULTS

We report our results on Hawkes and Stack Overflow data (table 2) and on Vinted data (table 3). First, we note that replication of the models' performance results provided by [16] and [7] are mostly successful. Notably, the results of SA-MLP-MC model trained on the Stack Overflow dataset are not similar to the ones presented in [16]. This could be caused by a mismatch in training environments or errors in our or [7] evaluation process. Notably, the best performing models were NJSDE and SA-COND-POISSON. Both presented great results while classifying synthetic Hawkes events and Stack Overflow users' actions. However, NJSDE model required much more resources and took longer to train to reach high accuracy (15 hours to train NJSDE model vs. 4 hours to train SA-COND-POISSON). NJSDE implementation proposed by [16] requires time series conversion into grid later used for modeling. This action consumes more memory as the sequence count in the batch grows. Because of this, we could not validate the NJSDE model's performance on Vinted data.

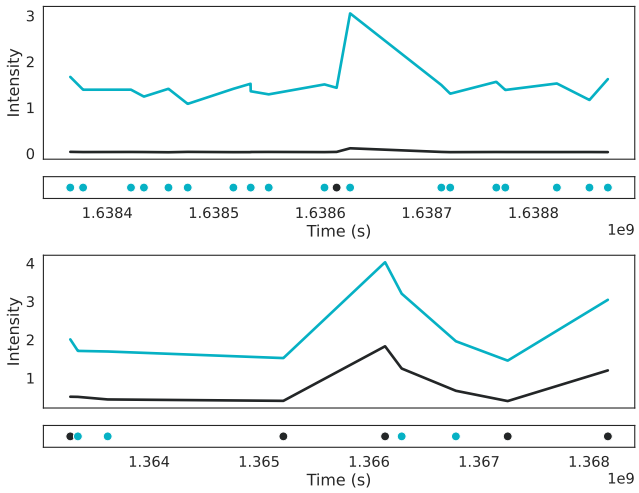


Figure 5: Intensity functions on several labels provided by SA-COND-POISSON model trained on Vinted (top) and Stack Overflow (bottom) datasets. Common event's intensity (marked green color) is always higher than rare event's one (marked black color). This results in the model omitting rare events while classifying the next event's type.

Notably, all models, including the best-performing ones, suffer from an inability to classify rare events. This can be seen while comparing the accuracy and F1 score metrics (also see classification report in Table 4). F1 score and accuracy used in the papers that inspired our research fail to report this phenomenon. This is caused by a significant data imbalance in the Stack Overflow and later discussed Vinted datasets. Also, some types of events could be hard to capture, not only based on their rarity but also their nature. For example, events inside Stack Overflow dataset like Yearling (awarded for being an active user for one year time period) or Caucus (awarded for visiting an election during any phase of an active election and having enough reputation for casting a vote) can be difficult to identify without a time dimension. Also, some of the badges like Promoter (given for the first bounty user offers on his question) are less based on the user's activity and more on external factors like the importance of the asked question.

Same tendencies can be seen while inspecting performance results of models trained on Vinted data. While the best performing SA-COND-POISSON model was able to reach an F1 score of 0.857, it still failed to predict some event types like Purchase and Sale (see Table 5). Interestingly Sale event had twice as many samples as the Listing but failed to be captured by the model. As with Stack Overflow badges, this can be explained by looking at the nature of the event. Searching for a new dress to buy or listing some pair of shoes are events caused mainly by users' habits. Purchases are motivated more by Vinted's search quality or recommendation system. The sale of the uploaded item is also less caused by its seller's past activity and more by the item's quality and appeal to the buyers. Inspecting differences between intensity values of the SA-COND-POISSON model trained on Vinted and Stack Overflow data (see Figure 5) gives a better understanding of the model's behavior. There is a significant difference between the intensity values of a frequent and less frequent event type. The difference tends to stay the same between all time points. This results in the model capturing overall intensity changes but lacking the ability to identify rare events. Furthermore, our experimentation with parameterizing encoder-decoder architecture-based models with user's static features did not produce valuable results (see Table 3). There is one case (SA-SA-MC) where parameterized model performs better than its counterpart, but this tendency is not identified while comparing the rest of the models. The lack of models' performance improvement is probably caused by the fact that selected user features do not directly impact user's behavior. However, to verify if the NTPPs parameterization with tabular features is beneficial, we need to conduct more detailed experiments with a more diverse set of features.

Table 2: Evaluation on Hawkes and Stack Overflow datasets. Best performances are boldened. Where comparable, results from [16] and [7] are displayed in *italic font style*.

	Hawkes		Stack Overflow	
Model	Accuracy	F1 score	Accuracy	F1 score
NJSDE	**.541**	**.552**	**.548** *(.527)*	**.363**
SA-COND-POISSON	**.538**	**.538**	.501	**.332** *(.326)*
SA-LNM	.537	.536	.369	.319 *(.305)*
SA-MLP-MC	.536	.531	.216	.194 *(.327)*
SA-RMTPP-POISSON	.526	.474	.515	.316 *(.288)*
SA-SA-MC	.507	.467	.382	.305 *(.324)*

Table 3: Evaluation on Vinted dataset. The scores are provided in two sections: where model was parameterised with user features / and where it was not. Best performances are boldened.

Model	Accuracy	F1 score
SA-LNM	.842 / .840	.771 / .768
SA-COND-POISSON	**.882** / **.882**	**.857** / **.856**
SA-SA-MC	.707 / .839	.657 / .767
SA-MLP-MC	.654 / .643	.676 / .670
SA-RMTPP-POISSON	.842 / .839	.771 / .767

6 CONCLUSION

We discussed broad industry application cases of NTPPs and their ability to capture online platform users' behavior. We used synthetic Hawkes dataset, Stack Overflow users' badges, and second-hand marketplace Vinted user activity as our experimentation datasets. Furthermore, we identify that Self-Attention and Neural Ordinary Differential Equations based NTPP models fail to capture the time of the next rare event but succeed in identifying overall user activity intensity. While the NJSDE model is optimal for identifying the next event's type and time, we note that it is not scalable to extensive industry data. Our experimentation with parametrizing Self-Attention-based NTPPs did not show any significant improvements. However, we identify that we need to conduct more experiments to verify if the method is beneficial.

6.1 Broader Impact

While the explored NTPPs are not suitable for predicting the time of the next rare event, it does not mean they are not beneficial for subscription and non-subscription-based online platforms. These models can be valuable when solving problems such as churn prediction. By grouping individual events into user sessions, we would be able to detect when a general user's intensity is decreasing, thus letting us know when to act. Furthermore, we can optimize recommendation systems or marketing strategies by knowing the time of the next user's arrival on the online platform.

REFERENCES

[1] Y. Bengio, P. Simard, and P. Frasconi. 1994. Learning long-term dependencies with gradient descent is difficult. *IEEE Transactions on Neural Networks* 5, 2 (March 1994), 157–166. https://doi.org/10.1109/72.279181

[2] Dimitri P Bertsekas and John N Tsitsiklis. 2000. *Introduction to probability*. Athena Scientinis.

[3] Ricky T. Q. Chen, Yulia Rubanova, Jesse Bettencourt, and David Duvenaud. 2018. Neural Ordinary Differential Equations. arXiv:arXiv:1806.07366

[4] Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation. arXiv:arXiv:1406.1078

[5] Nan Du, Hanjun Dai, Rakshit Trivedi, Utkarsh Upadhyay, Manuel Gomez-Rodriguez, and Le Song. 2016. Recurrent Marked Temporal Point Processes: Embedding Event History to Vector. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (San Francisco, California, USA) *(KDD '16)*. Association for Computing Machinery, New York, NY, USA, 1555–1564. https://doi.org/10.1145/2939672.2939875

[6] Nan Du, Yichen Wang, Niao He, Jimeng Sun, and Le Song. 2015. Time-Sensitive Recommendation From Recurrent User Activities. In *Advances in Neural Information Processing Systems*, C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett (Eds.), Vol. 28. Curran Associates, Inc. https://proceedings.neurips.cc/paper/2015/file/136f951362dab62e64eb8e841183c2a9-Paper.pdf

[7] Joseph Enguehard, Dan Busbridge, Adam Bozson, Claire Woodcock, and Nils Y. Hammerla. 2020. Neural Temporal Point Processes For Modelling Electronic Health Records. arXiv:arXiv:2007.13794

[8] Peter S. Fader and Bruce G.S. Hardie. 2009. Probability Models for Customer-Base Analysis. *Journal of Interactive Marketing* 23, 1 (2009), 61–69. https://doi.org/10.1016/j.intmar.2008.11.003 Anniversary Issue.

[9] Peter S. Fader, Bruce G. S. Hardie, and Ka Lok Lee. 2005. "Counting Your Customers" the Easy Way: An Alternative to the Pareto/NBD Model. *Marketing Science* 24, 2 (may 2005), 275–284.

[10] Carlos A. Gomez-Uribe and Neil Hunt. 2016. The Netflix Recommender System: Algorithms, Business Value, and Innovation. *ACM Trans. Manage. Inf. Syst.* 6, 4, Article 13 (dec 2016), 19 pages. https://doi.org/10.1145/2843948

[11] Ken Gu and Akshay Budhkar. 2021. A Package for Learning on Tabular and Text Data with Transformers. In *Proceedings of the Third Workshop on Multimodal Artificial Intelligence*. Association for Computational Linguistics, Mexico City, Mexico, 69–73. https://doi.org/10.18653/v1/2021.maiworkshop-1.10

[12] ALAN G. HAWKES. 1971. Spectra of some self-exciting and mutually exciting point processes. *Biometrika* 58, 1 (04 1971), 83–90. https://doi.org/10.1093/biomet/58.1.83 arXiv:https://academic.oup.com/biomet/article-pdf/58/1/83/602628/58-1-83.pdf

[13] J J Hopfield. 1982. Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the National Academy of Sciences* 79, 8 (1982), 2554–2558. https://doi.org/10.1073/pnas.79.8.2554 arXiv:https://www.pnas.org/doi/pdf/10.1073/pnas.79.8.2554

[14] Fakultit Informatik, Y. Bengio, Paolo Frasconi, and Jfirgen Schmidhuber. 2003. Gradient Flow in Recurrent Nets: the Difficulty of Learning Long-Term Dependencies. *A Field Guide to Dynamical Recurrent Neural Networks* (03 2003).

[15] Pavel Jasek, Lenka Vrana, Lucie Sperkova, Zdenek Smutny, and Marek Kobulsky. 2019. Comparative analysis of selected probabilistic customer lifetime value models in online shopping. *Journal of Business Economics and Management* 20, 3 (April 2019), 398–423. https://doi.org/10.3846/jbem.2019.9597

[16] Junteng Jia and Austin R. Benson. 2019. Neural Jump Stochastic Differential Equations. arXiv:arXiv:1905.10403

[17] Ali Khodadadi, Seyed Abbas Hosseini, Ehsan Pajouheshgar, Farnam Mansouri, and Hamid R. Rabiee. 2019. ChOracle: A Unified Statistical Framework for Churn Prediction. arXiv:arXiv:1909.06868

[18] James McInerney, Ehtsham Elahi, Justin Basilico, Yves Raimond, and Tony Jebara. 2021. *Accordion: A Trainable Simulator for Long-Term Interactive Systems*. Association for Computing Machinery, New York, NY, USA, 102–113. https://doi.org/10.1145/3460231.3474259

[19] Hongyuan Mei and Jason Eisner. 2017. The Neural Hawkes Process: A Neurally Self-Modulating Multivariate Point Process. In *Proceedings of the 31st International Conference on Neural Information Processing Systems* (Long Beach, California, USA) *(NIPS'17)*. Curran Associates Inc., Red Hook, NY, USA, 6757–6767.

[20] Hongyuan Mei and Jason M Eisner. 2017. The Neural Hawkes Process: A Neurally Self-Modulating Multivariate Point Process. In *Advances in Neural Information Processing Systems*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (Eds.), Vol. 30. Curran Associates, Inc. https://proceedings.neurips.cc/paper/2017/file/6463c88460bd63bbe256e495c63aa40b-Paper.pdf

[21] C. Gary Mena, Arno De Caigny, Kristof Coussement, Koen W. De Bock, and Stefan Lessmann. 2019. Churn Prediction with Sequential Data and Deep Neural Networks. A Comparative Analysis. arXiv:arXiv:1909.11114

[22] Florian Merchie and Damien Ernst. 2022. Churn prediction in online gambling. arXiv:arXiv:2201.02463

[23] H. Pishro-Nik. 2014. *Introduction to probability, statistics, and random processes. Available at https://www.probabilitycourse.com.* Kappa Research LLC.

[24] David C. Schmittlein, Donald G. Morrison, and Richard Colombo. 1987. Counting Your Customers: Who-Are They and What Will They Do Next? *Manage. Sci.* 33, 1 (jan 1987), 1–24.

[25] Ali Caner Türkmen, Yuyang Wang, and Alexander J. Smola. 2020. FastPoint: Scalable Deep Point Processes. In *Machine Learning and Knowledge Discovery in Databases*, Ulf Brefeld, Elisa Fromont, Andreas Hotho, Arno Knobbe, Marloes Maathuis, and Céline Robardet (Eds.). Springer International Publishing, Cham, 465–480.

[26] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention Is All You Need. arXiv:arXiv:1706.03762

[27] Artit Wangperawong, Cyrille Brun, Olav Laudy, and Rujikorn Pavasuthipaisit. 2016. Churn analysis using deep convolutional neural networks and autoencoders. arXiv:arXiv:1604.05377

[28] Qiang Zhang, Aldo Lipani, Omer Kirnap, and Emine Yilmaz. 2019. Self-Attentive Hawkes Processes. arXiv:arXiv:1907.07561

[29] Simiao Zuo, Haoming Jiang, Zichong Li, Tuo Zhao, and Hongyuan Zha. 2020. Transformer Hawkes Process. arXiv:arXiv:2002.09291

A CLASSIFICATION REPORTS

Table 4: Classification report of NJSDE model predictions on Stack Overflow dataset.

Event type	Precision	Recall	F1 Score	# samples
Nice Question	0	0	0	6,016
Good Answer	0	0	0	1.939
Guru	0	0	0	653
Popular Question	.471	.975	.634	42,599
Famous Question	0	0	0	5,544
Nice Answer	.317	.511	.392	6,032
Good Question	0	0	0	1,846
Caucus	0	0	0	1,788
Notable Question	0	0	0	21,956
Necromancer	0	0	0	1,308
Promoter	0	0	0	207
Yearling	0	0	0	2,400
Revival	0	0	0	832
Enlightened	0	0	0	1,859
Great Answer	0	0	0	251
Populist	0	0	0	99
Great Question	0	0	0	269
Constituent	0	0	0	526
Announcer	0	0	0	364
Stellar Question	0	0	0	65
Booster	0	0	0	17
Publicist	0	0	0	8

Table 5: Classification report of SA-COND-POISSON model predictions on Vinted dataset.

Event type	Precision	Recall	F1 score	# samples
Listing	.685	.654	.669	102,297
Purchase	0	0	0	23,642
Sale	0	0	0	293,68
Search	.904	.968	.935	820,458

B HAWKES DATASETS

The parameters of our Hawkes datasets are:

$$\mu = \begin{bmatrix} 0.1 \\ 0.05 \end{bmatrix}, \alpha = \begin{bmatrix} 0.2 & 0.0 \\ 0.0 & 0.4 \end{bmatrix}, \beta = \begin{bmatrix} 1.0 & 1.0 \\ 1.0 & 1.0 \end{bmatrix}$$