Combining RNN with Transformer for Modeling Multi-Leg Trips

Yoshihiro Sakatani
sakatani.yoshihiro@sdtech.co.jp
sdtech Inc.
Minato, Tokyo, Japan

ABSTRACT

Recommending destinations of trips based on user behavior is an important task for travel agencies such as Booking.com. The Booking.com Challenge - WebTour 2021 ACM WSDM workshop is aimed at building models for this task; the goal is to predict the final destination of multi-destination trips, based on a large dataset of over a million-trip reservations at Booking.com with date, destination, etc. In this paper, I present my approach where I leverage recent advances in language modeling techniques including Transformer. The approach, which used only a sequence of the visited cities for input, showed a top-4 accuracy of 0.4720 on the final result leaderboard. Full code is available here: https://github.com/sakatani/BookingcomChallenge2021.

CCS CONCEPTS

• **Information systems** → *Recommender systems*; • **Computing methodologies** → **Neural networks**.

KEYWORDS

Recommender Systems, Recurrent Neural Network, Transformer, Sequence-Aware Recommendation

1 INTRODUCTION

Booking.com is the world's largest online travel agency, which is being used by millions of users to find accommodations. Recommending destinations based on user behavior is an important task for travel agencies such as Booking.com [1][6]. Within the domain of accommodation, where people experience much time and pay much money for it, the accuracy of recommenders is highly important.

Booking.com Challenge [5] was aimed at building a recommender system that could perform the task of estimating the final destination city of multi-destination trips based on the visited cities (i.e. cities where the user stayed before the final destination) and as well as additional contextual information such as reservation date. The challenge was based on over a million anonymized real-world reservations made on Booking.com in recent years. Thus, there were tens of thousands of possible final destination cities. The training data with complete itineraries and the test data with hidden final destinations were released for the development of models. The evaluation dataset was not partitioned into public and private leaderboards; two submissions of predictions for a single evaluation dataset were allowed, one for the intermediate leaderboard and one for the final leaderboard, and the final results were used for the evaluation.

Every record in the data was a user's reservation and contained information such as city of stay, country of stay, reservation ID, user ID, reservation date, check-in date, check-out date, affiliation channel, and country where the reservation was made. That information other than the city of stay was expected to play an important role in the performance of inference, as shown in a previous study [8]. Due to time constraints, however, my approach did not make use of this kind of contextual information. The approach, which focused on integrating several recent natural language processing techniques for modeling sentences and used only a sequence of the visited cities for input, showed a top-4 accuracy of 0.4720 in the final leaderboard results.

2 APPROACH

2.1 Data Splitting

In this challenge, as the evaluation data consisted of travel reservations of four or more legs only, the provided training data was grouped by reservation ID and then only the records for four or more legs were extracted. In addition to the extracted training data, the evaluation data was also used for building models; the evaluation data was grouped by reservation IDs in the same way for the training data, and the records with four or more legs before the final destination were extracted. Finally, the collected data was randomly split into 15% for the local evaluation, 15% for the validation during training, and 70% for the training.

2.2 Models

The model used in my approach is illustrated in Figure 1, where the positional encoding of a Transformer is replaced with a single-layer LSTM.

For modeling multi-destination trips, an RNN-based model has been proposed [8] that predicts the next destination of a sequence of destinations in a similar way that RNN-based language models predict the next word in a sentence. Recently, representative models in the field of natural language processing have been dominated by Transformer[10]-based models, as seen in the examples of Google BERT [4] and OpenAI GPT-3 [2]. Hence, the Transformer architecture was adopted as the basis for my approach.

Although Transformer-based models have achieved a number of state-of-the-arts techniques in the field of natural language processing, they have one computational cost weakness against conventional RNN models when considering its use in the generative task for destination recommendation. As the Transformer does not preserve the hidden state unlike RNNs but uses positional encoding to represent the sequential nature of tokens, it is expected to be computationally more expensive than RNNs because it needs to recompute the entire history in the context window at each time step.

To address this problem, a model called LSTM + Transformer, in which the positional encoding of Transformer is replaced with a

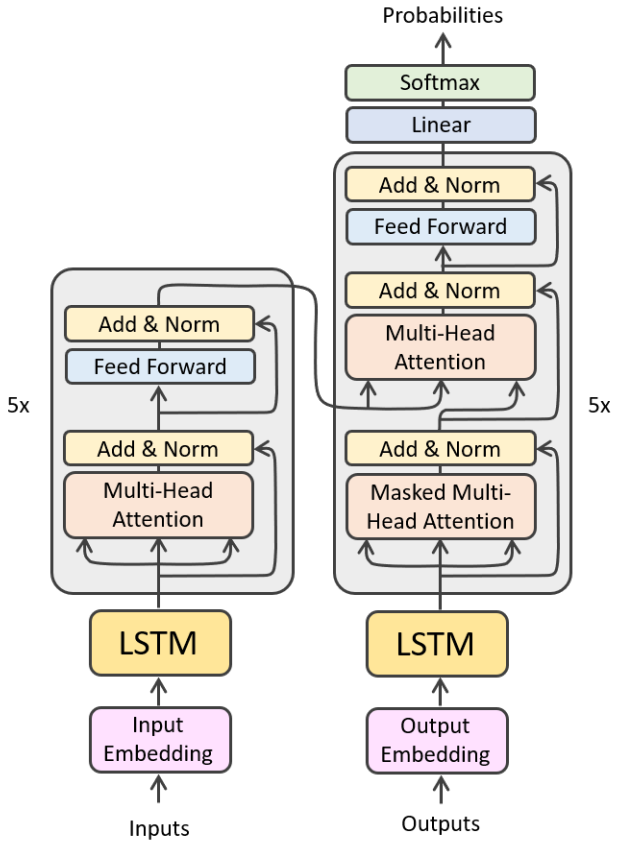

Figure 1: Model Architechture. LSTM-Transformer combined model is used for the prediction of the final destinations. This uses a single-LSTM layer to represent the sequential nature of tokens instead of the positional encoding.

single-layer LSTM, has been proposed in a previous study [9]. Transformer with positional encoding needs to recompute all the tokens in a context window at each timestep as the window slides, while the LSTM + Transformer model only needs to compute for a new token at each timestep since the LSTM keeps the hidden state. The study reported that this model achieved 86% of the computational cost of a Transformer-based model in the sentence generation task with the WikiText-103 dataset [7]. Although this LSTM-Transformer combined model is similar to the Cascaded Encoder model [3], it uses RNNs for decoding as well as for encoding.

3 EXPERIMENTS

All the experiments were conducted on Google Colaboratory and it was made sure that Intel(R) Xeon(R) CPU @ 2.30GHz, 250GB RAM, and NVIDIA Tesla T4 GPUs were allocated.

The LSTM-Transformer combined model consisted of one LSTM layer and five Transformer layers, with 512 dimensions of the LSTM hidden layer and Transformer, 1024 dimensions of the feedforward Transformer layers, and eight attention heads. The batch size was set to 16. The Adam optimizer was used during training. The learning rate was linearly warmed up to 7×10^{-4} with 4000 iterations, and then decayed in proportion to the inverse square root of the iteration number according to the formula in [10].

In addition to the LSTM-Transformer combined model, a model consisting of six layers of Transformers and a model consisted of LSTM or GRU were also prepared for comparison. The six-layer Transformer model was identical to the LSTM-Transformer model, except that a Transformer layer was assigned instead of the LSTM layer. The LSTM and GRU models have two layers and 512 hidden units. The GRU model was based on the previous study on the RNN-based multi-destination trip model [8], but the model details, including hyperparameters, were unable to be reproduced exactly due to the lack of sufficient information.

The models were evaluated using perplexity, which was based on the negative log-likelihood loss, and top-4 accuracy on the local evaluation data according to the rules of the challenge. Figure 2 shows the perplexity during training on the training and validation datasets. The perplexity of each model on the training dataset decreased steadily with the number of iterations. On the validation dataset, both the perplexity of Transformer and LSTM-Transformer combined model decreased steadily until at least 350,000 iterations, while that of the GRU and LSTM model decreased shakingly and then increased after 50,000 iterations.

The perplexity and top-4 accuracy on the local evaluation dataset were then evaluated using the checkpoint with the minimum perplexity on the validation dataset (Table 2). The LSTM-Transformer combined model showed the lowest perplexity and highest top-4 accuracy. The Transformer model performed as well as the LSTM-Transformer combined model. These results indicate that the replacement of the positional encoding of Transformer with the LSTM layer does not adversely affect the accuracy for this type of task, but rather it can have a positive effect on performance. The LSTM and GRU models showed smaller perplexities than the Transformer-based models during training on both the training and validation datasets, whereas they showed worse perplexities and top-4 accuracy on the evaluation data than the Transformer-based models. A possible explanation for this is that the RNN-based models might have been overfitting to the training or validation dataset.

Finally, the local evaluation dataset was split into 18% for validation and 82% for training, then the LSTM-Transformer model was re-trained with the datasets. The predictions submitted to the final result leaderboard were generated by the retrained model and showed 0.4720 for the top-4 accuracy score.

While this is irrelevant for the evaluation of the challenge, the time required for each model to generate tokens for 10000 steps was measured three times on both GPU and CPU (Table 2). The processing time of the LSTM-Transformer combined model was reduced to 59% of that of the Transformer model on the CPU and 90% on the GPU. These results indicate that replacing positional encoding of Transformer with LSTM reduces the computational cost. The processing time on GPU, however, was not reduced as much as on CPU. The LSTM-Transformer model may have some processing bottlenecks on GPU, as pointed out in a previous study[9].

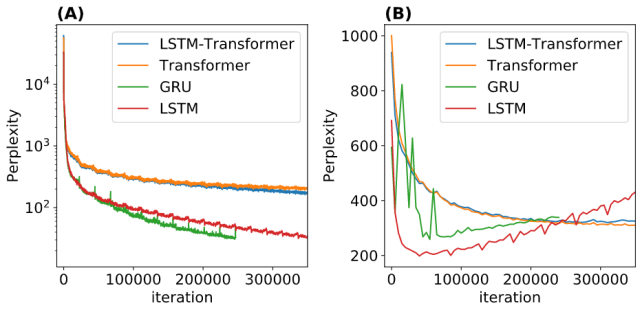

Figure 2: Perplexities of each model on the training and validation dataset. A). Training perplexity. B) Validation perplexity.

Table 1: Model performance on the local evaluation dataset.

Model	Loss	Perplexity	Top-4 accuracy
LSTM-Transformer	5.69	269.85	0.451
Transformer	5.62	276.98	0.440
GRU	5.80	328.74	0.397
LSTM	5.92	376.98	0.365

4 CONCLUSION

In this paper, I describe my approach to the Booking.com Challenge WebTour 2021 ACM WSDM workshop. The approach showed the LSTM substitution of positional encoding can have a positive effect on both the prediction and computational performance.

Because of time constraints, the approach did not make use of any information other than the visited destinations for building the models. Although the LSTM-Transformer model used only sequences of the visited cities for input, it achieved the 13th score on the final result leaderboard. A previous study has shown that the prediction performance of RNN models can be improved by combining contextual information such as users' home country with a sequence of visited destinations [8]. Thus, it is expected that the model used in this paper would improve in the prediction performance by considering such contextual information. A further study of how to combine the contextual information with visited destinations for Transformer-based models including the LSTM-Transformer combined model should be conducted.

The effect of replacing the positional encoding with LSTM on the computational cost was briefly investigated, but not fully explored in this study. It is expected that the LSTM-Transformer model reduces the computational cost especially when multiple destinations are predicted as the model has the advantage that the expansion of the context window does not affect the computational cost for the input. Another future work involves an investigation on how replacing positional encoding with LSTM affects the computational cost of Transformer-based models in generative tasks for destination recommendation.

Table 2: The processing time to generating tokens for 10,000 steps.

Model	CPU	GPU
LSTM-Transformer	192.5 (2.0)	56.32 (0.77)
Transformer	324.5 (3.8)	62.47 (0.94)
GRU	200.2 (1.6)	20.53 (0.18)
LSTM	237.3 (0.5)	21.55 (0.17)

REFERENCES

[1] Lucas Bernardi, Themistoklis Mavridis, and Pablo Estevez. 2019. 150 successful machine learning models: 6 lessons learned at booking. com. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 1743–1751.

[2] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language Models are Few-Shot Learners. arXiv:2005.14165 [cs.CL]

[3] Mia Xu Chen, Orhan Firat, Ankur Bapna, Melvin Johnson, Wolfgang Macherey, George Foster, Llion Jones, Niki Parmar, Mike Schuster, Zhifeng Chen, Yonghui Wu, and Macduff Hughes. 2018. The Best of Both Worlds: Combining Recent Advances in Neural Machine Translation. arXiv:1804.09849 [cs.CL]

[4] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. arXiv:1810.04805 [cs.CL]

[5] Dmitri Goldenberg, Kostia Kofman, Pavel Levin, Sarai Mizrachi, Maayan Kafry, and Guy Nadav. 2021. Booking.com WSDM WebTour 2021 Challenge. In *ACM WSDM Workshop on Web Tourism (WSDM WebTour'21)*. https://www.bookingchallenge.com/

[6] Julia Kiseleva, Melanie JI Mueller, Lucas Bernardi, Chad Davis, Ivan Kovacek, Mats Stafseng Einarsen, Jaap Kamps, Alexander Tuzhilin, and Djoerd Hiemstra. 2015. Where to go on your next trip? Optimizing travel destinations based on user preferences. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 1097–1100.

[7] Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. 2016. Pointer Sentinel Mixture Models. arXiv:1609.07843 [cs.CL]

[8] Sarai Mizrachi and Pavel Levin. 2019. Combining Context Features in Sequence-Aware Recommender Systems. In *RecSys (Late-Breaking Results)*. 11–15.

[9] Akihiro Tanikawa. 2020. [Deep Learning Study] Text Generation with LSTM + Transformer Model (Japanese). https://note.com/diatonic_codes/n/nab29c78bbf2e

[10] 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:1706.03762 [cs.CL]