24th European Conference on Artificial Intelligence - ECAI 2020
Santiago de Compostela, Spain

Capturing Attraction Distribution: Sequential Attentive Network for Dwell Time Prediction

Tianxin Wang[1,2] and **Jingwu Chen**[1,2] and **Fuzhen Zhuang**[1,2,*] and **Leyu Lin**[3] and **Feng Xia**[3] and **Lihuan Du**[3] and **Qing He**[1,2]

Abstract. In article recommendation, the dwell time is an important metric to measure user engagement on content and has been widely used as a proxy for user satisfaction. Therefore, predicting the dwell time is very helpful for making better recommendations and improving user experience. Modeling the interaction between user and content is the key for dwell time prediction. However, conventional methods usually model the content with document-level representation in a non-personalized way, which ignores the natural reading process of the reader and the reader attraction in sub-document level, this might lead to a bias for analyzing the user reading behavior. Since the attraction level of different parts is different for the user, the user attention changes dynamically while reading. The former content affects the reading for the latter content via the change of attraction level. Therefore, considering the attraction level of each part of the article, i.e., attraction distribution, is quite necessary for content modeling. In this paper, we propose the **S**equential **A**ttentive **N**etwork (SAN) for dwell time prediction, which effectively models the attraction distribution of the article reading process. We collect the data from WeChat, a widely-used mobile app in China, for experiments. The results demonstrate the advantages of our model over several competitive baselines on dwell time prediction.

1 Introduction

Recommender system (RS) is an important information filtering tool for guiding users in a personalized way of discovering products or products that they might be interested in from a large space of possible options [26]. Article recommendation is one of such ocassions that the users are facing very large space of possible options, the user experience thus heavily relies on the recommender system. Many recommender systems focus on optimizing the click through rate (CTR) or predicting the ratings to give better recommendation. These two metrics are quite straightforward, but have some drawbacks. CTR fails to capture the real satisfaction of user in the post-click, while the ratings are usually extremely sparse and rare in the system. In recent years, many studies have shown that dwell time is an important metric to measure user engagement with the content and should be used as a proxy to user satisfaction [1, 18]. Therefore, predicting the dwell time is helpful for ranking the article list more reasonably and improving the user experience in article recommendation service.

Since the dwell time is the result of the interaction between user and content, modeling the content is quite important. However, traditional methods usually model the content with document-level representation in a non-personalized way, which ignores the natural reading process of the reader and the user attraction in sub-document level. When visiting the article recommendation service, users are generally presented with a stream of titles, then they get interested in some and click in for reading. The reading process usually follows a sequential pattern, from the beginning to the end, and the reader can quit reading at any time during the reading process. Since the attraction level of different parts is different for the user, the user attention changes dynamically while reading. We denote the dynamic attention of user over different sentences as the attraction distribution.

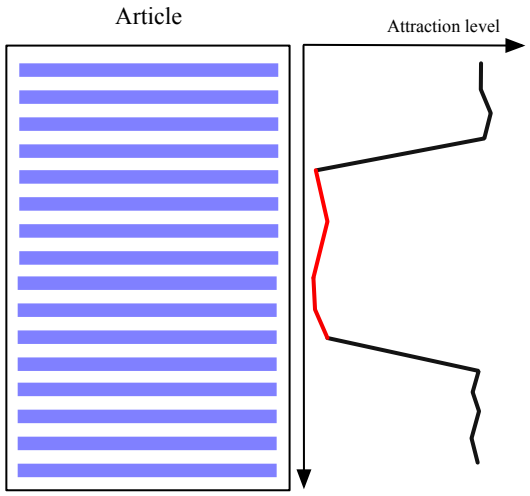

Figure 1. The attraction level changes over different sentences.

We present a typical example in Fig.1 to better illustrate the importance of attraction distribution, where the left part shows the sentences in the article and the right part indicates the attraction distribution for a specific user. As shown in the attraction distribution, the middle part of the article attracts the user at a very low level, thus the user is very likely to quit the reading process or the user may read this part very quickly. If the reader quits at the middle part, then the following content actually contributes nothing for the dwell time.

In addition to the example above, the special peculiarity in the article recommendation service makes the content modeling quite dif-

[1] Key Lab of Intelligent Information Processing of Chinese Academy of Sciences (CAS), Institute of Computing Technology, CAS, Beijing 100190, China, email: eteyogi688@gmail.com, {chenjingwu, zhuangfuzhen, heqing}@ict.ac.cn , ∗corresponding author: Fuzhen Zhuang.
[2] University of Chinese Academy of Sciences, Beijing 100049, China.
[3] WeChat Group, Tencent, China, email: {goshawklin, xiafengxia, lihuandu}@tencent.com.

ferent from other tasks, e.g., document classification. In the dwell time prediction, the probability that the reader has already quit reading, as well as the changing reading speed due to the attraction level, needs to be taken into consideration. On the other hand, the former content also affects the reading for the latter content via the change of attraction level.

To better model the reading process, we take the attraction distribution into consideration and model it in sub-document level. We believe that the attraction level of a sentence to a reader is closely related to the relevance between the interests of the user and the semantic information of this sentence. Since users are unaware of the particular content of any article, the click action is mainly driven by the information provided by the title. Therefore, we believe that the relevance between the title and the sentence also influences the attraction level of this sentence. Based on the above findings, we model the attraction of a particular sentence by modeling the three-side interaction among this sentence, user interests and title.

In this paper, we propose Sequential Attentive Network (SAN), a novel framework to model article reading process, in which we captured attraction distribution effectively. We collect the data from the article recommendation service on a widely-used mobile app in China for experiments. The results demonstrate the advantages of our model over several competitive baselines on dwell time prediction.

The main contributions of this work are summarized as follows.

- To the best of our knowledge, this is the first work to model the sequential reading process at sub-document level in article recommendation.
- We propose a novel efficient framework for dwell time prediction.
- We perform experiments on real-world data to demonstrate the advantages of SAN over several competitive baselines on dwell time prediction.

The rest of the paper is organized as follows: Section 2 reviews related work. Section 3 introduces the proposed approach in detail. Section 4 evaluates the proposed models on two datasets and Section 5 concludes the paper.

2 Related Works

2.1 Dwell Time Analysis

Dwell time has been widely modeled for helping us better understand user satisfaction and behavior in many domains in addition to article recommendation. Click followed by a long dwell time has generally been seen as satisfied click and been successfully used in web search [5]. Specifically, [17] proposed a time-aware model for evaluating the probability of user satisfaction, the model incorporates both click dwell time information and click sequence information. In addition to search result interactions, [25] utilized dwell time of items to pseudo ratings for prediction of user voting actions in social media analysis. [16] predicted the dwell time distribution based on page-level features to understand browsing behavior. Besides, [12] utilized document factors to further model the correlation between dwell time and user interest. [24] modeled normalized dwell time for directly estimating the user satisfaction. Recently, [29] proposed a joint predictor for dwell time and user click in recommendation. Although dwell time has been widely used to estimate user satisfaction and engagement, none of the existing models have modeled the article reading process sequentially in sub-document level.

2.2 Text Representation Learning

Text representation learning helps us effectively capture semantic information of the text [14], which is important in article recommendation. As deep learning based methods have been reported effective in many domains [3, 7], RNN- and CNN-based models have widely used in many text modeling tasks and achieved remarkable performance [11, 10, 9, 20, 1]. Recently, [19] conducted word-embedding-based models with parameter-free pooling operations, which achieved comparable performance over deep models in some cases. Recent years, attention mechanism has achieved dramatic success in many fields, e.g. self-attention methods for machine translation [21]. [23] proposed the Hierarchical Attention Network (HAN) for document classification, which provides promising results and makes the algorithm more interpretable. It is demonstrated that one can drop convolutional and recurrent block in deep model thus allows for significantly more parallelization and acheive even better results beacause self-attention machanism is proven to be more effective modeling long-term dependencies in sequence [21]. In the case of article recommendation, traditional methods usually model the content with document-level representation in a non-personalized way, which ignores the natural reading process of user and the user attraction in sub-document level. In this work, we design a novel framework to model the reading process, which is capable of modeling the attraction distribution in the article.

3 Model

In this section, we introduce the proposed Sequential Attentive Network (SAN) in following order. We first explain how we estimate the dwell time in Section 3.1. Then we present the framework of SAN in Section 3.2. Finally, we introduce the network training of the SAN in Section 3.3.

3.1 Estimation of Dwell Time

We model dwell time using survival analysis methods, which has sound underlying mathematical principles [22]. Survival analysis aims to model survival time, which is interpreted as the latency until the occurrence of a certain event. Therefore, it is very natural to view dwell time as the survival time of leaving the content page and model it with survival analysis methods.

We now review some basic concepts and methods in survival analysis [22, 15], which will be useful when we introduce our model. Let O denotes a random variable representing survival time. One way of modeling O is to suppose the existence of a invertible function $g(\cdot)$ such that

$$g(O) = \mu + \sigma\epsilon, \epsilon \sim f, \tag{1}$$

where f is generally selected from some simple distributions, e.g., Gaussian. With samples of O available, parameters of $g(O)$ can be estimated by maximum likelihood estimation.

Back to our task, let T denotes a random variable representing the dwell time. To motivate our choice of modeling T, we draw the histograms of T and $\log T$ in Fig.2, based on the statistics of our dataset. The figure on the left shows that the possibility decays exponentially fast over the dwell time, while the histogram of log dwell time approximately subjects to Gaussian distribution, which agrees with the findings in some previous researches [25, 24, 29]. We therefore model the $\log T$ with Gaussian distribution as follows,

$$\log T = \mu + \sigma\epsilon, \epsilon \sim N(0, 1). \tag{2}$$

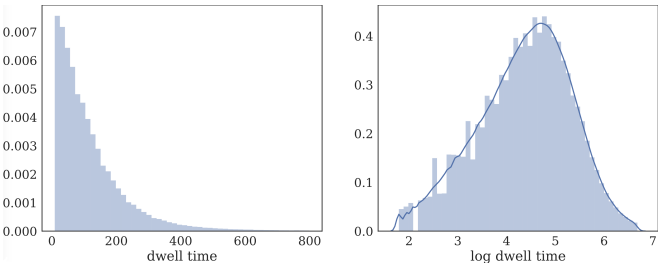

Figure 2. Histograms of dwell time and log dwell time.

And its density can be expressed as

$$P(\log T) \propto \exp(-\frac{(\log T - \mu)^2}{2\sigma^2}), \tag{3}$$

where μ is modeled by SAN representing the expectation of $\log T$, and σ is set as a hyper-parameter.

3.2 Framework

The proposed SAN is illustrated in Fig.4. In article recommendation settings, the inputs usually include three parts: user field, title field and content field. User field usually contains user ID, age and so on; title field refers to the title of article, and content field refers to the body of the article. These raw inputs will first be transformed to fixed-length vectors by the embedding layer, then we model the attraction interaction for each sentence in the attraction interaction layer, and then generate attraction distribution via the following attraction self-attention layer. We will demonstrate each step in detail in the following sub-sections.

3.2.1 Embedding Layer

We transform the raw inputs into fixed-length vectors in embedding layer. For user field, we transform each feature into an embedding vector then average it to generate the user representation $\boldsymbol{u} \in \mathbb{R}^d$. For title field, we average all the embedding of words in the title to generate title representation $\boldsymbol{v} \in \mathbb{R}^d$. Since we want to model the sequential reading process, we view the article as a sequence, each unit of the sequence contain several continuous sentences, the number of sentence in one single unit has a significant impact on the computational efficiency of the model, as well as the number of parameter in our model. Within the units, we introduce attention mechanism to extract such words that the user cares much about and aggregate the representation of those informative words to form a vector which represent the unit. Specifically,

$$\boldsymbol{z}^w = \boldsymbol{H}_w \boldsymbol{u}, \tag{4}$$

$$a_{p,q} = \frac{\exp((\boldsymbol{z}^w)^\top \boldsymbol{w}_{p,q})}{\sum_t \exp((\boldsymbol{z}^w)^\top \boldsymbol{w}_{p,t})}, \tag{5}$$

$$\boldsymbol{s}_p = \sum_t a_{p,t} \boldsymbol{w}_{p,t}. \tag{6}$$

We first transform the user vector $\boldsymbol{u}$ into the word-level attention vector $\boldsymbol{z}^w$ using the transition matrix $\boldsymbol{H}_w \in \mathbb{R}^{d \times d}$. Then we measure the importance of the q-th word in the p-th unit as the similarity of the word embedding $\boldsymbol{w}_{p,q}$ with the word-level attention vector $\boldsymbol{z}^w$ and get a normalized importance weight using a softmax function. After that, we generate the sentence vector $\boldsymbol{s}_p$ of the p-th unit as a weighted sum of the word embeddings based on the weights.

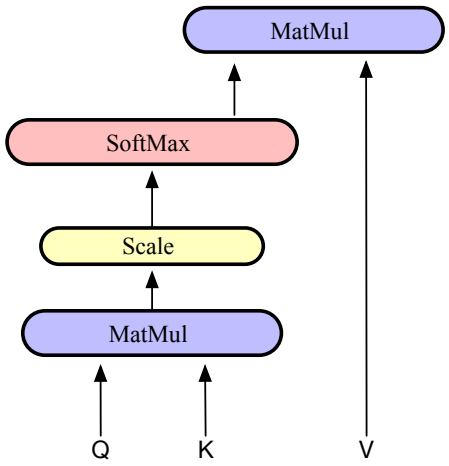

Figure 3. Scaled Dot-Product Attention.

Finally, we represent the article as a matrix $\boldsymbol{E} \in \mathbb{R}^{n \times d}$, where n donates the number of units of the article. Each row in $\boldsymbol{E}$ is a representation vector for a unit. The attention block does not include any recurrent or convolutional module, so it is not aware of the order of sentences. We need to inject some information about the relative or absolute position of the tokens in the sequence, and add positional embedding $\boldsymbol{P} \in \mathbb{R}^{n \times d}$ into the sentence embedding $\boldsymbol{E} \in \mathbb{R}^{n \times d}$:

$$\boldsymbol{E} = \begin{bmatrix} \boldsymbol{s}_1 + \boldsymbol{P}_{1,:} \\ \boldsymbol{s}_2 + \boldsymbol{P}_{2,:} \\ \cdots \\ \boldsymbol{s}_n + \boldsymbol{P}_{n,:} \end{bmatrix}, \tag{7}$$

where $\boldsymbol{s}_i$ is the representation vector for the i-th unit, $\boldsymbol{P}_{i,:}$ is the corresponding position embedding vector, which are learnable parameters that trained simultaneously with the weights of the network by optimizing a single loss function defined with respect to the end task. We adopt the positional embedding method proposed by [6].

3.2.2 Attraction Interaction Layer

In order to model attraction distribution, we first model the attraction level of each sentence in the article. Therefore we need a measurement for attraction. We believe the attraction level of a sentence to a reader is closely related to the relevance between the interests of user and the semantic information of this sentence. Since reader is first attracted by information provided by the title, then decides to click in and reads the article, we believe that the relevance between the title and a sentence is also a factor influencing the attraction level of this sentence. Therefore, we model the attraction of a particular sentence by modeling the three-side interaction between this sentence, user interests and title.

Specifically, we take the element-wise product to model the three-way interaction. The output of this layer $\boldsymbol{H} \in \mathbb{R}^{n \times d}$ can be formulated as:

$$\boldsymbol{H} = \begin{bmatrix} \boldsymbol{E}_{1,:} \odot \boldsymbol{v} \odot \boldsymbol{u} \\ \boldsymbol{E}_{2,:} \odot \boldsymbol{v} \odot \boldsymbol{u} \\ \cdots \\ \boldsymbol{E}_{n,:} \odot \boldsymbol{v} \odot \boldsymbol{u} \end{bmatrix}, \tag{8}$$

where $\boldsymbol{E}_{i,:}$ is the i-th row of $\boldsymbol{E}$, equally the representation of i-th sentence in the article, and $\boldsymbol{v} \in \mathbb{R}^n$ is the representation of the title,

Table 1. Maximum path lengths, per-layer complexity and minimum number of sequential operations for different layer types.

Layer Type	Complexity per Layer#	Sequential Operations#	Max Path Length#
Self-Attention	$O(n^2d)$	$O(1)$	$O(1)$
Recurrent	$O(nd^2)$	$O(n)$	$O(n)$
Convolutional	$O(knd^2)$	$O(1)$	$O(\log_k(n))$

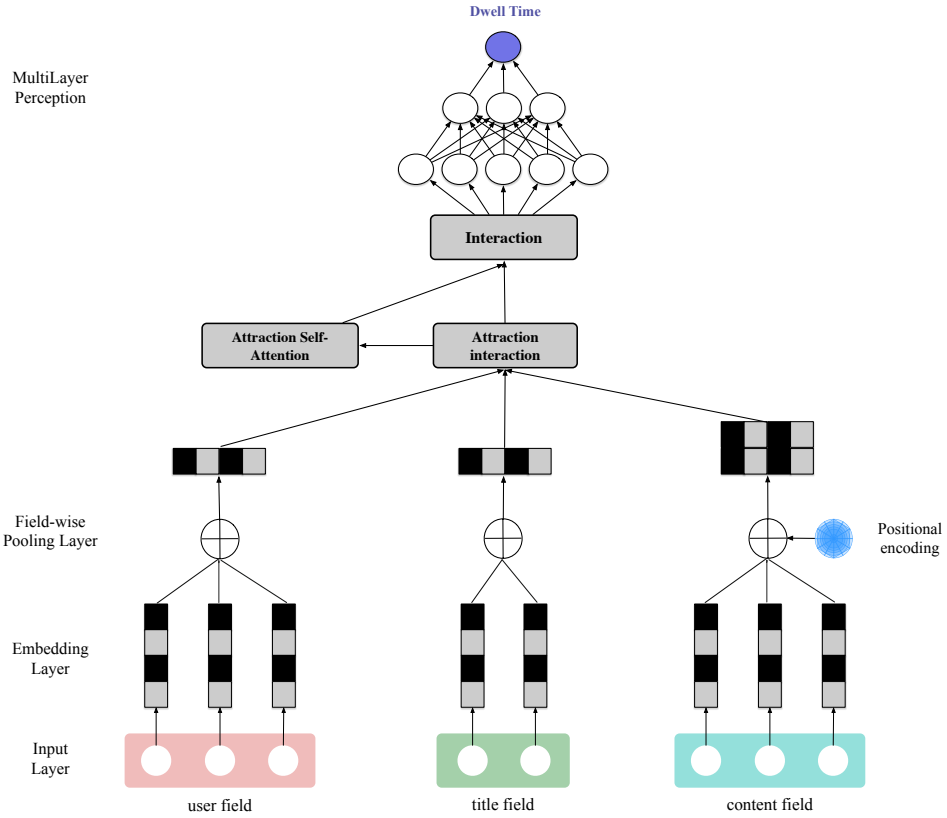

Figure 4. Architecture overview of SAN.

$\boldsymbol{u} \in \mathbb{R}^n$ is the representation of the user, which represents interests of the corresponding user.

3.2.3 *Attraction Self-attention Layer*

With the output of attraction interaction layer, we are ready to model the distribution of attraction. The insight here is that the former content affects the reading for the latter content via the change of attraction level, thus we need to effectively capture dependencies in attraction level of different sentences. Since the article is relatively long, the dependencies are likely to be long-term dependencies. One key factor affecting the ability to learn such dependencies is the length of the paths forward and backward signals have to traverse in the network. The shorter these paths between any combination of positions in the input and output sequences, the easier it is to learn long-range dependencies [8]. A typical way to model dependency in sequence is to utilize recurrent neural networks (RNNs). However, RNNs have a much longer path length than self-attention based methods. As illustrated in Table 1, where n is the sequence length, d is the representation dimension, k is the kernel size of convolutions. A self-attention layer connects all positions with a constant number of sequentially executed operations, whereas a recurrent layer requires $O(n)$ sequential operations [21]. Besides, attention block allows for significantly more parallelization since there is no recurrence in the architecture. Thus we utilize self-attention mechanism to model attraction distribution.

Self-attention, sometimes called intra-attention is an attention mechanism relating different positions of a single sequence in order to compute a representation of the sequence [21]. The general framework of scaled dot-product attention is illustrated in Fig.3, and is formally defined as:

$$\text{Attention}(\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V}) = \text{softmax}\left(\frac{\boldsymbol{Q}\boldsymbol{K}^T}{\sqrt{d}}\right)\boldsymbol{V}, \tag{9}$$

where $\boldsymbol{Q}$ represents the queries, $\boldsymbol{K}$ the keys and $\boldsymbol{V}$ the values. It calculates the products of the query with all keys, divide each by $\sqrt{d}$, which is a scale factor, and applies a softmax function to obtain the weights on the values. Self attention is a special case of scaled dot-product attention, where the queries, keys, and values are all the same.

Specifically, we first convert $\boldsymbol{H}$ to three matrices through linear

projections, then feed them into the self-attention layer:

$$\boldsymbol{M} = \text{Attention}(\boldsymbol{H}\boldsymbol{W}^Q, \boldsymbol{H}\boldsymbol{W}^K, \boldsymbol{H}\boldsymbol{W}^V), \tag{10}$$

where the projection matrices $\boldsymbol{W}^Q, \boldsymbol{W}^K, \boldsymbol{W}^V \in \mathbb{R}^{d\times d}$. Note that we uses three different projection matrices, which make the model more flexible.

In the self-attention block, we model the interaction between all pairs of rows in $\boldsymbol{H}$, each row of the outputs $\boldsymbol{M}$ is a weighted sum of all rows of $\boldsymbol{H}\boldsymbol{W}^V$, the weight is the corresponding pairwise interaction, thus we model all the sequential dependencies in the sequence of attraction level directly. In other words, the model has unit-length paths between any combination of positions in the input, which is a key factor that the model captures attraction distribution effectively.

3.2.4 Prediction Layer

We find that it is helpful to consider the outputs of previous layers, similar as residual connection [7], instead of only considering the output of the last layer. We first concat the outputs of previous layers and the context information:

$$\boldsymbol{F} = \text{concat}[\boldsymbol{M}, \boldsymbol{H}, \boldsymbol{C}], \tag{11}$$

where $\boldsymbol{M}$, $\boldsymbol{H}$, $\boldsymbol{C} \in \mathbb{R}^{n\times d}$, and $\boldsymbol{F} \in \mathbb{R}^{n\times 3d}$. $\boldsymbol{C}$ is context information matrix, in which each row is a context vector, e.g., the length of the article. We then endow the model with nonlinearity and to consider interactions between different latent dimensions:

$$\log \widehat{t} = \phi_{MLP}(\boldsymbol{F}\boldsymbol{k}), \tag{12}$$

where $\boldsymbol{k} \in \mathbb{R}^{3d}$ is trainable parameter vector, and ϕ_{MLP} refers to a fully-connected feed forward neural network, which maps the input to a real number. Note that the ϕ_{MLP} can be replaced by more sophisticated recommendation frameworks. In this work, we focus on content modeling for dwell time prediction and thus we adopt simple methods for the mapping.

3.3 Training SAN

The loss function of our model is defined as:

$$\mathcal{L} = \sum_{(i,j)\in\mathcal{Y}} l(\log t_{ij}, \log \hat{t}_{ij}), \tag{13}$$

where t_{ij} denote the dwell time for i-th user on j-th article, and $\mathcal{Y}$ denotes the instances, and $l(\cdot,\cdot)$ is the loss function. To be specific, we take the Eq.(3) as the likelihood for the logarithm of dwell time and define the loss as follows:

$$l(\log t_{ij}, \log \hat{t}_{ij}) = (\log t_{ij} - \log \hat{t}_{ij})^2. \tag{14}$$

We adopt Adam optimizer [13], a variant of Stochastic Gradient Descent (SGD) to train the network.

4 Experiments

We conduct experiments to answer the following questions:

- **(Q1)** How does our proposed SAN perform in dwell time prediction compared to the baselines?
- **(Q2)** Is the model effective in modeling the attraction distribution in the content?
- **(Q3)** Is it helpful considering the order of sentences?
- **(Q4)** How do the settings of networks influence the performance of SAN?

In what follows, we first present the experimental settings, followed by answering the above research questions one by one.

4.1 Experiment Setup

4.1.1 Datasets

We collect our datasets from Top Stories (a built-in service of WeChat), which provides article recommendation service. WeChat is a Chinese multi-purpose messaging, social media and mobile payment app developed by Tencent, with over 1 billion daily active users. And there are millions of users using Top Stories in WeChat everyday.

Since the preference of users might change over a long time [28], we decide to focus on our study over a short period of time. We collect two datasets with different size, which helps to verify the impact of the volume of the dataset. The first dataset is collected from five consecutive days' logs, while another dataset is collected from seven consecutive days' logs. Besides, there are more than a month gap between the two sampled datasets, and no overlap user or articles between them. The statistics of the two datasets are summarized in Table 2, where M indicates million and K indicates thousand.

Table 2. Statistics of the datasets.

Dataset	User#	Instance#	Article#
ST10K	10K	550K	79K
ST50K	50K	2.5M	340K

Instances in the two datasets have the same format, as described in Table 3. The ID in title field is the same with the ID in content field within an instance, since they indicate the same article.

Table 3. Descriptions of the data format in the datasets.

User	Title	Content	DT
ID,sex,age	ID,title,tag	ID,doc,tag,length	log DT

As suggested in [4], for both datasets, we keep the last day's data for testing, and the rest for training and validation, this is more convincing than testing on the randomly held-out data because it avoids leaking future information.

4.1.2 Evaluation Metrics

We use three metrics to evaluate the performance: **MAE** (Mean Absolute Error), **RMSE** (Rooted Mean Squared Error), **NDCG** (Normalized Discounted Cumulative Gain). MAE and RMSE measure the point-wise error between the predicted value and the real value, which are both widely used for evaluating regression tasks. NDCG measures the list-wise rank quality of the predicted values. Specifically, we rank the articles according the estimated dwell time and then calculate the NDCG score. In our experiments, we calculate NDCG@5 and NDCG@10 for each test user and report the average score.

4.1.3 Baselines

Since we focus on content modeling for dwell time prediction, we choose several representative baselines for text modeling and conduct experiments over them to answer the four questions. Note that some methods mentioned in related works are not word content modeling methods, thus they are not included in the baselines.

BOW. This model utilizes bag-of-words method to generate the text representation, then predict dwell time with the inner-product of user embedding and the text representation.

Table 4. Performance of compared models on two datasets.

	ST10K				ST50K			
Model	MAE	RMSE	NDCG@5	NDCG@10	MAE	RMSE	NDCG@5	NDCG@10
BOW	61.74	87.17	0.419	0.571	60.63	86.24	0.362	0.422
Char CNN	58.11	83.69	0.462	0.633	56.93	84.27	0.412	0.472
HAN-*gru*	57.73	83.49	0.469	0.642	56.69	82.83	0.420	0.481
HAN-*pooling*	58.52	84.21	0.451	0.621	57.42	84.87	0.403	0.461
SAN-*single*	56.81	83.13	0.474	0.642	55.43	81.38	0.417	0.486
SAN-*unordered*	56.62	82.79	0.477	0.643	55.54	82.12	0.421	0.488
SAN-*gru*	56.11	82.27	0.484	0.655	55.03	81.77	0.434	0.499
SAN-*complete*	**54.87**	**80.72**	**0.502**	**0.675**	**53.82**	**80.63**	**0.447**	**0.516**

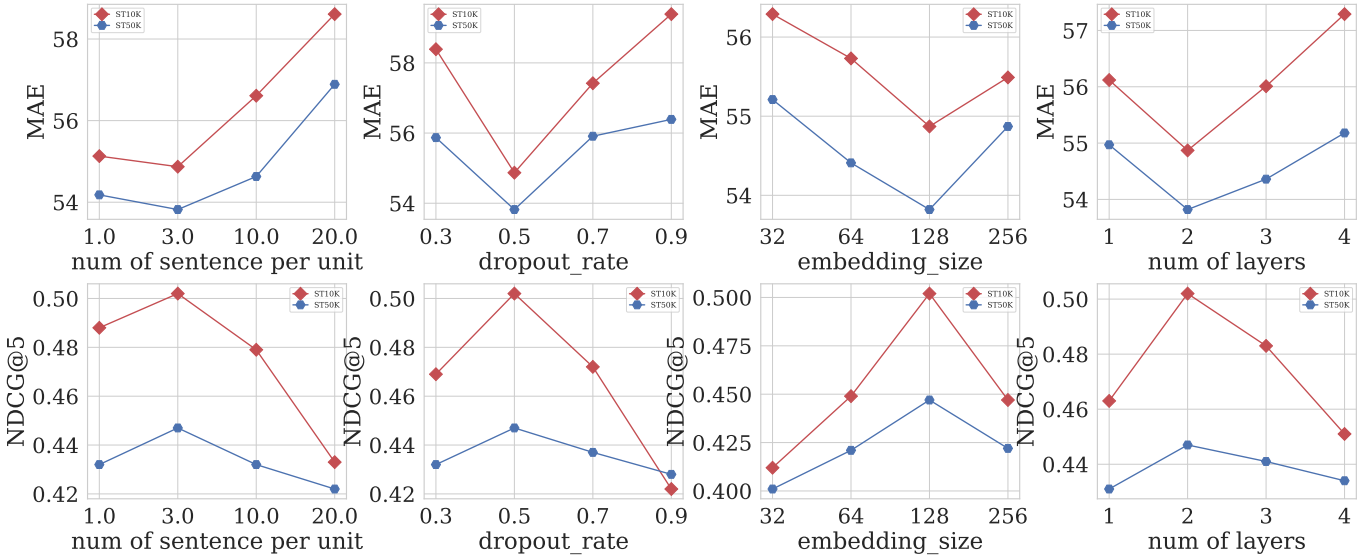

Figure 5. Impact of network hyper-parameters on dwell time prediction.

Char CNN [27]. This model utilizes character-level convolutional networks to generate text representation. Specifically, 1-D convolution operation is applied at character-level, and the order of sentences is considered. However, this still belongs to document-level modeling since the user-engagement is evaluate at document-level.

HAN-*gru* [23]. This model utilizes attention mechanism to consider the importance of different parts of article and utilizes GRU [2] as encoder to generate a better text representation. The attention mechanism is applied both at word-level and sentence-level, in which the user vector are transformed to scalar as an measurement of importance for each word and sentence. The sentences are weighted and sent to GRU to generate the final vector representing the article.

HAN-*pooling*. This refers to a simplified version of HAN, which replaces GRU with bag-of-words method as the encoder. The sentences are weighted by attention mechanism and summed to generate the final vector representing the article.

SAN-*single*. This refers to our proposed model without the attraction interaction layer. In this way, we utilize self attention method for document-level modeling, the user representation is not involved in the document modeling process.

SAN-*unordered*. This refers to our proposed model without the positional encoding.

SAN-*gru*. This model utilizes GRU to model attraction distribution instead of self-attention, we view the rows of $\boldsymbol{H}$ as the input of GRU to capture dependencies within the sequence.

SAN-*complete*. This refers to the complete version our proposed model.

4.1.4 *Parameter Settings*

We implement our model using Tensorflow[4]. For optimization, we use the Adam [13] with a mini-batch size of 1024, where the learning rate is set to 0.001. Hyper-parameters in our model and all the baselines are tuned on the validation set, and the final comparison is conducted based on their best settings in our experiment. To alleviate the overfitting problem, we use a L_2 regularization with $\lambda = 1e^{-6}$, and dropout 0.5 in the MLP. For the activation function, we choose ReLU function. We use the vocabulary of 20,000 words (a mixture of Chinese words and characters) on ST10K, and 50,000 words on ST50K, both covering more than 98% words appeared in the documents, where all the other characters are replaced by a special token "UNK".

4.2 Performance Comparison on Dwell Time Prediction (Q1)

From Table 4, We observe that the proposed SAN outperforms the other baselines on two datasets. BOW and Char-CNN generate non-personalized text representation and completely separate the text

[4] https://www.tensorflow.org/

modeling process from user interests and ignore the sub-document level interaction between user and content, thus fails to capture the dynamic during reading. HAN-gru and HAN-pooling generate personalized text representation by using user interests to measure the importance of the sentences in article, but they still firstly generate a separate text representation, and then measure the one-off interaction between the user and text to predict dwell time, instead of interacting user interests with sub-document units. These methods are modeling dwell time on document level, while we model the natural sequential reading process of user and the user attraction in sub-document level, which is able to uncover more complex and fine-grained dynamics during the reading process.

4.3 The Effectiveness of Modeling Attraction Distribution (Q2)

SAN-*gru* and SAN-*complete* utilize different methods to model attraction distribution. As shown in Table 4, they outperform all other models, note that SAN-*complete* and SAN-*single* both utilizes self-attention method, the only difference is that SAN-*complete* involves user representation in the document modeling process in order to model attraction distribution, the results indicate that it is helpful to model dynamic changes of attraction during reading. On the other hand, SAN-*complete* is the best individual model, which demonstrates the effectiveness of self-attention mechanism for capturing sequential dependencies because the model has unit-length paths between any combination of positions in the input.

4.4 The Effectiveness of Considering the Order of Sentences (Q3)

In order to model attraction distribution, the model has to be aware of the order of sentences. In SAN-*unordered*, we drop the positional encoding for comparison. That is, the model ignores the natural sequential reading process of user, which leads to a worse performance than SAN-*complete*. Among the baselines, HAN-*gru* and Char CNN consider the order of sentence, they perform better than BOW and HAN-*pooling*. This indicates that it is worthwhile considering the order. HAN-*gru* outperforms the baselines that model dwell time on document level, which further demonstrates the importance of sub-document level modeling.

4.5 Hyper-Parameter Study (Q4)

We study the impact of hyper-parameters on SAN for dwell time prediction in this section, including (1) the number of sentence per unit; (2) dropout rate; (3) the embedding size of feature mapping; and (4) the number of hidden layers in the MLP. We choose SAN-*complete* for following experiments.

Number of sentences per unit. This parameter indicates the granularity of our sub-document modeling. As the number of sentences per unit increases, the number of parameters in SAN decreases, which alleviates over-fitting problem and optimization difficulty. As Fig.5 shows, when there is one sentence per unit, the performance is slightly worse than two sentences, this may because of over-fitting problem and optimization difficulty. However, when the number of sentences per unit is large, the model fails to capture fine-grained dynamics during reading process, as Fig.5 shows, this hurts the performance. When a single unit contain all of the sentences in article, the model degenerated to document-level modeling. The results show the importance of sub-document modeling.

Dropout rate. As Fig.5 shows, the performance increases with dropout rate at the beginning, since it alleviates the over-fitting problem. However, the performance degrades when dropout rate is too large because this leads to under-fitting problem. We also observe that ST50K suffers less over-fitting problem when dropout rate is low, this maybe because this dataset is much larger.

Embedding size for feature mapping. The parameters in embedding layer usually contributes large part of the parameters in the whole network. Increasing the embedding size has a risk of over-fitting. As shown in Fig.5, the performance in both datasets begins decreasing when the embedding size is greater than 128.

Number of hidden layers in the MLP. In this experiment, we fix the number of neurons per layer as 128. Fig.5 demonstrates the impact of the number of hidden layers in the MLP. We can observe that the performance of SAN-*complete* increases with the depth of network at the beginning. However, model performance degrades when the depth of network is set greater than 2. This might be also caused by overfitting since the loss of training data keeps decreasing when we add more hidden layers.

5 Conclusions and Future Work

In this paper, we proposed a framework named Sequential Attentive Network (SAN), which models the natural sequential reading process of the user and the user attraction distribution in sub-document level. The proposed model captures sequential dependencies in attraction distribution effectively, thus is able to uncover more complex and fine-grained dynamics during the reading process. We conducted comprehensive experiments and the results demonstrate that our SAN outperforms other typical baselines on real-world datasets. For the future work, we believe it is worth considering other methods of making use of context information, as well as combining document level and sub-document level modeling together for dwell time prediction.

Acknowledgements

The research work is supported by the National Key Research and Development Program of China under Grant No. 2018YFB1004300, the National Natural Science Foundation of China under Grant NOs. U1811461, 61773361, U1836206, the Project of Youth Innovation Promotion Association CAS under Grant No. 2017146. This work is also partly supported by the funding of WeChat cooperation project.

REFERENCES

[1] Jingwu Chen, Fuzhen Zhuang, Tianxin Wang, Leyu Lin, Feng Xia, Lihuan Du, and Qing He, 'Follow the title then read the article: Click-guide network for dwell time prediction', *IEEE Transactions on Knowledge and Data Engineering*, (2019).

[2] Kyunghyun Cho, Bart Van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio, 'On the properties of neural machine translation: Encoder-decoder approaches', *arXiv preprint arXiv:1409.1259*, (2014).

[3] Ronan Collobert and Jason Weston, 'A unified architecture for natural language processing: Deep neural networks with multitask learning', in *ICML*, (2008).

[4] Paul Covington, Jay Adams, and Emre Sargin, 'Deep neural networks for youtube recommendations', in *RecSys*. ACM, (2016).

[5] Steve Fox, Kuldeep Karnawat, Mark Mydland, Susan Dumais, and Thomas White, 'Evaluating implicit measures to improve web search', *TOIS*, (2005).

[6] Jonas Gehring, Michael Auli, David Grangier, Denis Yarats, and Yann N Dauphin, 'Convolutional sequence to sequence learning', in *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pp. 1243–1252. JMLR. org, (2017).

[7] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun, 'Deep residual learning for image recognition', in *CVPR*, (2016).
[8] Sepp Hochreiter, Yoshua Bengio, Paolo Frasconi, Jürgen Schmidhuber, et al. Gradient flow in recurrent nets: the difficulty of learning long-term dependencies, 2001.
[9] Sepp Hochreiter and Jürgen Schmidhuber, 'Long short-term memory', *Neural computation*, (1997).
[10] Andrej Karpathy and Li Fei-Fei, 'Deep visual-semantic alignments for generating image descriptions', in *CVPR*, (2015).
[11] Yoon Kim, 'Convolutional neural networks for sentence classification', in *EMNLP*, (2014).
[12] Youngho Kim, Ahmed Hassan, Ryen W White, and Imed Zitouni, 'Modeling dwell time to predict click-level satisfaction', in *WSDM*, (2014).
[13] Diederik P Kingma and Jimmy Ba, 'Adam: A method for stochastic optimization', *arXiv preprint arXiv:1412.6980*, (2014).
[14] Quoc Le and Tomas Mikolov, 'Distributed representations of sentences and documents', in *ICML*, (2014).
[15] Yan Li, Kevin S Xu, and Chandan K Reddy, 'Regularized parametric regression for high-dimensional survival analysis', in *SIAM*, (2016).
[16] Chao Liu, Ryen W White, and Susan Dumais, 'Understanding web browsing behaviors through weibull analysis of dwell time', in *SIGIR*, (2010).
[17] Yiqun Liu, Xiaohui Xie, Chao Wang, Jian-Yun Nie, Min Zhang, and Shaoping Ma, 'Time-aware click model', *TOIS*, (2017).
[18] Hongyu Lu, Min Zhang, and Shaoping Ma, 'Between clicks and satisfaction: Study on multi-phase user preferences and satisfaction for online news reading', in *SIGIR*, (2018).
[19] Dinghan Shen, Guoyin Wang, Wenlin Wang, Martin Renqiang Min, Qinliang Su, Yizhe Zhang, Chunyuan Li, Ricardo Henao, and Lawrence Carin, 'Baseline needs more love: On simple word-embedding-based models and associated pooling mechanisms', in *ACL*, (2018).
[20] Ilya Sutskever, Oriol Vinyals, and Quoc V Le, 'Sequence to sequence learning with neural networks', in *NIPS*, (2014).
[21] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin, 'Attention is all you need', in *NIPS*, (2017).
[22] Ping Wang, Yan Li, and Chandan K Reddy, 'Machine learning for survival analysis: A survey', *arXiv*, (2017).
[23] Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Hovy, 'Hierarchical attention networks for document classification', in *NAACL*, (2016).
[24] Xing Yi, Liangjie Hong, Erheng Zhong, Nanthan Nan Liu, and Suju Rajan, 'Beyond clicks: dwell time for personalization', in *RecSys*. ACM, (2014).
[25] Peifeng Yin, Ping Luo, Wang-Chien Lee, and Min Wang, 'Silence is also evidence: interpreting dwell time for recommendation from psychological perspective', in *KDD*, (2013).
[26] Shuai Zhang, Lina Yao, Aixin Sun, and Yi Tay, 'Deep learning based recommender system: A survey and new perspectives', *ACM Computing Surveys (CSUR)*, **52**(1), 1–38, (2019).
[27] Xiang Zhang, Junbo Zhao, and Yann LeCun, 'Character-level convolutional networks for text classification', in *Advances in neural information processing systems*, pp. 649–657, (2015).
[28] Guanjie Zheng, Fuzheng Zhang, Zihan Zheng, Yang Xiang, Nicholas Jing Yuan, Xing Xie, and Zhenhui Li, 'Drn: A deep reinforcement learning framework for news recommendation', in *WWW*, (2018).
[29] Tengfei Zhou, Hui Qian, Zebang Shen, Chao Zhang, Chengwei Wang, Shichen Liu, and Wenwu Ou, 'Jump: A joint predictor for user click and dwell time', in *IJCAI*, (2018).