TaoType: Predicting Fine-Grained Typing Intent for Faster Search

Yipeng Yu*, **Yichen Yuan, Chengxiao Feng, Xu Liu**
Taobao & Tmall Group of Alibaba, Hangzhou, China
{linxin.yyp,xiaochen.yyc,yinfeng.fcx,steven.liu}@alibaba-inc.com

Abstract

"*Is the user's current query input exactly what they intend to search for?*" Our work aims to answer this question by determining, at each typing, whether the current query is complete. If so, a search is implicitly triggered in advance without waiting for user confirmation. This approach reduces response time and enhances the user search experience. Specifically, we propose TaoType, a client-side framework that introduces innovation in data sampling, feature selection, model design and training, and online strategy. Experiments on a leading mobile shopping application named Taobao validate its effectiveness, achieving offline precision/recall/accuracy of 0.7936/0.8196/0.7742, respectively, and decreasing online response time by 640.51±93.65 milliseconds, which is of great benefit to the search system. Unlike prior work that focuses on optimizing server-side engineering pipelines or simplifying ranking models, our method leverages client-side typing behavior for real-time early prediction, utilizing on-device computation to gain response time reducing. To the best of our knowledge, our work is the first to identify and address this problem. This work also introduces App Intelligence, a new paradigm for enhancing mobile applications by integrating on-device AI to boost business value and user experience.

1 Introduction

Large-scale industrial ranking systems, prevalent in domains such as search, recommendation, and advertising, operate under stringent real-time performance requirements. For instance, Taobao's ranking pipeline, which serves roughly 2 billion items, comprises recall, pre-ranking, fine ranking, and re-ranking stages, each with an RT budget of under ~100 ms. While shorter RT directly contribute to a better user experience, they can potentially compromise the quality of the delivered items. As a result, a trade-off between effectiveness and performance is often necessary. The rise of large language models (LLMs) has ushered in the era of generative ranking (Zhai et al., 2024; Tian et al., 2025; Zhang et al., 2026), a new paradigm that is being successfully implemented in industry. However, the performance gains from this approach are accompanied by the challenge of increased RT, further complicating the established trade-off.

To reduce the ranking RT, previous work have typically focused on two main strategies: optimizing the engineering pipeline (Shi et al., 2025) or applying model compression techniques such as distillation, pruning, and quantization (Wei et al., 2025). Nevertheless, the scope for RT reducing with these strategies is constrained, and they often introduce trade-offs such as greater machine costs and a decline in ranking quality.

Unlike the two strategies, our approach reduces RT by proactively initiating searches. Our method is based on the key observation that a time delay, denoted as dT, exists between the moment a user completes a query input and the moment they initiate the search. This insight led us to introduce a novel binary classification challenge: "*Is the user's current query input exactly what they intend to search for?*" To address this challenge, we propose TaoType, an innovative and effective client-side framework that models typing dynamics based on conditional random field (CRF) model. This framework performs a prediction within the app each time the query is modified. If the prediction is true, i.e., the input is complete, it launches the search for the query in advance without waiting for the user's confirmation, thereby reducing the RT by approximately dT.

Essentially, TaoType leverages on-device computing power to reduce RT. As on-device hardware and AI capabilities continue to advance, the AI functionalities within applications can be progressively enhanced. Therefore, we propose a new

*Correspondence: yypzju@163.com

Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 6: Industry Track), pages 184–192

paradigm which we term "App Intelligence". It aims to improve business value and user experience by enhancing AI capabilities directly within applications. Our work serves as a case study for this paradigm. In addition, this paper thoroughly differentiates App Intelligence from related concepts such as web intelligence and edge AI, and discusses its synergistic relationship with LLMs. Our contributions are as follows:

- **RT Reducing by Presearch** - We introduce a new perspective to reducing RT by predicting whether a query is complete based the typing behavior, enabling initiating search requests in advance and thereby minimizing user waiting time. Typing patterns during search on the client-side exhibit both population-level and individual-level regularities. By leveraging these behavioral patterns, we can make accurate early predictions and act in advance.

- **TaoType Framework** - We build TaoType, a client-side framework that introduces innovation in data sampling, feature selection, model design and training, and online strategy. Offline and online experiments validate its effectiveness and superiority. It delivers significant RT benefits while introducing minimal computational overhead.

- **App Intelligence** - We propose a novel intelligence paradigm termed App Intelligence by synthesizing the advancements in AI technology with the demands of practical application scenarios. It integrates lightweight AI capabilities into mobile applications. This paradigm enhances user experience while simultaneously delivering measurable improvements in business value.

2 Related Work

2.1 From Query to Intent

The query-to-intent tasks are predominantly processed on the server-side (Li et al., 2022), while only a very small fraction are handled on-device (Kag et al., 2022). Moreover, prior work primarily focuses on text semantics, paying little attention to on-device user behaviors. Even when such behaviors are considered, they are often captured at a coarse granularity and lack real-time responsiveness (Jiang et al., 2014; Fan et al., 2022). Nevertheless, the value of the fine-grained user behaviors had already been demonstrated and validated in the context of web intelligence (Guo et al., 2009; Guo and Agichtein, 2010). Another notable shift is that the intent of determining whether the current query is complete has not been thoroughly investigated, and no prior work has applied it to RT reducing.

2.2 From Edge Signal to Edge AI

Edge signals have been shown valuable for understanding user intent, attention, and preferences. Li found that users' explicit and implicit feedback behaviors were closely associated with their purposes on mobile social media platforms (Li et al., 2025). Cai designed a gaze estimation method with finger-swipe gestures (Cai et al., 2025). These works merely incorporate edge signals but do not deploy the models directly on the device. Beyond that, Guo proposed deep neural networks to predict real-time user purchasing intent at the edge (Guo et al., 2019, 2020). Qian designed an intelligent request strategy to capture the dynamic change of users' intentions by matching their real-time mobile behaviors (Qian et al., 2022). Recent research in edge AI also encompasses TinyML (Lin et al., 2023), which aims to enable on-device model training and co-adaptation, as well as MobileLLM (Zhang et al., 2024), which deploys LLMs on device to serve as the brain of mobile applications. Nevertheless, the body of work on edge AI for query-related tasks is exceptionally limited, let alone for the purpose of RT reduction. In our scenario, an intent classification is triggered with every typing by the user. This high-frequency execution faces more challenges in model size, efficiency, and performance.

3 Problem Statement

After each typing of a query session, the user has three possible intentions (see Figure 1). The first is to continue typing and editing the query (intent 1). The second is to submit the current query for search (intent 2). This can be done in two ways: either by clicking the "搜索(search)" button located on the right side of the query input box (intent 2A), or by pressing the "search" key on the keyboard (intent 2B). The third is to select one of the query suggestions that listed below the input box (intent 3). Taking the completion of Type 3 in Figure 1 as an example, if the user intends to search for "iphone 17 pro", they will directly initiate a search (intent 2). If the user intends to search for "iphone 17 pro case", they may either select the first dropdown

Figure 1: Fine-grained user typing intents.

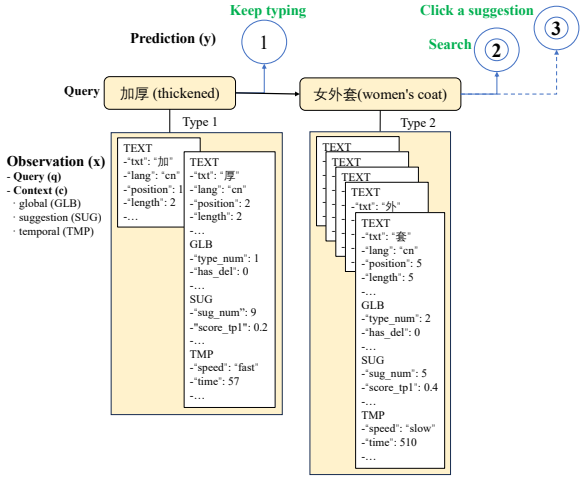


Figure 2: The CRF implementation of TaoType.

suggestion "iphone17pro手机壳(case)" (intent 3) or continue typing the word "case" (intent 1). Thus, if the user's intent is predicted as intent 2 at the end of each typing, we can initiate the search request immediately without waiting for an explicit click action. The time delay (dT) is eliminated, thereby reducing RT by ~dT. Furthermore, we report the mean dT for intent 2: for intent 2A, the mean dT was 2506 ms on Android and 3605 ms on iOS; for intent 2B, the mean dT was 845 ms on Android and 687 ms on iOS. This indicates that a substantial reduction in RT can be achieved if intent 2 is predicted accurately.

Here, we state the problem as a classification of the dynamic sequential data. Let $\mathbf{x} = \langle x_1, x_2, \cdots, x_T \rangle$ denotes a query session, where each observation x_t, $t = 1, \cdots, T$ includes a query q_t and a set of contextual features c_t. Let $\mathcal{D} = \{(\mathbf{x}^{(n)}, \mathbf{y}^{(n)})\}_{n=1}^{N}$ be a set of labeled query sessions, where each $\mathbf{x}^{(n)}$ is a query session, and $\mathbf{y}^{(n)}$ is its corresponding labeling sequence. Note that $\mathbf{y}^{(n)}_{t=T}$ is either intent 1, intent 2, or intent 3, and $\mathbf{y}^{(n)}_{t<T}$ is intent 1. Our goal is to learn a classifier $h(\mathbf{x}; \mathbf{w})$ based on the labeled query sessions, so that given a new query q_t, it can correctly classify q_t into one of the intents, taking into account the search context of q_t.

4 Modeling Typing Dynamics

As shown in Figure 2, we model search contexts embedded in the query with a linear-chain CRF model (Lafferty et al., 2001). After each typing, TaoType predicts the user's current intent, specifically whether they will keep typing, search, or click one of the suggestions. If the predicted intention is to search, the system can proactively retrieve results for the current query. The prediction is based on features derived from both the textual content of the query characters and contextual information. The character-level features are sequentially concatenated in the order of input. Contextual features include global (GLB), suggestions (SUG), and temporal (TMP) features. These contextual features are appended to the textual representation of the final character in the sequence. Their details are presented in Figure 2 and Section 5.1. In mathematical terms, the prediction is defined as:

$$p(\mathbf{y}|\mathbf{x}) = \frac{1}{Z(\mathbf{x})} \exp\left(\sum_t \sum_k \omega_k f_k(y_{t-1}, y_t, \mathbf{x})\right) = \frac{e^{\mathbf{w}^\top \mathbf{f}_\mathbf{x}(\mathbf{y})}}{Z(\mathbf{x})} \tag{1}$$

, where $\mathbf{f} = \{f_k\}$ is a set of feature functions, and $Z(\mathbf{x}) = \sum_{y'} e^{\mathbf{w}^\top \mathbf{f}_\mathbf{x}(y')}$ is a normalization function. Given a set of trained parameters $\mathbf{w}^*$, a query session $\mathbf{x}$ is classified as

$$\hat{\mathbf{y}} = h(\mathbf{x}; \mathbf{w}^*) = \arg\max_{y'} \mathbf{w}^{*\top} \mathbf{f}_\mathbf{x}(y') \tag{2}$$

5 Offline Experiments and Analysis

5.1 Experimental Setup and Dataset

Data split Training data are from dates $[T-6, T]$, and testing data are from date $T+1$.

Table 1: Baselines. The unit of size is KB.

Model	Precision	Recall	Accuracy	Size
TaoType	**0.7936**	**0.8196**	**0.7742**	78
LightGBM	0.7282	0.7523	0.7101	75
CatBoost	0.7233	0.7491	0.7054	76.8
ANN	0.6786	0.7018	0.6726	144
CNN	0.6953	0.7180	0.6854	95.8
BiLSTM	0.6853	0.7082	0.6786	153.5
GCN	0.5757	0.7735	0.6923	69.6
TinyBERT	0.6856	0.6982	0.6800	>160
BERT-CRF	0.7555	0.7827	0.7352	>160
Transformer	0.6824	0.7459	0.7051	>160

Table 2: Intent number.

Intent Number	Precision	Recall	Accuracy
TaoType	**0.7936**	**0.8196**	**0.7742**
TaoType-2	0.7333	0.7867	0.7726
TaoType-4	0.6203	0.5620	0.7351

Data preprocessing After removing emojis and URLs, each query session is converted into one or more training samples. Taking the query in Figure 1 as an example, suppose a user inputs "iphone 17 pro" and then selects the first dropdown suggestion. This query is then converted into 4 training samples: ("iphone", "intent 1"), ("iphone 17", "intent 1"), ("iphone 17 pro", "intent 3"), and ("iphone17pro手机壳", "intent 2"). The query in a training sample is converted into a feature sequence defined as: $f = \bigoplus_{i=1}^{n} \mathbf{TEXT}_i \oplus \mathbf{GLB} \oplus \mathbf{SUG} \oplus \mathbf{TMP}$. The value n denotes the number of characters in the query. The TEXT includes the character itself, language (lang), character position (position), and query length (length). The GLB consists of the typing step (type_num) and a binary indicator for whether deletion has occurred (has_del). The SUG includes the number of suggestions (sug_num) and the click score of the top-1 suggestion (score_tp1). The TMP captures typing speed (speed) and inter-type interval (time). Figure 2 also shows how the query "加厚女外套" is converted into 2 samples.

Data sampling We adopt a downsampling strategy. All instances of intent 2 and intent 3 are retained, while intent 1 is randomly undersampled to achieve an approximate ratio of intent 1 : intent 2 : intent 3 $\approx$ 4:3:2.

Metric A model prediction of 1 (true) indicates intent 2, which includes both intent 2A and intent 2B, while a prediction of 0 (false) corresponds to all other intents. Precision $= \frac{TP}{TP+FP}$, recall $= \frac{TP}{TP+FN}$, accuracy $= \frac{TP+TN}{TP+TN+FP+FN}$

5.2 Baselines

We first compare TaoType with other baseline models. To ensure fair comparison under mobile constraints, we configure all baselines to match TaoType's parameter count, latency behavior, and input feature set (see appendix A for their configuration). As shown in Table 1, TaoType demonstrates superior performance in precision, recall, and accuracy, whether compared against leading machine learning methods (LightGBM(Ke et al., 2017) and CatBoost(Dorogush et al., 2018)), small deep neural networks (ANN, CNN, BiLSTM, GCN(Yao et al., 2019)), or Transformer-based models (TinyBERT(Jiao et al., 2020), BERT-CRF, and Transformer(Vaswani et al., 2017)). The CRF-based TaoType framework is well suited for dynamic sequence modeling, whereas the baseline models are primarily designed for standalone classification tasks. Furthermore, the Transformer-based models are optimized for text-only tasks, while the query text in our training data are extremely short and accompanied by additional non-textual features. Moreover, we do not perform extensive hyperparameter tuning on the three models to adapt them for our task. Even if their performance surpassed TaoType, they would still be unsuitable for deployment on mobile clients due to their large model size and inference latency.

5.3 Intent Number

We converted the TaoType annotation scheme from three classes (TaoType), namely intent 1, intent 2, and intent 3, into two-class and four-class schemes. The two-class scheme (TaoType-2) consists of intent 2 and other, and the four-class scheme (TaoType-4) consists of intent 1, intent 2A, intent 2B, and intent 3. Table 2 shows that TaoType with three intents has the best performance. This indicates that at the feature level the differences between intent 3 and intent 1 are substantial and they should not be merged into a single intent, whereas the differences between intent 2A and intent 2B are negligible and they can be combined into one intent.

5.4 Ablation Study of Features

The ablation study on features is presented in Table 3. It can be observed that the model with only text features performs the worst, while TaoType with all three contextual features achieves

Table 3: Ablation study on contextual features.

Features	Precision	Recall	Accuracy
w/o TMP, GLB, SUG	0.5275	0.4381	0.6641
w/o GLB, SUG	0.6148	0.7396	0.6737
w/o TMP, SUG	0.6945	0.7807	0.7317
w/o TMP, GLB	0.6452	0.7650	0.7155
w/o SUG	0.7242	0.7864	0.7449
w/o GLB	0.7020	0.7756	0.7381
w/o TMP	0.7655	0.8174	0.7626
TaoType	**0.7936**	**0.8196**	**0.7742**

Table 4: Device-wise training and testing.

Train	Test	Precision	Recall	Accuracy
All	ALL	0.7936	0.8196	**0.7742**
	Android	0.7725	0.8150	0.7705
	iOS	0.8145	0.8240	0.7713
Android	ALL	0.7681	0.7920	0.7654
	Android	0.7851	0.7882	**0.7801**
	iOS	0.7529	0.7750	0.7480
iOS	ALL	0.7511	0.8314	0.7582
	Android	0.6283	0.8520	0.7248
	iOS	0.8081	0.8302	**0.7727**

the best performance. The order of importance for these contextual features is GLB$\succ$SUG$\succ$TMP. It should be noted that although incorporating the SUG feature improves prediction performance, it increases RT because generating the SUG feature after the user completes typing incurs additional latency. Therefore, in subsequent online experiments we conduct an A/B test based on the model of "TaoType" and "TaoType w/o SUG" (see Subsection 6.2).

5.5 Devices

Since our model is designed for on-device deployment, we compared the performance differences between training on combined device data and training separately on each device type. As shown in Table 4, while separate training leads to an improvement in accuracy, it results in a decline in either recall or precision. In addition, to reduce engineering complexity in on-device deployment, we adopt a single model trained on combined data from both Android and iOS devices, which is then deployed independently on each platform.

5.6 Scaling and Stability

Using date T as the cutoff day, we train the model with historical data spanning 1 day, 2 days, 3 days, 1 week, 2 weeks, 3 weeks, and 1 month, respectively. We then evaluate the model's performance over the next three days, namely $T+1$, $T+2$, and $T+3$ (see Figure 3). We first examine the effect of training data duration. When the training period is extended to 1 week (1w), precision, recall and accuracy reach their peak values of 0.7936, 0.8196, and 0.7742, respectively. Further increasing the amount of training data leads to a decline in both precision and accuracy. Therefore we arrive at a conclusion that bears some resemblance to the scaling law (Kaplan et al., 2020): when the number of model parameters is limited, increasing the amount of training data does not necessarily lead to better performance. For dynamic intent prediction with compact models, historical data beyond one week often introduces distribution-shift noise. Recent high-quality data proves more effective than massive legacy datasets. Therefore, in the online experiments, the models were trained by historical data from one week (**1w**). Next, we turn to the stability of model prediction. For the three curves of the same color, one is not consistently above or below the others across consecutive days, and the curves remain tightly clustered. This indicates that the model produces stable predictions over time.

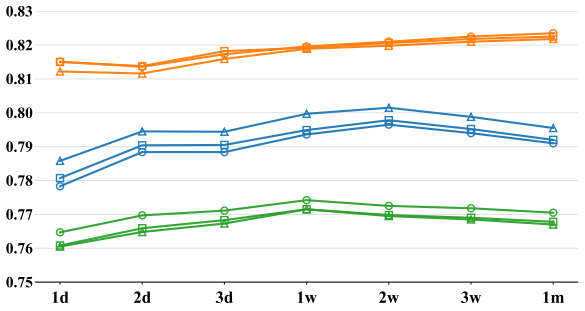
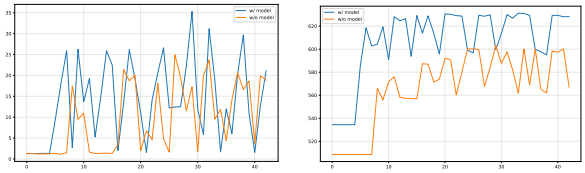


Figure 3: Data scalability of model training and stability of model inference. 1d indicates the training data is from date T, 2d is $[T-1, T]$, 3d is $[T-2, T]$, 1w is $[T-6, T]$, 2w is $[T-13, T]$, 3w is $[T-20, T]$, 1m is $[T-29, T]$. Orange: recall, blue: precision, green: accuracy. Circle ◯, square □ and triangle up △ indicate testing on date $T+1$, $T+2$, $T+3$, respectively.

6 Deployment and Online Testing

6.1 Implementation and Efficiency

We employ TaoType to product search on a mobile shopping app named Taobao. After each typing, the system uses TaoType to predict whether the user intends to search directly for the current query, and if so, it proactively initiates the search before the user explicitly submits it. TaoType is trained offline, and implemented through C++ within the client-side code.



(a) CPU utilization (%). (b) Memory usage (M).

Figure 4: CPU and memory change w/ and w/o TaoType on an iPhone 15 Pro.

Table 5: Suggestion delay.

Measure	Variable	Android	iOS
Statistics	Mean	291.68	279.99
	Median	215.00	220.00
	P95	432	449
	P99	1144	1153
Percentage	$<$100 ms	1.1	1.0
	$[100, 300)$ ms	82.9	81.4
	$[300, 1000)$ ms	14.8	16.4
	$>$1000 ms	1.2	1.2

Notably, TaoType does not affect the initial installation size of the app, as the model weights are only loaded onto the client side when the user begins typing search queries. Since search typing is a highly frequent operation that significantly impacts user experience, we conducted performance evaluation. TaoType has a file size of ~78 KB within the client side. The first inference, including model loading time, takes ~15 ms, while each subsequent inference takes ~0.5 ms. We then compared battery usage w/ and w/o TaoType, and found that the difference was negligible. This may be because TaoType accounts for only a very small portion of the overall battery consumption of the app. We further compared the change of CPU and memory on high-/mid-/low-end Android and iOS devices, respectively. The results show that CPU utilization increased by only +0.46%/+0.5%/+0.56% on Android and only +0.22%/+0.54%/+0.50% on iOS, while memory usage increased by +35/+40/+58 MB on Android and +30/+36/+45 MB on iOS. These results indicate that the additional performance overhead is minimal, and TaoType significantly reduces response time without degrading the user experience. Figure 4 shows the change of CPU and memory w/ and w/o TaoType on an iPhone 15 Pro. Note that the comparisons were conducted using realistic 5-minute user sessions (with or without preloading) from the same user on the same device during normal browsing.

6.2 Predicting Strategies

As stated in Subsection 5.4, there is a certain delay between the end of each typing and the arrival of SUG features. We conducted a statistical analysis of this delay, as summarized in Table 5. Most delays fall within $[100, 300)$ ms, which is non-negligible and must be considered when reducing RT. However, removing SUG features leads to a drop in performance. To balance performance and latency, we designed three online strategies. (1) TaoType-o1: prediction is performed immediately after each typing using the model of "TaoType w/o SUG", without waiting for SUG features to arrive. (2) TaoType-o2: prediction is made only after the SUG features are received, using the model of "TaoType". (3) TaoType-o3: an initial prediction is made right after each typing step using the model of "TaoType" with default values for SUG features; if the result indicates the query is not intent 2, a second prediction is performed once the SUG features become available. Additionally, we implemented two online baseline strategies for comparison. The first, called random-r1, applies equal random sampling between intent 2 and not intent 2 after each typing, with a ratio of 5:5. The second, called random-r2, uses ratios of 4:6 for the first typing, 2:8 for the second typing, and 1:9 for all subsequent typing; these three sampling ratios were derived from the training data of one week.

6.3 A/B Testing

By randomly dividing device IDs into five groups according to their OS (Android or iOS), we evaluated not only precision and recall but also the average reduction in response time ($\overline{RT}-$) and the corresponding increase in average QPS (queries per second) waste ($\overline{QPS}+$). The results of A/B testing over 6 days are shown in Table 6. (1) Compared with the offline testing results shown in Table 3, the model's online performance shows a decline. There are two main reasons. First, in a few cases, users issue search queries before the SUG features are returned. Second, there is a discrepancy between the online prediction and the offline evaluation, which leads to an increase in FP and FN. As a result, the online precision and recall decrease. (2) $\overline{QPS}$ waste is negatively correlated with precision. Note that in elastic cloud environments, marginal cost of handling minor QPS surges during off-peak hours is negligible. (3) TaoType-o3 outperforms random-r1 and random-r2 across all metrics, demonstrating

Table 6: Results of online A/B testing.

Strategy	Metric		$\overline{RT}(ms)\uparrow$	$\overline{QPS}(\%)\downarrow$
	Prec.↑	Recall↑		
random-r1	0.1892	0.4991	248.21±25.11	5.02±0.18
random-r2	0.4046	0.3863	432.14±31.87	4.82±0.40
TaoType-o1	0.6314	0.3847	516.49±34.51	3.01±0.26
TaoType-o2	0.6852	0.4221	235.59±12.41	2.52±0.28
TaoType-o3	0.5201	0.6462	640.51±93.65	4.19±0.22

the effectiveness of our framework. (4) A selection from o1, o2 and o3 can be determined from Table 6 by taking into account the target RT reduction, the tolerable increase in hardware costs caused by QPS waste, and the required user coverage as measured by recall. (5) The reduction in RT is substantial and can provides significant benefits to the ranking pipeline.

6.4 Case Study

We provide three cases to make the TaoType framework more intuitive. (1)[“电热” *(heater)*; 0.38]→[“电热水” *(hot water)*; 0.49]→[“电热水龙头” *(electric water heater)*; 0.84]. As new queies increase the prediction score to $0.84 \geq 0.5$, so presearch is initiated. (2)[“海尔电热水龙头” *(Haier electric water heater)*; -]→[“电热水龙头” *(electric water heater)*; 0.43]→[“美的电热水龙头” *(Midea electric water heater)*; 0.86]. The brand replacement from “海尔” to “美的” increases the prediction score to $0.86 \geq 0.5$, triggering presearch. (3)[(“袜子” *(socks)*); 0.46]→[“袜子” *(socks)*, score_tp1: 0.032; 0.79]. After adding the SUG feature “score_tp1: 0.032”, the prediction score increases from 0.46 to 0.79, enabling presearch. This is because the click score of the top suggestion is too low, leading TaoType to infer that the user is unlikely to click on any suggestion.

7 App Intelligence

At the intersection of mobile applications and the ascent of AI, we propose a new intelligence paradigm termed “App Intelligence”. This concept refers to enhancing an app’s native AI capabilities to improve both business outcomes and user experience. Our work, TaoType, serves as a representative case study. Synthesizing insights from related work, we identify three key features of App Intelligence: (1) it delivers measurable business value or enhances user experience; (2) the core intelligent functionality operates primarily within the app itself; (3) the AI solutions are adaptable, ranging from rules and strategies to advanced machine learning and LLMs, selected according to the specific needs of the application and its scenarios. We also discuss the key distinctions between App Intelligence and similar concepts to position our contribution clearly:

- Web intelligence - It is on the web, while ours is within the app.
- Federated learning - It focuses on cloud-device collaboration and operates mainly on the server-side, while ours operates mainly on the client-side.
- TinyML - It is one of the AI solutions to achieve App Intelligence.
- MobileLLM - It is also one of the AI solutions to achieve App Intelligence.
- Edge AI - It is on-device, while ours is within an app with lower permission requirements, less GPU consumption, and reduced memory and storage demands. App Intelligence is a subset of edge AI.

8 Conclusions

With the escalating volume of searchable items and the growing sophistication of search technologies powered by LLMs, minimizing RT has become increasingly important. We propose a novel approach to minimize RT by leveraging client-side algorithms to predict whether users will initiate a search directly, enabling proactive loading and thereby decreasing user waiting time. We developed a lightweight and efficient client-side intent prediction framework TaoType. It demonstrates a high offline classification performance with precision/recall/accuracy of 0.7936/0.8196/0.7742. Results from online A/B testing show a significant RT decrease of 640.51±93.65 ms, which is highly beneficial to the search system. Importantly, TaoType operates without compromising app performance, while the resultant increase in QPS due to prefetching remains within acceptable bounds. Our work is a case study of the newly introduced “App Intelligence”. We detail its primary features and its distinctions from similar concepts. In future research, we aim to advance the field of App Intelligence by enhancing application-level AI capabilities, ultimately establishing a robust and user-friendly AI interface between users and applications.

Limitations

Although the system supports various languages (e.g., English, Japanese, and Arabic) and employs language-agnostic features within the TaoType framework, our current user base and data are predominantly Chinese. Therefore, the results may not be fully representative of user behaviors in other language environments.

Ethical Considerations

Our method utilizes log datasets without sensitive user information. Moreover, practical deployment should guarantee data privacy while ensuring that the system does not degrade user experience on lower-end hardware.

Acknowledgments

We sincerely thank Zhen Yang, Mingming Pan, Chaoxu Sui, and Jie Fang for their expertise and time in the engineering deployment and online experiments.

References

Zhuojiang Cai, Jingkai Hong, Zhimin Wang, and Feng Lu. 2025. Gazeswipe: Enhancing mobile touchscreen reachability through seamless gaze and finger-swipe integration. In *Proceedings of the 2025 CHI Conference on Human Factors in Computing Systems*.

Anna Veronika Dorogush, Vasily Ershov, and Andrey Gulin. 2018. Catboost: gradient boosting with categorical features support. *arXiv preprint arXiv:1810.11363*.

Lu Fan, Qimai Li, Bo Liu, Xiao-Ming Wu, Xiaotong Zhang, Fuyu Lv, Guli Lin, Sen Li, Taiwei Jin, and Keping Yang. 2022. Modeling user behavior with graph convolution for personalized product search. In *Proceedings of the ACM Web Conference 2022*, page 203–212.

Long Guo, Lifeng Hua, Rongfei Jia, Fei Fang, Binqiang Zhao, and Bin Cui. 2020. Edgedipn: a unified deep intent prediction network deployed at the edge. *Proc. VLDB Endow.*, 14(3):320–328.

Long Guo, Lifeng Hua, Rongfei Jia, Binqiang Zhao, Xiaobo Wang, and Bin Cui. 2019. Buying or browsing?: Predicting real-time purchasing intent using attention-based deep network with multiple behavior. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, page 1984–1992.

Qi Guo and Eugene Agichtein. 2010. Ready to buy or just browsing? detecting web searcher goals from interaction data. In *Proceedings of the 33rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, page 130–137.

Qi Guo, Eugene Agichtein, Charles LA Clarke, and Azin Ashkan. 2009. In the mood to click? towards inferring receptiveness to search advertising. In *2009 IEEE/WIC/ACM International Joint Conference on Web Intelligence and Intelligent Agent Technology*, volume 1, pages 319–324.

Jyun-Yu Jiang, Yen-Yu Ke, Pao-Yu Chien, and Pu-Jen Cheng. 2014. Learning user reformulation behavior for query auto-completion. In *Proceedings of the 37th International ACM SIGIR Conference on Research & Development in Information Retrieval*, page 445–454.

Xiaoqi Jiao, Yichun Yin, Lifeng Shang, Xin Jiang, Xiao Chen, Linlin Li, Fang Wang, and Qun Liu. 2020. TinyBERT: Distilling BERT for natural language understanding. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4163–4174.

Anil Kag, Igor Fedorov, Aditya Gangrade, Paul Whatmough, and Venkatesh Saligrama. 2022. Efficient edge inference by selective query. In *The Eleventh International Conference on Learning Representations*.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. 2020. Scaling laws for neural language models. *Preprint*, arXiv:2001.08361.

Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. 2017. Lightgbm: A highly efficient gradient boosting decision tree. In *Advances in Neural Information Processing Systems*, volume 30.

John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the Eighteenth International Conference on Machine Learning*, page 282–289.

Sen Li, Fuyu Lv, Taiwei Jin, Guiyang Li, Yukun Zheng, Tao Zhuang, Qingwen Liu, Xiaoyi Zeng, James Kwok, and Qianli Ma. 2022. Query rewriting in taobao search. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*, page 3262–3271.

Wenqi Li, Jui-Ching Kuo, Manyu Sheng, Pengyi Zhang, and Qunfang Wu. 2025. Beyond explicit and implicit: How users provide feedback to shape personalized recommendation content. In *Proceedings of the 2025 CHI Conference on Human Factors in Computing Systems*, pages 1–17.

Ji Lin, Ligeng Zhu, Wei-Ming Chen, Wei-Chen Wang, and Song Han. 2023. Tiny machine learning:

Progress and futures. *IEEE Circuits and Systems Magazine*, 23(3):8–34.

Xufeng Qian, Yue Xu, Fuyu Lv, Shengyu Zhang, Ziwen Jiang, Qingwen Liu, Xiaoyi Zeng, Tat-Seng Chua, and Fei Wu. 2022. Intelligent request strategy design in recommender system. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 3772–3782.

Chufeng Shi, Yangsu Liu, Rui Qiu, Zhenzhe Zheng, Dagui Chen, Ruitao Zhu, and Fan Wu. 2025. Comrecycle: An intelligent computation recycling framework for online advertising. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V.2*, page 4851–4860.

Xianyang Tian, Xiang Xu, Chao Wang, and Others. 2025. Towards explainable search results in e-commerce. In *Companion Proceedings of the ACM on Web Conference*, page 476–484.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Zhuoxing Wei, Qi Liu, and Qingchen Xie. 2025. Deep multiple quantization network on long behavior sequence for click-through rate prediction. In *Proceedings of the 48th International ACM SIGIR Conference on Research and Development in Information Retrieval*, page 3090–3094.

Liang Yao, Chengsheng Mao, and Yuan Luo. 2019. Graph convolutional networks for text classification. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33(01):7370–7377.

Jiaqi Zhai, Lucy Liao, Xing Liu, Yueming Wang, Rui Li, Xuan Cao, Leon Gao, Zhaojie Gong, Fangda Gu, Jiayuan He, Yinghai Lu, and Yu Shi. 2024. Actions speak louder than words: trillion-parameter sequential transducers for generative recommendations. In *Proceedings of the 41st International Conference on Machine Learning*.

Bin Zhang, Weipeng Huang, Dimin Wang, Jialin Zhu, Yuning Jiang, Zhaode Wang, Chengfei Lv, Jian Wang, Qichao Ma, Li Chen, Junqing Wu, and Yipeng Yu. 2026. Recgpt-mobile: On-device large language models for user intent understanding in taobao feed recommendation. In *Proceedings of the 49th International ACM SIGIR Conference on Research and Development in Information Retrieval*.

Shiquan Zhang, Ying Ma, Le Fang, Hong Jia, Simon D'Alfonso, and Vassilis Kostakos. 2024. Enabling on-device llms personalization with smartphone sensing. In *Companion of the 2024 on ACM International Joint Conference on Pervasive and Ubiquitous Computing*, page 186–190.

Appendix

A Configuration of baseline models

- LightGBM: 100 trees, num_leaves=32;
- CatBoost: 100 iterations, depth=6;
- ANN: Hidden 128, Hidden 64;
- CNN: InputDim 11, Hidden 32×3, K 3;
- BiLSTM: InputDim 11, Hidden 128;
- GCN: InputDim 20, Hidden 64×2.