DuIVA: An Intelligent Voice Assistant for Hands-free and Eyes-free Voice Interaction with the Baidu Maps App

Jizhou Huang*
Haifeng Wang
huangjizhou01@baidu.com
wanghaifeng@baidu.com
Baidu Inc.
Haidian District, Beijing, China

Shiqiang Ding
Shaolei Wang
dingshiqiang01@baidu.com
wangshaolei@baidu.com
Baidu Inc.
Haidian District, Beijing, China

ABSTRACT

Mobile map apps such as the Baidu Maps app have become a ubiquitous and essential tool for users to find optimal routes and get turn-by-turn navigation services while driving. However, interacting with such apps while driving through visual-manual interaction modality inevitably causes driver distraction, due to the highly conspicuous nature of the time-sharing, multi-tasking behavior of the driver. In this paper, we present our efforts and findings of a 4-year longitudinal study on designing and implementing DuIVA, which is an intelligent voice assistant (IVA) embedded in the Baidu Maps app for hands-free, eyes-free human-to-app interaction in a fully voice-controlled manner. Specifically, DuIVA is designed to enable users to control the functionalities of Baidu Maps (e.g., navigation and location search) through voice interaction, rather than visual-manual interaction, which minimizes driver distraction and promotes safe driving by allowing the driver to keep "eyes on the road and hands on the wheel" while interacting with the Baidu Maps app. DuIVA has already been deployed in production at Baidu Maps since November 2017, which facilitates a better interaction modality with the Baidu Maps app and improves the accessibility and usability of the app by providing users with in-app voice activation, natural language queries, and multi-round dialogue. As of December 31, 2021, over 530 million users have used DuIVA, which demonstrates that DuIVA is an industrial-grade and production-proven solution for in-app intelligent voice assistants.

CCS CONCEPTS

• **Computing methodologies** → **Natural language processing**.

KEYWORDS

Intelligent voice assistant, voice interaction, hands-free, eyes-free, user-to-app interaction, task-oriented dialogue, Baidu Maps

*Corresponding author: Jizhou Huang.

KDD '22, August 14–18, 2022, Washington, DC, USA

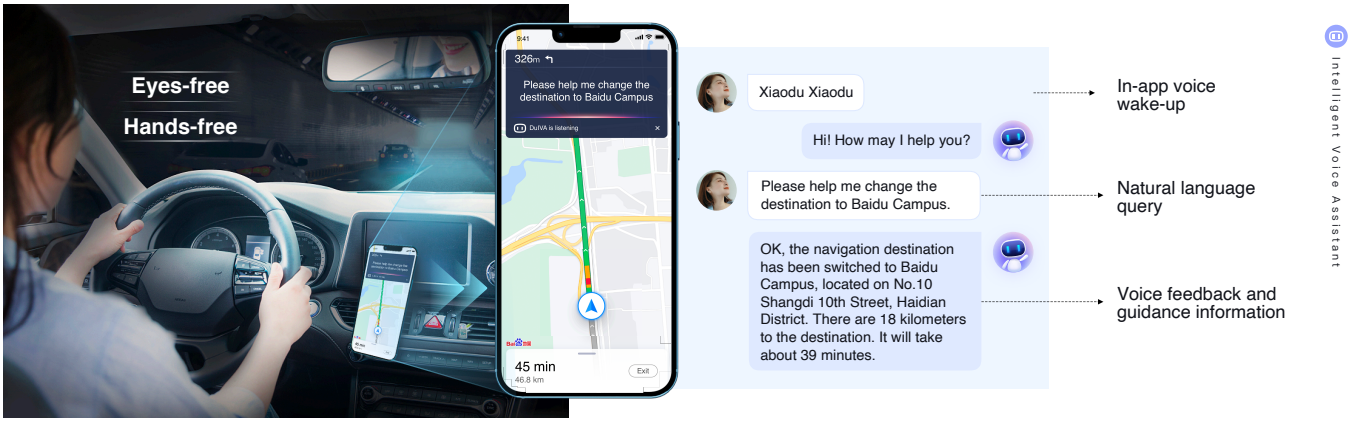
ACM ISBN 978-1-4503-9385-0/22/08…$15.00
https://doi.org/10.1145/3534678.3539030

ACM Reference Format:
Jizhou Huang, Haifeng Wang, Shiqiang Ding, and Shaolei Wang. 2022. DuIVA: An Intelligent Voice Assistant for Hands-free and Eyes-free Voice Interaction with the Baidu Maps App. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '22), August 14–18, 2022, Washington, DC, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3534678.3539030

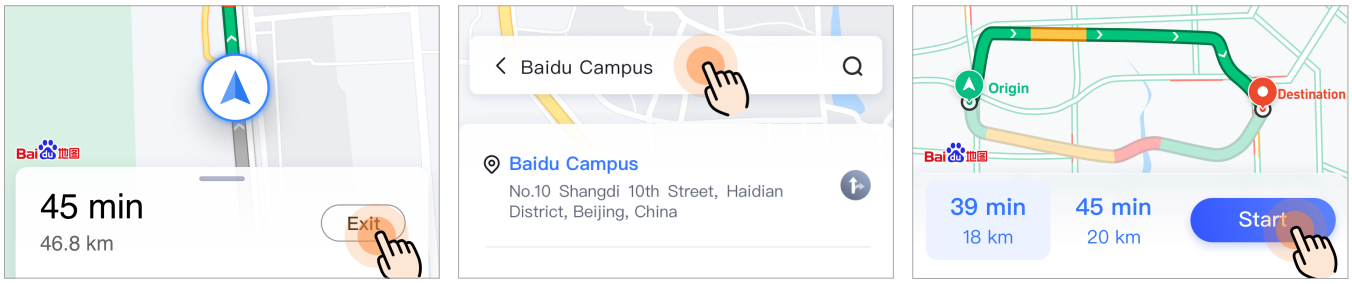
1 INTRODUCTION

Mobile map apps have become a ubiquitous and essential tool for users to find optimal routes and get turn-by-turn navigation services. The mainstream modality to interact with the apps provided on smartphones is visual-manual interaction, which is generally performed through hand-held operations and multi-touch gesture controls on the graphical user interface (GUI). However, interacting with map apps while driving through visual-manual interaction inevitably causes driver distraction, due to the highly conspicuous nature of the time-sharing, multi-tasking behavior of the driver. Quantitative results from a naturalistic driving study have shown that performing visual-manual tasks on cell phones while driving would significantly degrade driver performance and increase safety-critical event risk [6]. Therefore, in order to minimize driver distraction and promote safe driving, it is important to enable users to interact with map apps in a completely hands-free and eyes-free manner. One optimal modality in this context is voice interaction based on voice user interface (VUI) [15], which attempts to control map apps in a fully voice-controlled manner.

The significance of enabling a voice interaction modality within map apps as a supplementary interface could be drawn from the four observations in our usability evaluation of map apps. (1) It is not easy to input point of interest (POI) names into map apps on smartphones by finger-based typing or handwriting with on-screen keyboards, since POI names are often rare or newly-coined words. The constraints of such text entry methods lead to inefficiency for location search, and they even increase driver distraction due to time-sharing behavior when performing finger-based tasks while driving. (2) While navigating to a destination, making adjustments to map apps through visual-manual interaction requires users to switch between driving and the adjustment task (e.g., mute voice guidance and change destinations on the fly), which inevitably causes driver distraction. (3) Only the high-frequency features are presented graphically on the user interface, due to the limited space for display on smartphones, making it difficult for users to access the underexposed features, such as turning on a dark theme. (4) There lacks a quick way or shortcut to accomplish a task that



(a) Hands-free and eyes-free control of the Baidu Maps app for changing the destination while driving through voice interaction.



(b) The major steps involved when changing the destination through visual-manual interaction.

Figure 1: Comparison between voice interaction and visual-manual interaction for changing the destination.

involves multiple steps and a series of visual-manual interactions with map apps. For example, the task of finding the quickest route to a destination with multiple stops is made up of a tedious process exhibiting at least these interactions: find an optimal route, search places along the route, add waypoints, and navigate.

In this paper, we present our efforts and findings of a 4-year longitudinal study on designing and implementing DuIVA, which is an intelligent voice assistant (IVA) embedded in the Baidu Maps app for hands-free, eyes-free human-to-app interaction in a fully voice-controlled manner. Specifically, DuIVA is designed to enable users to control the functionalities of Baidu Maps through voice interaction rather than visual-manual interaction, which allows the driver to keep "eyes on the road and hands on the wheel" while interacting with the app. To enable users to command and converse with the Baidu Maps app using natural language, we have developed over 860 voice-invocable skills with voice-based commands, which can be categorized into five major categories: navigation, location search, information inquiry, app settings, and shortcut. DuIVA facilitates a better interaction modality with the Baidu Maps app and improves user experience by providing users with in-app voice activation, natural language queries, and multi-round dialogue.

Take the task of changing the destination while driving as an example. Figure 1 shows a comparison between voice interaction (see Figure 1a) and visual-manual interaction (see Figure 1b) for accomplishing this task using the Baidu Maps app. The task completion time and the allocation of visual attention differ significantly between the two interaction modalities. Specifically, DuIVA enables hands-free and eyes-free interaction with the app by just speaking the command "change the destination to Baidu Campus" after the voice activation, which takes only a few seconds. On the contrary, accomplishing this task with visual-manual interaction necessitates visual attention sharing and hand movements, which typically involves a longer timescale of minutes for drivers (see §4.3 for a more detailed comparison of interaction efficiency).

Our main contributions can be summarized as follows:

- **Potential impact:** We suggest a production-proven and industrial-grade solution for building an in-app intelligent voice assistant that enables users to interact with map apps in a completely hands-free and eyes-free manner. We document our efforts and findings of a 4-year longitudinal study on designing and developing DuIVA. We hope that it could be a stepping stone to more intelligent assistant generalizations.
- **Novelty:** On the one hand, the design and development of DuIVA are driven by the novel idea that controls the functionalities of map apps during driving through complete voice interaction, rather than visual-manual interaction, which can significantly minimize driver distraction. On the other hand, to the best of our knowledge, this is the first attempt to present a production-proven, industrial-grade solution for voice interaction with mobile map apps via an in-app intelligent voice assistant. DuIVA has already been deployed in production at Baidu Maps since November 2017. As of December 31, 2021, over 860 skills are developed, and over 530 million users have used DuIVA with a customer satisfaction score of 91.9%.
- **Technical quality:** Quantitative experiments, conducted on large-scale real-world datasets, demonstrate that the amount of time and effort required to accomplish driver-to-app interaction during driving is greatly reduced.

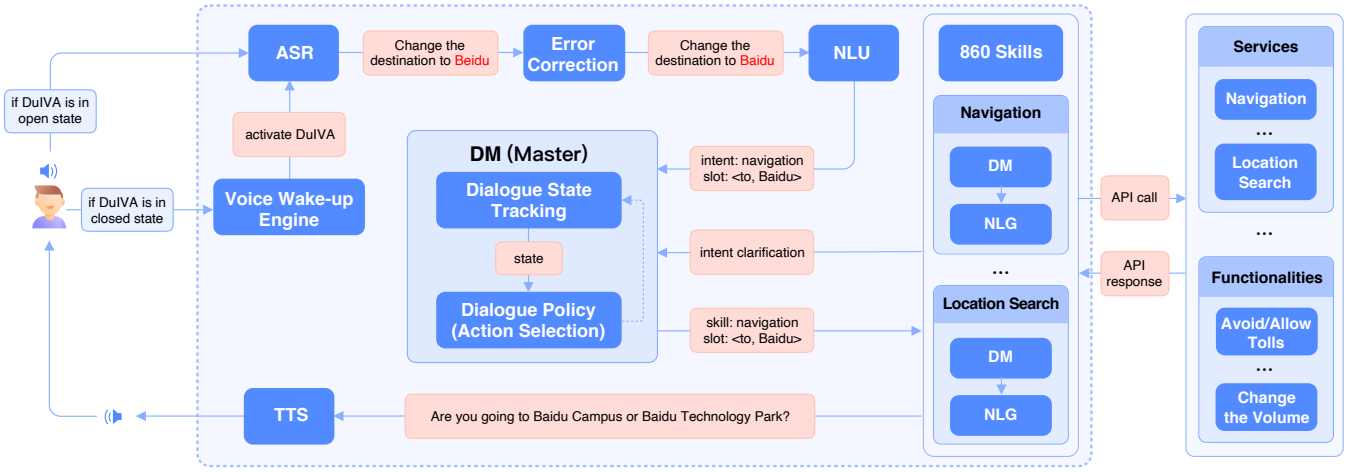


Figure 2: Overall architecture of DuIVA.

2 DuIVA

In this section, we present the design and implementation of DuIVA for hands-free, eyes-free human-to-app interaction in a fully voice-controlled manner.

2.1 Background

Compared with the visual and manual cognitive resources of drivers, the auditory cognitive resource is less strained [29]. For this reason, voice user interfaces have become indispensable for automotive users [21–23], which enable them to make distraction-free voice interactions with considerable efficiency. Intelligent voice assistant (IVA), enabling drivers to conduct tasks through voice interface and natural voice interactions, has shown to be a reliable means that can significantly reduce cognitive load of drivers [16, 17, 31].

It is highly challenging to build an in-app IVA system that characterizes robust stability and industrial-grade reliability, mainly because of three facts: (1) user expressions can be of high lexical and syntactic diversity for a same meaning in real scenarios, which makes it difficult to understand user intents; (2) the massive and ever-increasing dialogue skills place a huge demand on the scalability and robustness of the system; and (3) the complex acoustic environment of the moving vehicle can always result in channel distortion and non-stationary noise presence, which poses a huge challenge to spoken language understanding. To solve these challenges and build an industrial-grade in-app IVA system, we adopt a modular framework that decomposes the whole system into eight cascaded components, which benefits from the recent advances in IVA systems [27] and task-oriented dialogue systems [2, 8, 32].

2.2 Overview

The overall architecture of DuIVA is shown in Figure 2. It consists of eight components in sequence: (1) the voice wake-up engine (VWE) component, which monitors a stream of audio for the special wake-word "xiaodu xiaodu" and activates DuIVA upon detecting it; (2) the automatic speech recognition (ASR) component, which transcribes the voice query into text automatically; (3) the error correction (EC) component, which recovers the clean text from the erroneous text; (4) the natural language understanding (NLU) component, which parses the clean text from EC into a machine-readable structured semantic representation including user intents and slot values; (5) the dialogue management (DM) component, which keeps track of the dialogue state and selects a dialogue skill to respond to the query; (6) the skill component, which is designed to help the user complete a specific task; (7) the service/functionality component, which is requested to fulfill user intent when a skill is activated; and (8) the text to speech (TTS) component, which synthesizes natural-sounding speech from the generated responses of the skill component. DuIVA also supports multi-round dialogue with users, which enables them to interact with the Baidu Maps app in a completely hands-free and eyes-free manner.

2.3 Voice Wake-up Engine

2.3.1 Problem Statement. VWE aims at detecting the special wake-word "xiaodu xiaodu" from environmental audio by deploying a sophisticated ASR engine to run continuously in the background. At runtime, voice-initiated devices (e.g., smartphone) begin to stream user speech to DuIVA when VWE successfully detects "xiaodu xiaodu". The development of VWE for DuIVA is challenged by five major problems. (1) *Limited computation resources.* VWE is deployed locally on battery-powered devices such as tablets and smartphones, to keep listening, which requires that it must be resource and power efficient. (2) *Sound quality.* The sound quality of output signals varies greatly among hand-held devices. (3) *Environmental noise.* The interaction environment may be full of noise, such as car stereo and tyre noise, which inevitably affects the performance of VWE. (4) *Accurate and fast detection.* VWE is the gateway between the user and DuIVA, and thus, accurate and fast detection of "xiaodu xiaodu" is critical to improve user experience. (5) *Low latency.* The low latency requirement of real-time speech interaction necessitates processing the speech signals as rapidly as possible.

2.3.2 Our Practice. We address these challenges by: (1) adopting depthwise separable convolutions to reduce computation and model size of VWE; (2) compressing the speech features with variable lengths into a fixed-size representation to reduce power and computational resource consumption; and (3) applying data augmentation techniques to reduce the false acceptance rate.

First, to efficiently achieve real-time detection of the wake-word on local devices that have limited computational capability and power capacity, we use depthwise separable convolutions (DSC) [9] to encode the speech signals. Specifically, DSC factorizes a standard convolution into a separate layer for filtering and a separate layer for combining. By tuning the two key hyper-parameters of width multiplier and resolution multiplier, DSC can reduce computational cost and the number of parameters quadratically, which enables a smaller and faster VWE by trading off a reasonable amount of accuracy to reduce both model size and latency.

Second, we further adopt the structured self-attention mechanism (SSAM) [19] to compress the speech features. Specifically, SSAM performs multiple hops of attention to convert the speech features with variable lengths into a fixed-size representation. In this way, the power and computational resource consumption of VWE can be dramatically reduced.

Third, to reduce the false acceptance rate, we adopt negative sampling to construct negative instances for augmenting the training set. Specifically, we sample large-scale speech fragments with similar pronunciations to "xiaodu xiaodu" from the massive audio stream as the negative instances, and then incorporate them into the initial training set to improve the robustness and performance of VWE. In addition, we utilize focal loss [18] to focus training on hard examples, and to prevent the vast number of easy negative instances from overwhelming the training procedure. Our practical results demonstrate that the combination of negative sampling and focal loss can significantly reduce the false acceptance rate of VWE.

2.4 Automatic Speech Recognition

2.4.1 Problem Statement. ASR aims at automatically transcribing the speech signals into text. Compared to general-purpose applications, the development of ASR for DuIVA is challenged by five problems. (1) *Chinese accents.* DuIVA has attracted over 530 million users from all over China, among which many Chinese people speak non-standard varieties of Mandarin with regional accents ("Difang Putonghua" in Chinese), such as Shanghai, Sichuan, Guandong, and Fujian, which inevitably results in possible mispronunciations and accented speech. (2) *Domain-specific words.* Many domain-specific words (e.g., business name and address) involve rare, non-idiomatic, or code-switched proper nouns and acronyms, which poses a major challenge to the generic ASR systems. (3)-(5) *Sound quality*, *environmental noise*, and *low latency* (see §2.3 for details).

2.4.2 Our Practice. The generic ASR systems built at Baidu have achieved significant improvements in accuracy and latency. However, the recognition performance drops when deploying them in DuIVA due to the combination of the above-mentioned challenges. Hence, we have further improved ASR accuracy for DuIVA by leveraging a geographic-enhanced language model [25]. In addition, we use the mechanisms of both self-training and active learning with feedback from successes and failures (see §3 for details) to iteratively improve the performance of the ASR model.

2.5 Error Correction

2.5.1 Problem Statement. In real-world scenarios of DuIVA, ASR errors can be ubiquitous due to poor articulation and acoustic variability caused by various factors, such as environmental noise,

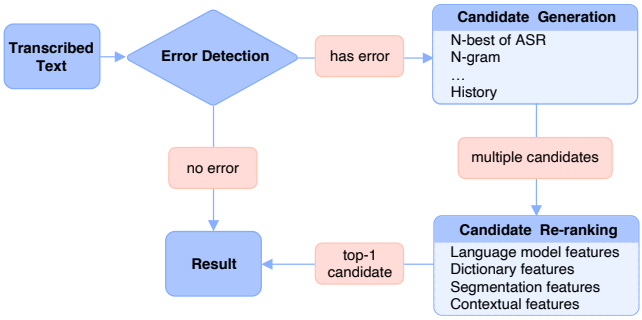


Figure 3: Overview of the error correction component.

Chinese accents, and poor sound quality. Our statistical analysis shows that the largest proportion of ASR errors involved homophone words or words with similar pronunciation, which is also a special phenomenon in Chinese ASR systems. To mitigate the impact of ASR errors on the downstream NLU component, we introduce an EC component to detect and correct errors in the output of ASR. The development of EC for DuIVA is challenged by two major problems. (1) *Homonyms in Chinese.* Errors related to homonyms account for a large proportion of errors in DuIVA, which are difficult to correct in practice. For example, although the two Chinese words "凤城" and "丰城" share the same Chinese Pinyin "feng cheng", they are two distinct cities located in different provinces of China. To accurately recover the city that a user would eventually visit, it necessitates full context and background knowledge (e.g., "where the query was issued") for the EC component. (2) *Newly-coined words.* Newly-coined words, such as business names and entities, emerge endlessly, which inevitably lead to the ubiquity of out-of-vocabulary (OOV) words and thus a loss of accuracy for error correction. To alleviate this problem, it necessitates consistently updating the EC component to better deal with OOV words.

2.5.2 Our Practice. To address the two challenges, we develop an effective re-ranking solution for EC, which explicitly integrates additional information that has not yet been considered by the ASR system into the decoding process of EC. As shown by Figure 3, the proposed solution is divided into three major steps.

First, an error detection module is used to determine whether the transcribed text is accurate or not. Specifically, it uses a multi-layer bidirectional transformer encoder to capture contextual features of the transcribed text, and then makes predictions with a CRF layer. In addition, we conduct a series of refinements to improve the performance and robustness of this module, including: (1) embedding Chinese characters and Pinyin symbols in a shared space [34]; (2) adding lexical and syntactic features (i.e., word boundaries, POS tags, and dependency labels) to the CRF layer; and (3) introducing a pre-training task that masks characters with Chinese Pinyin or similar pronounced characters [34], and then pre-training the model on a large collection of Chinese sentences. The text will be sent to the next candidate generation module if some errors are detected, and sent to the NLU component otherwise.

Second, the candidate generation module is used to generate correction candidates. To improve the performance, we integrate multiple error correction methods including: (1) integrating the n-best list of candidates by the ASR system as candidates; (2) retrieving a user's query history and then selecting the queries that are

similar to the output of the ASR system as candidates; (3) using an n-gram language model [1]; (4) adopting an extended HMM-based approach [30]; (5) adopting a pre-trained sequence-to-sequence model; and (6) adopting the pre-trained models of ERNIE-GeoL [13] and ERNIE [26]. The ensemble of various methods can integrate complementary information into the process of candidate generation, which can improve the upper bound of error correction.

Third, the candidate re-ranking module is used to re-rank the candidates by introducing more rich features. Specifically, we adopt a learning to rank model to rank the candidates in accordance with four categories of features. (1) Language model features, which include the language model scores coming from an n-gram language model, a pre-trained language model, and a geographic-enhanced language model [25]. (2) Dictionary features, which include the number, the length, and the proportion of phrases in Chinese word segmentation results of each candidate. (3) Segmentation features, which are the Chinese word segmentation results of each candidate. (4) Contextual features, which contain in-session queries and the city where the query was issued. Finally, the candidate with the highest ranking score is considered as the correction result.

2.6 Natural Language Understanding

2.6.1 Problem Statement. NLU aims at understanding a user's intent, which is a core component of DuIVA. It typically consists of two subtasks: intent detection and slot filling [28]. For example, given an utterance "change the destination to Baidu Campus", the output of NLU consists of a user intent "navigation" and a slot "<to, Baidu Campus>". The development of NLU for DuIVA is challenged by three major problems. (1) *ASR errors.* ASR errors cannot be escaped even after error correction, which will be propagated and inevitably have effects downstream. (2) *Diverse expressions.* User expressions can be of high lexical and syntactic diversity for the same meaning, which pose a huge challenge to the generalization of NLU. (3) *Newly emerging intents and slots.* In real-world scenarios of DuIVA, the number of intents and slots keeps increasing along with the increasing of new skills, which necessitates regularly updating the NLU component.

2.6.2 Our Practice. We address the above-mentioned challenges through a series of refinements, including: (1) adopting the template-based method as our primary tool for intent detection and slot filling; (2) using the neural network model as our supplementary tool; and (3) applying few-shot learning to address the problems of long-tail and newly emerging intents and slots.

First, an intelligent voice assistant that characterizes robust stability and industrial-grade reliability necessitates detecting user intents and filling the slots as accurately as possible. To this end, we adopt the template-based method (see Appendix A.1 for more details) as our primary tool for joint intent detection and slot filling, which has the advantages of high accuracy and interpretability with low computational cost.

Second, although the template-based method can achieve high accuracy, the coverage is insufficient because it is difficult to cover the less frequently used expressions. To increase the coverage, we further develop a neural network model for joint intent detection and slot filling as our supplementary tool. Specifically, we use a transformer encoder network initialized with ERNIE [26] to encode the query. Then, we apply a CRF layer to predict each slot, and a feed-forward layer to predict the user intent. Our practice suggests that the template-based method and the neural model are complementary: the former provides high accuracy and interpretability, and meanwhile the latter offers high coverage and generalization.

Third, the neural network model inevitably suffers from insufficient training examples when dealing with the long-tail intents (<100 visits per day). To address this problem, we adopt few-shot learning to improve the generalization ability of NLU. Specifically, we develop a few-shot joint intent detection and slot filling system based on nearest neighbor learning and structured inference [33]. The few-shot system provides us with a viable alternative for dealing with the long-tail intents, and its role is becoming increasingly important as its performance keeps improving.

In our practice, we deploy the three NLU models via a cascade manner to perform NLU, which enables a controllable trade-off between accuracy, interpretability, generalization, and computational cost. Specifically, the query is first sent to the template-based model. If it fails, the query is then sent to the neural network model. If both fail, the few-shot learning model is adopted as the last resort. To evaluate the query coverage per model, we monitored logs of DuIVA from a 1-month period of December 2021. Statistics show that 93.7%, 5.2%, and 1.1% of the total traffic are handled by the template-based model, the neural network model, and the few-shot learning model, respectively.

2.7 Dialogue Management

2.7.1 Problem Statement. DM is the central controller of DuIVA. It consists of the *dialogue state tracking*, which is responsible for keeping track of the current dialogue state s, and *dialogue policy* π, which selects an action based on the dialogue state as $a = \pi(s)$. Each action corresponds to a skill that is triggered to help a user complete a specific task. The development of DM for DuIVA is challenged by two problems. (1) *Massive action space.* The over 860 skills result in a massive action space, which not only causes difficulty for action selection, but also makes the complexity of DM increase. (2) *Scalability.* The skills keep increasing during the evolution of DuIVA, which necessitates a strong scalability of DM.

2.7.2 Our practice. To better handle the massive skills, and to improve the generalization of DuIVA when adding new skills, we decouple the invocation of skills from their specific implementations by mathematically casting the dialogue controller as a hierarchical decision-making process. As shown in Figure 2, the master DM is mainly responsible for managing the overall conversation by selecting, at each dialogue turn, a skill to handle different types of conversation segments. Each skill is designed to handle a certain task (e.g., navigation and location search) and manage a specific type of conversation segment (see §2.8 for details).

Dialogue state tracking. This module maintains and updates the dialogue state according to the observations perceived during the dialogue. In practice, DuIVA creates and maintains a working memory to keep track of the dialogue state $s = (Q, C, H, K)$ for individual dialogue sessions, where Q represents the results of NLU, C represents the context information such as the current service in use by a user (e.g., navigation or location search), H represents the conversation history, and K represents the skill that was triggered in the last

dialogue turn. The contents in the working memory are encoded into a dialogue state vector, and are then fed to the dialogue policy module to predict the next skill s_t. Once s_t is executed, the working memory is updated according to the feedback of s_t.

Dialogue policy. This module is implemented to predict the next skill from a set of skill triggers, which consists of two consecutive steps. First, we design abundant heuristic rules to decide whether or not to switch topics or skills at each dialogue turn. For example, as shown in Figure 2, DuIVA generates a response of intent clarification ("Are you going to Baidu Campus or Baidu Technology Park?") based on the current query ("Change the destination to Baidu"), and expects the user to provide an answer with binary selection. If the user provides a query with one of the two suggested locations at the next dialogue turn, the current skill will remain active. Otherwise, a new skill is activated by a skill switching engine. In practice, we build a classifier to implement the skill switching engine, which is an ensemble model that uses both a boosted tree and a neural network model with diverse features from the dialogue state. By decoupling the invocation of skills from their specific implementations, it only requires to make adjustments to the skill switching engine when launching new skills into DuIVA.

In the case of hands-free and eyes-free voice interaction, the accuracy is a key point, since it relates with reliability, usability, and efficiency of DuIVA. Our usability evaluation reveals that misinterpretations of user intents are highly correlated with inconvenience and dissatisfaction for users. To minimize misinterpretation, a series of strategies have been designed and used in DM. Here we highlight two representative strategies. (1) If the confidence score of DuIVA is below a certain threshold, a feedback mechanism will be triggered to generate a kindly reminder to suggest the user to refine the original query. (2) For queries that need to be provided with risk-free decision-making opportunities (e.g., change the destination while driving), a double-check mechanism is introduced, which generates a voice prompt or a confirmation question using the interpretation results. This offers the user an opportunity to accept the current interpretation or revise it. Statistics show that 2.98% of interpretations made by DuIVA were subsequently revised by users over the one-year period ending in December 2021.

2.8 DuIVA Skills

2.8.1 Problem Statement. The skills are designed to handle different types of services and functionalities, and each skill only deals with a specific and fundamental function. Specifically, each skill consists of two components: (1) a lightweight DM component, which is used to manage the fulfillment of a service or a functionality; and (2) a lightweight natural language generation (NLG) component, which is used to generate responses that are grammatically fluent, unambiguous, and easy to understand for users. The development of skills for DuIVA is challenged by three major problems. (1) *NLU errors.* NLU errors cannot be escaped even by combining multiple NLU models, which will be propagated and inevitably have an adverse impact on DM. (2) *Vague or incomplete queries.* Due to the uncertainty of these kinds of queries, it inherently suffers from the lack of solid evidence on the understanding of user intents behind them. This necessitates clarifying and reasoning about such queries as accurately as possible. (3) *Ever-increasing skills.* As new functionalities are being developed over time in Baidu Maps, more skills need to be implemented consistently for DuIVA. For example, there strongly emerges the new skill "find nearby charging piles" as electric vehicle penetration grows.

2.8.2 Our practice. We address the above-mentioned challenges through a series of refinements, including: (1) adopting the template-based method to achieve a desired trade-off between performance and development effort (e.g., R&D cycle time and deployment cost) of massive skills; (2) integrating clarification and reasoning strategies to handle the problems of NLU errors and vague queries; and (3) building an effective mechanism to consistently uncover new intents and develop new skills from the failed queries of DuIVA.

First, the DM and NLG components of each skill are required to realize high precision. Moreover, due to the diversity of skills and the ever-increasing skills, it is impractical to label large-scale training data for individual skills. In view of such constraints and the fact that each skill only deals with a specific task, we use the template-based method to build the DM and NLG components for each skill. Specifically, we adopt the finite state model [7] to implement the DM, which explicitly enumerates all possible dialogue states and allowable transitions between different states. In addition, to increase the diversity of DuIVA responses, we build multiple templates for each state or action. In this way, DuIVA can achieve promising performance with minimum dependence on labeled data and model training when adding a new skill.

Second, to handle the NLU errors and vague queries, we integrate clarification and reasoning strategies into the process of skill construction. Specifically, if uncertain or ambiguous facets are detected in a conversation segment, DuIVA will take initiative and ask additional clarifying questions related to such facets. For example, as shown in Figure 2, given the query "change the destination to Baidu", the navigation skill identifies multiple candidates for "Baidu" and then will, in turn, ask the user for the desired one. In addition, we also integrate reasoning strategy into the skill to conduct entity coreference resolution. For example, given a query "navigate from the company to the car wash near home", the navigation skill is developed to infer the specific addresses of "the company" and "home" based on the attributes set by the user, as well as the specific address of "the car wash" w.r.t. the inferred home address.

Third, we build an effective mechanism to consistently uncover new user intents and develop new skills. Specifically, we accumulate the failed queries of DuIVA, and routinely cluster these queries into different intents based on the assumption that queries with the same intent tend to share more similar context features such as keywords. Then, we manually analyze the clustering results to discover new intents, develop new skills, and fine-tune different components of DuIVA to support the launch of new skills. As of December 31, 2021, over 860 skills are developed for DuIVA.

2.9 Services and Functionalities

DuIVA mainly provides two categories of fulfillment including: (1) services that help to fulfill the user needs like navigation and location search; and (2) functionalities that help to adjust app settings like volume adjustment and screen orientation toggle. During the dialogue, once a skill is activated and all slots are successfully filled, DuIVA will make an HTTP POST request that contains information

about the skill and the filled slots, to the webhook. After receiving the request, the webhook request is fulfilled through the deployed service or functionality of the Baidu Maps app. Then, a webhook response message is sent back to the DM of the activated skill.

2.10 Text To Speech

TTS aims at synthesizing natural-sounding speech from the generated responses of the skills, which is required to efficiently synthesize human-sounding speech with low latency. In practice, we adopt an in-house TTS system built at Baidu, which has achieved significant improvement in naturalness and latency.

3 CONTINUAL LEARNING WITH FEEDBACK FROM BOTH SUCCESSES AND FAILURES

Building an industrial IVA system that characterizes robust stability and industrial-grade reliability is not something that can be done overnight, it is the culmination of step-wise processes and a series of consecutive refinements. The initial deployment of DuIVA inevitably suffers from the new-system cold-start problems, such as the ability to generalize to unseen user intents, the ability to handle queries beyond the manually defined templates, and the ability to be more agile and responsive to newly emerging skills. To address these problems, after deployment, we constantly improve the overall performance of DuIVA with user feedback and continual learning, which includes self-training and active learning with user feedback from both successes and failures.

3.1 Self-training with Positive Feedback

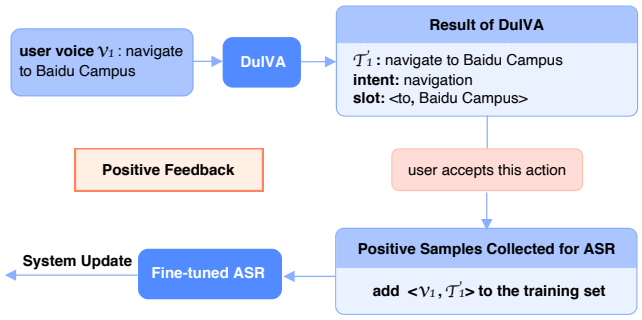
Traditional self-training method [20] typically selects the samples with the most confidence scores, which suffers from two main problems. (1) *Error propagation.* The selection criteria inevitably introduce plenty of noise. (2) *Selection bias.* The selection criteria tend to neglect the hard and potentially informative samples.

To alleviate these problems, we leverage positive feedback from users to help select samples, which can consistently provide reliable guidance during the selection process. Figure 4a shows the overall process of self-training with positive feedback on the ASR task. Given an input user voice V_1, DuIVA generates an action by conducting its components in sequence. If the user accepts the action proposed by DuIVA with successful action completion, we can anticipate that the ASR component has correctly transcribed the speech V_1 into the text T_1. Then, we add $<V_1, T_1>$ to the training set of ASR. By utilizing the positive user feedback in the self-training process, we can select high-quality real-world samples with less noise. The augmented training set will be used to fine-tune the ASR component. In practice, the strategy of self-training with positive feedback is also used to improve other components of DuIVA.

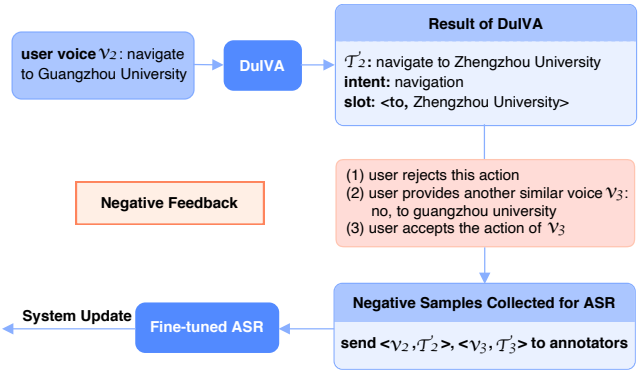
3.2 Active Learning with Negative Feedback

Traditional active learning method [24] typically selects the samples with the least confidence scores, which suffers from the problem of selection bias since a large amount of samples may be neglected.

To alleviate this problem, we leverage negative feedback from users to help select samples. Figure 4b shows the overall process of active learning with negative feedback on the ASR task. Given an input user voice V_2, DuIVA generates an action based on it. If the



(a) An example of positive feedback.



(b) An example of negative feedback.

Figure 4: Continual learning with user feedback.

user rejects this action, and meanwhile provides another similar voice V_3, DuIVA will process V_3 and provide a new action for it. If the user accepts the new action proposed by DuIVA with successful action completion, we can assume that the ASR component has incorrectly transcribed the speech V_2 into the text T_2. Then, we send both $<V_2, T_2>$ and $<V_3, T_3>$ to annotators to get the corrected result $<V_2, T_2^{'}>$, which is used to augment the training set of ASR. The augmented training set will be used to fine-tune the ASR component. With the help of negative feedback, we can effectively find the samples that DuIVA fails to handle, which results in less selection bias. In practice, the strategy of active learning with negative feedback is also used to improve other components of DuIVA.

4 RESULTS AND ANALYSIS

In this section, we present our main results and findings, including DuIVA performance, user growth, user preferences for different skills, and the comparison of interaction efficiency.

4.1 DuIVA Performance and User Growth

As detailed in Section 2, the DuIVA system is cascaded over multiple components to understand a user's query and map the query to a variety of different skills. As a consequence, the overall performance of DuIVA follows the behavior of Liebig's law[1] that it is limited by the component with the worst performance. To this end, we use an end-to-end system-level evaluation method to address the challenge in obtaining a list of locally optimal components rather than

[1]https://en.wikipedia.org/wiki/Liebig%27s_law_of_the_minimum

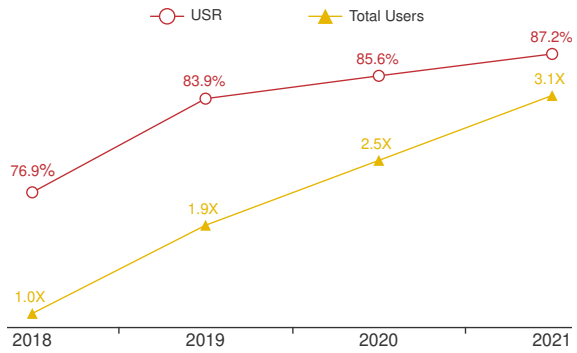


Figure 5: The year-over-year improvement of *USR* and users.

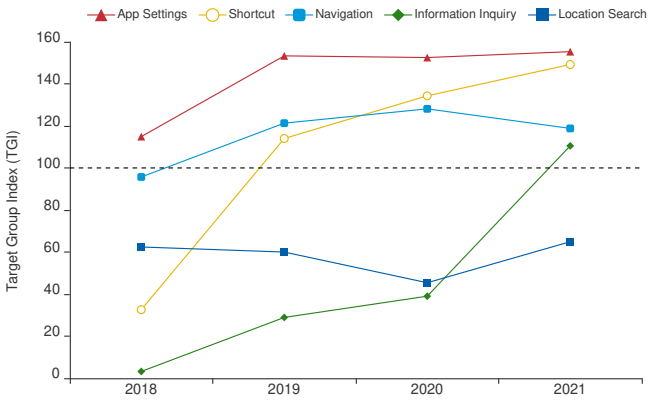


Figure 6: User preferences for different skills.

yielding a globally optimal system. Specifically, we employ user satisfaction rate (*USR*) to evaluate the performance of DuIVA, which is an evaluation metric that propagates the combination of multiple component-level optimizations to the system-level evaluation. *USR* is calculated by $USR = \frac{N_C}{N_T} \times 100\%$, where N_T is the total number of voice queries issued by users, and N_C is the number of suggested actions that are successfully accepted by users. *USR* is a utility that can not only measure user satisfaction of DuIVA, but also drive optimizations of different components of DuIVA. Large-scale online A/B testing is widely used to improve production applications at Baidu such as search [10–12, 14] and ETA [4, 5]. We also use online A/B testing on the live traffic of DuIVA to evaluate its performance.

DuIVA has already been embedded in the Baidu Maps app since November 2017, and we have improved its performance with hundreds of optimizations over the last four years. As shown in Figure 5, the *USR* of DuIVA has been continuously improved. Compared with the initial deployment, the recently deployed DuIVA has achieved 10.3% point absolute improvement, which demonstrates the effectiveness of the proposed continual learning method (see §3 for details). In addtion, the customer satisfaction score has remained an average of 91.9 between the years 2019 to 2021 based on the customer satisfaction surveys made by a third-party market research company. This confirms that the recent score of *USR* (87.2%) is highly correlated with the customer satisfaction score of DuIVA.

Figure 5 shows that the number of users has maintained sustained and rapid growth in the past four years, which indicates that DuIVA is more and more popular as a result of the hands-free, eyes-free user-to-app voice interaction modality, the significantly improved interaction efficiency (see §4.3 for details), and the continuously improved *USR*. By providing users with in-app voice activation, natural language queries, and multi-round dialogue, the accessibility and usability of the Baidu Maps app have been significantly improved. As of December 31, 2021, over 530 million users have used DuIVA, which demonstrates that DuIVA is an industrial-grade and popular in-app intelligent voice assistant.

4.2 User Preferences for Different Skills

To understand the popularity trends of DuIVA skills across the two interaction modalities (voice interaction and visual-manual interaction), we use target group index (*TGI*) to investigate user preferences for different skills. See Appendix A.2 for more details about this metric. *TGI* is a relatively unbiased indicator that reflects the preferences of users in terms of interaction modality when interacting with the app. In general, if the *TGI* score of a skill exceeds 100, it implies that most users prefer to use this skill via voice interaction. We calculate the *TGI* scores for the most representative skills that can be interacted with both modalities over the 4-year period beginning in 2018.

As shown in Figure 6, the *TGI* scores for four categories of skills exceed 100 in 2021, which indicates that users prefer to use these skills via DuIVA. Among which, the skills of app settings and shortcut have maintained much higher *TGI* scores, as such skills typically involve multiple steps and a series of visual-manual interactions with the app. Compared with the initial scores in 2018, the *TGI* scores of these skills have achieved sustained growth in the past four years. However, the *TGI* score of location search skills has remained an average of 58.2 between the years 2018 to 2021, which shows that the location search skills of DuIVA are the least popular ones. One of the main reasons for this plateau is that the location search based on visual-manual interaction has been fully optimized with much higher interaction efficiency, where users only need to type less than 2.62 keystrokes to obtain the desired POIs [3]. This is also evident from the comparison of skill completion time between DuIVA and visual-manual interaction (see §4.3 for details), where DuIVA achieves the minimal relative time saving for the location search skills compared with visual-manual interaction.

4.3 Voice Interaction vs. Visual-manual Interaction on Efficiency

To investigate whether DuIVA could improve the efficiency of user-to-app interaction, we calculate the skill completion time of interacting with the Baidu Maps app while driving and not driving via two interaction modalities (DuIVA and visual-manual interaction). The completion time of a skill is computed by the averaged time of successful skill completions for millions of queries using the anonymized query logs of Baidu Maps. Table 1 shows the results. From the results, we have the following observations.

First, compared with visual-manual interaction (VMI), DuIVA can greatly reduce the skill completion time in all scenarios. Among which, the average improvement of relative time saving (RTS) in driving scenario is much higher than that in non-driving scenario. This shows that DuIVA is able to significantly improve the efficiency of driver-to-app interaction, because it can minimize driver

Table 1: Skill completion time (in seconds) for the five categories of skills in different scenarios. VMI = visual-manual interaction. RTS = relative time saving, which is computed by RTS = (VMI − DuIVA)/VMI**.**

Skill	Use while not driving			Use while driving		
	VMI	DuIVA	RTS	VMI	DuIVA	RTS
Shortcut	9.82	5.86	40.3%	191.17	6.29	96.7%
Information inquiry	5.16	4.77	7.6%	8.97	4.79	46.6%
Navigation	8.02	5.02	37.4%	8.20	4.97	39.4%
App settings	6.16	5.52	10.4%	6.69	4.96	25.9%
Location search	5.21	4.93	5.4%	5.60	4.91	12.3%

distraction by freeing both eyes and hands from the necessity to perform tasks on smartphones during driving. Notably, the completion time of using shortcut skills while driving is reduced from 191.17 to 6.29 seconds (a RTS gain of 96.7% by a large margin). This demonstrates that voice interaction is a better interaction modality with the Baidu Maps app to accomplish a task that involves multiple steps and a series of visual-manual interactions, because it can significantly minimize driver distraction and promote safe driving.

Second, in terms of VMI, the completion time of all skills in driving scenario is much higher than that in non-driving scenario. The main reason is that interacting with the app while driving through VMI modality inevitably causes driver distraction, due to the highly conspicuous nature of the time-sharing, multi-tasking behavior of switching between driving and performing tasks on smartphones. This suggests that it is necessary to enable a voice interaction modality within map apps as a supplementary interface.

Third, in terms of DuIVA, there is little difference between the completion time of individual skills in both driving and non-driving scenarios. This demonstrates that voice interaction is a more efficient and stable interaction modality with the Baidu Maps app to accomplish different tasks. To have an intuitive understanding of the voice interaction, see Appendix A.3 for showcases on how DuIVA can improve the efficiency of user-to-app interaction.

5 CONCLUSIONS

In this paper, we suggest an industrial-grade and production-proven solution DuIVA for building an in-app intelligent voice assistant. DuIVA is designed to enable users to interact with map apps through voice interaction in a completely hands-free and eyes-free manner. Experiments and analysis on real-world datasets demonstrate that the amount of time and effort required to accomplish user-to-app interaction with DuIVA during driving is greatly reduced.

REFERENCES

[1] Eric S Atwell and Stephen Elliot. 1987. Dealing with ill-formed English text. *The computational analysis of English: a corpus-based approach* (1987), 120–138.

[2] Hongshen Chen, Xiaorui Liu, Dawei Yin, and Jiliang Tang. 2017. A Survey on Dialogue Systems: Recent Advances and New Frontiers. *ACM SIGKDD Explorations Newsletter* 19, 2 (2017), 25–35.

[3] Miao Fan, Yibo Sun, Jizhou Huang, Haifeng Wang, and Ying Li. 2021. Meta-Learned Spatial-Temporal POI Auto-Completion for the Search Engine at Baidu Maps. In *KDD*. 2822–2830.

[4] Xiaomin Fang, Jizhou Huang, Fan Wang, Lihang Liu, Yibo Sun, and Haifeng Wang. 2021. SSML: Self-Supervised Meta-Learner for En Route Travel Time Estimation at Baidu Maps. In *KDD*. 2840–2848.

[5] Xiaomin Fang, Jizhou Huang, Fan Wang, Lingke Zeng, Haijin Liang, and Haifeng Wang. 2020. ConSTGAT: Contextual Spatial-Temporal Graph Attention Network for Travel Time Estimation at Baidu Maps. In *KDD*. 2697–2705.

[6] Gregory M Fitch, Susan A Soccolich, Feng Guo, Julie McClafferty, et al. 2013. *The Impact of Hand-Held and Hands-Free Cell Phone Use on Driving Performance and Safety-Critical Event Risk*. Technical Report.

[7] David Goddeau, Helen Meng, Joseph Polifroni, Stephanie Seneff, and Senis Busayapongchai. 1996. A form-based dialogue manager for spoken language applications. In *ICSLP*. 701–704.

[8] Agustín Gravano and Julia Hirschberg. 2011. Turn-taking cues in task-oriented dialogue. *Computer Speech & Language* 25, 3 (2011), 601–634.

[9] A. Howard, Menglong Zhu, Bo Chen, D. Kalenichenko, Weijun Wang, T. Weyand, M. Andreetto, and H. Adam. 2017. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861* (2017).

[10] Jizhou Huang, Shiqiang Ding, Haifeng Wang, and Ting Liu. 2018. Learning to Recommend Related Entities With Serendipity for Web Search Users. *ACM Trans. Asian Low-Resour. Lang. Inf. Process.* 17, 3, Article 25 (April 2018), 22 pages.

[11] Jizhou Huang, Haifeng Wang, Miao Fan, An Zhuo, and Ying Li. 2020. Personalized Prefix Embedding for POI Auto-Completion in the Search Engine of Baidu Maps. In *KDD*. 2677–2685.

[12] Jizhou Huang, Haifeng Wang, Yibo Sun, Miao Fan, Zhengjie Huang, Chunyuan Yuan, and Yawen Li. 2021. HGAMN: Heterogeneous Graph Attention Matching Network for Multilingual POI Retrieval at Baidu Maps. In *KDD*. 3032–3040.

[13] Jizhou Huang, Haifeng Wang, Yibo Sun, Yunsheng Shi, Zhengjie Huang, An Zhuo, and Shikun Feng. 2022. ERNIE-GeoL: A Geography-and-Language Pre-trained Model and its Applications in Baidu Maps. In *KDD*.

[14] Jizhou Huang, Haifeng Wang, Wei Zhang, and Ting Liu. 2020. Multi-Task Learning for Entity Recommendation and Document Ranking in Web Search. *ACM Trans. Intell. Syst. Technol.* 11, 5, Article 54 (July 2020), 24 pages.

[15] Bret Kinsella and Ava Mutchler. 2019. In-Car Voice Assistant Consumer Adoption Report.

[16] David R Large, Gary Burnett, Ben Anyasodo, and Lee Skrypchuk. 2016. Assessing cognitive demand during natural language interactions with a digital driving assistant. In *AutomotiveUI*. 67–74.

[17] Nilli Lavie. 2010. Attention, distraction, and cognitive control under load. *Current directions in psychological science* 19, 3 (2010), 143–148.

[18] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. 2017. Focal loss for dense object detection. In *ICCV*. 2980–2988.

[19] Zhouhan Lin, Minwei Feng, C. N. Santos, Mo Yu, Bing Xiang, Bowen Zhou, and Y. Bengio. 2017. A structured self-attentive sentence embedding. In *ICLR*.

[20] David McClosky, Eugene Charniak, and Mark Johnson. 2006. Effective self-training for parsing. In *NAACL*. 152–159.

[21] Bastian Pfleging, Stefan Schneegass, and Albrecht Schmidt. 2012. Multimodal interaction in the car: combining speech and gestures on the steering wheel. In *AutomotiveUI*. 155–162.

[22] Andreas Riener, Myounghoon Jeon, Ignacio Alvarez, and Anna K Frison. 2017. Driver in the loop: Best practices in automotive sensing and feedback mechanisms. In *AutomotiveUI*. 295–323.

[23] Florian Roider, Sonja Rumelin, Bastian Pfleging, and Tom Gross. 2017. The effects of situational demands on gaze, speech and gesture input in the vehicle. In *AutomotiveUI*. 94–102.

[24] Burr Settles. 2009. *Active learning literature survey*. Technical Report. University of Wisconsin–Madison Department of Computer Sciences.

[25] Yibo Sun, Jizhou Huang, Chunyuan Yuan, Miao Fan, Haifeng Wang, Ming Liu, and Bing Qin. 2021. GEDIT: Geographic-Enhanced and Dependency-Guided Tagging for Joint POI and Accessibility Extraction at Baidu Maps. In *CIKM*. 4135–4144.

[26] Yu Sun, Shuohuan Wang, Yukun Li, Shikun Feng, Hao Tian, Hua Wu, and Haifeng Wang. 2020. Ernie 2.0: A continual pre-training framework for language understanding. In *AAAI*. 8968–8975.

[27] Amrita S Tulshan and Sudhir Namdeorao Dhage. 2018. Survey on virtual assistant: Google assistant, siri, cortana, alexa. In *SIRS*. 190–201.

[28] Gokhan Tur and Renato De Mori. 2011. *Spoken language understanding: Systems for extracting semantic information from speech*. John Wiley & Sons.

[29] Walter W Wierwille. 1993. Demands on driver resources associated with introducing advanced technology into the vehicle. *TR_C* 1, 2 (1993), 133–142.

[30] Jinhua Xiong, Qiao Zhang, Shuiyuan Zhang, et al. 2015. HANSpeller: a unified framework for Chinese spelling correction. In *IJCLCLP*. 1–22.

[31] Baoshi Yan, Fuliang Weng, Zhe Feng, Florin Ratiu, et al. 2007. A conversational in-car dialog system. In *NAACL*. 23–24.

[32] Zhao Yan, Nan Duan, Peng Chen, Ming Zhou, et al. 2017. Building Task-Oriented Dialogue Systems for Online Shopping. In *AAAI*. 4618–4625.

[33] Yi Yang and Arzoo Katiyar. 2020. Simple and Effective Few-Shot Named Entity Recognition with Structured Nearest Neighbor Learning. In *EMNLP*. 6365–6375.

[34] Ruiqing Zhang, Chao Pang, Chuanqiang Zhang, Shuohuan Wang, Zhongjun He, Yu Sun, Hua Wu, and Haifeng Wang. 2021. Correcting Chinese spelling errors with phonetic pre-training. In *Findings of ACL*. 2250–2261.

A APPENDIX

In this section, we detail the template-based NLU method and the evaluation metric of target group index. We also showcase how DuIVA can improve the efficiency of user-to-app interaction.

A.1 Template-based NLU

A template is a sequence of words and slots, where each slot can be a word or a fragment matched by a dictionary or a function. A template can be used to jointly detect user intent (e.g., navigation) and extract semantic slots (e.g., the destination and options for routing) from a query (e.g., "navigate to the nearest gas station"). For example, we can obtain the NLU results—intent: navigation, slot: <to, Baidu Campus>, from the query "change destination to Baidu Campus" by using the template "change destination to [D: POI]". In practice, we have built more than 35,000 templates for the 860 skills of DuIVA to handle as many diverse queries as possible, which poses a great challenge to the decoding speed of template matching. To address this problem, we adopt the Trie tree structure to accelerate the template matching process. Specifically, we construct two kinds of Trie trees, including a template tree and a dictionary tree.

Figure 7 shows a schematic diagram of the template tree structure. For the template tree, each node in the tree represents a transition state. The leaf node is the final state, which represents a successful match between a query and a template. Each directed edge shows the state transition when matching the special state transition condition of a word (e.g., "destination" and "theme"), a dictionary (e.g., "[D: POI]" and "[D: mode]"), or a function ("[F: functionality]"). The dictionary tree shares a similar structure with that of the template tree, which is used to extract and determine the type of the transition condition of the current query. For the dictionary tree, each node is a transition state, and each directed edge shows the state transition when matching the special state transition condition of a fixed word. During the template matching, the dictionary tree scans words in the query one by one, and sends the successfully matched item to the template tree once a state transition condition of a word, a function, or an ignored word is fulfilled. In practice, to build the Trie tree and perform the template matching, we use the in-house lexparser framework that has been developed and maintained for more than 11 years, which offers the advantages of low memory consumption, fast matching speed, and high flexibility.

A.2 Target Group Index

Target group index (TGI) is calculated by $TGI = (S_v/S_t)/(N_v/N_t) \times 100$, where S_v is the traffic (the number of times a skill is invoked) of a specific skill through DuIVA, S_t is the overall traffic of this skill through both DuIVA and visual-manual interaction (VMI), N_v is the overall traffic of DuIVA, and N_t is overall traffic of all skills through both DuIVA and VMI. Here we present a toy example to explain the calculation of TGI. Suppose that a total of 1,000 queries are issued by users within a certain period of time, of which 300 are interacted with DuIVA and 700 are interacted with VMI. Hence, the traffic proportion of DuIVA is 30%. Further, among the 1,000 queries, the navigation skills are totally invoked 100 times, of which 40 are interacted with DuIVA and 60 are interacted with VMI. Hence, the traffic proportion of navigation through DuIVA is 40%. In this case, the traffic proportion of navigation through DuIVA (40%) is greater than the overall traffic proportion of DuIVA (30%), which results in a TGI score of 133.3. This implies that most users prefer to use the navigation skills via voice interaction. It shows that TGI can be used to quantify user preferences for DuIVA skills, since it measures the popularity of a category of DuIVA skills by normalizing by their overall traffic via both interaction modalities.

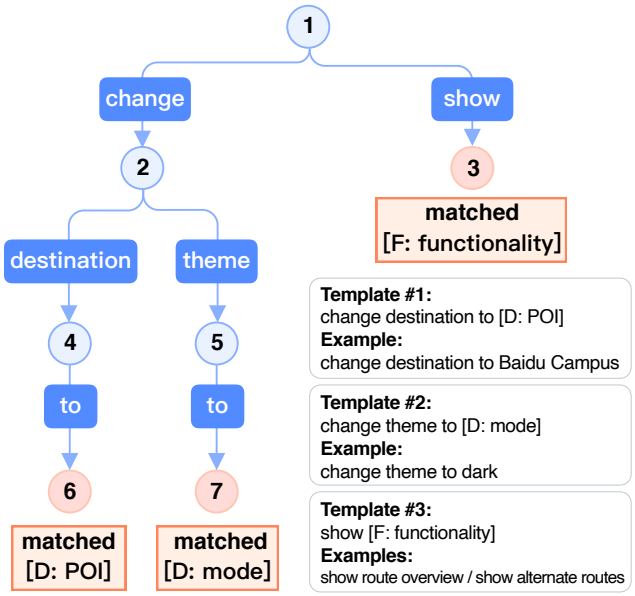


Figure 7: The template tree.

A.3 User-to-app Voice Interaction Examples

As shown in Figure 8-12, we showcase four DuIVA examples for individual categories of skills. We omit the wake-up round of each dialogue session due to space constraints.

Figure 8 shows four examples of app settings including "brightness control", "volume control", "enable dark mode", and "save where you parked". These examples show that DuIVA is able to significantly improve the efficiency of changing app settings with a simple voice command, and to perform query intent clarification.

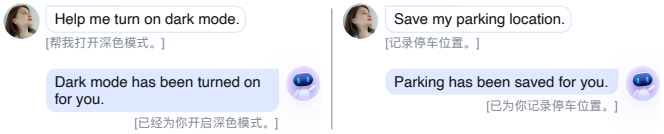
Figure 9 shows four examples of information inquiry including "find driving restriction policy", "view live traffic conditions", "find business hours", and "find the current location". These skills enable users to get travel-related information in a hands-free and eyes-free manner. They are particularly useful when driving.

Figure 10 shows four examples of location search including "find the nearest gas station", "find pet-friendly restaurants", "find top rated hot pot", and "find POIs for vague queries". These examples show that DuIVA offers an advanced location search feature.

Figure 11 shows four examples of shortcut including "set multiple stops", "change the destination during driving", "add a waypoint", and "plan a route with options". These skills can significantly promote safe driving, because they can accomplish a task that involves a series of visual-manual interactions with just a single voice query.

Figure 12 shows four examples of navigation including "faster route suggestion", "fast navigation", "the nearest navigation point", and "check traffic conditions". These skills can significantly reduce a driver's attention allocation towards finishing a secondary task (e.g., check traffic information) while driving (the primary task).

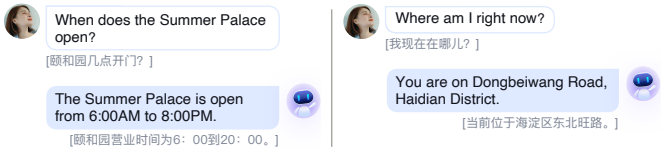
(a) Brightness control. (b) Volume control.



(c) Enable dark mode. (d) Save the parking location.

Figure 8: DuIVA examples of app settings.

(a) Find driving restriction policy. (b) View live traffic conditions.



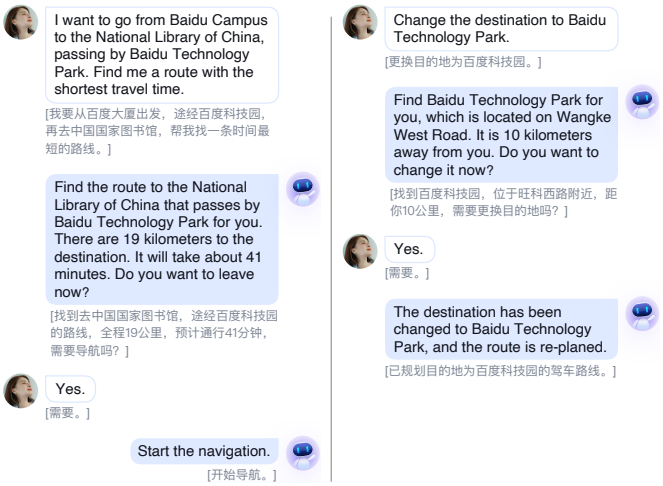
(c) Find business hours. (d) Find the current location.

Figure 9: DuIVA examples of information inquiry.

(a) Find the nearest gas station. (b) Find pet-friendly restaurants.

(c) Find top rated hot pot. (d) Find POIs for vague queries.

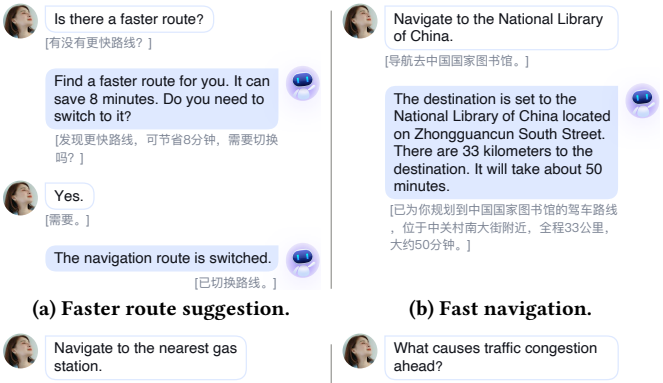
Figure 10: DuIVA examples of location search.



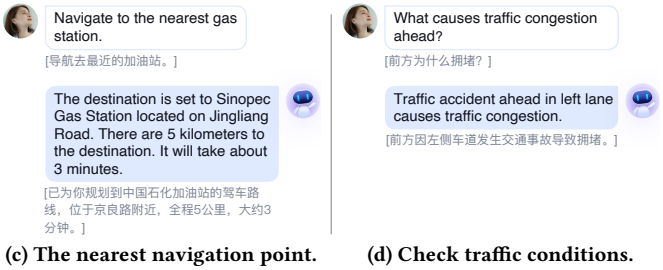
(a) Set multiple stops. (b) Change the destination.

(c) Add a waypoint. (d) Plan a route with options.

Figure 11: DuIVA examples of shortcut.



(a) Faster route suggestion. (b) Fast navigation.



(c) The nearest navigation point. (d) Check traffic conditions.

Figure 12: DuIVA examples of navigation.