A Speech-enabled Virtual Assistant for Efficient Human-Robot Interaction in Industrial Environments

LI, Chen; Chrysostomou, Dimitrios; Yang, Hongji

Published in:
Journal of Systems and Software

DOI (link to publication from Publisher):
10.1016/j.jss.2023.111818

Publication date:
2023

Document Version
Publisher's PDF, also known as Version of record

The Journal of Systems & Software 205 (2023) 111818

Contents lists available at ScienceDirect

The Journal of Systems & Software

journal homepage: www.elsevier.com/locate/jss

A speech-enabled virtual assistant for efficient human–robot interaction in industrial environments☆

Chen Li [a,*], Dimitris Chrysostomou [a], Hongji Yang [b]

[a] Department of Materials and Production, Aalborg University, Aalborg East, Aalborg, DK-9220, Denmark
[b] School of Computing and Mathematical Sciences, University of Leicester, Leicester, LE1 7RH, UK

ARTICLE INFO

Article history:
Received 9 February 2023
Received in revised form 1 July 2023
Accepted 28 July 2023
Available online 3 August 2023

Dataset link: https://github.com/lcroy/Virtual-Assistant-Max

Keywords:
Human–robot interaction
Natural language processing
Interactive systems
Client–server systems

ABSTRACT

This paper presents a natural language-enabled virtual assistant (VA), named Max, developed to support flexible and scalable human–robot interactions (HRI) with industrial robots. Regardless of the numerous natural language interfaces already proposed for intuitive HRI on the industrial shop floor, most of those interfaces remain tightly bound with a specific robotic system. Besides, the lack of a natural and efficient human–robot communication protocol hinders the user experience. Therefore three key elements characterize the proposed framework. First, a Client–Server style architecture is introduced so Max can provide a centralized solution for managing and controlling various types of robots deployed on the shop floor. Second, inspired by human–human communication, two conversation strategies, lexical-semantic and general diversion strategies, are used to guide Max's response generation. These conversation strategies were embedded to improve the operator's engagement with the manufacturing tasks. Third, we fine-tuned the state-of-the-art (SOTA) pre-trained model, Bidirectional Encoder Representations from Transformers (BERT), to support a highly accurate prediction of requested intents from the operator and robot services. Multiple experiments were conducted using the latest iteration of our autonomous industrial mobile manipulator, "Little Helper (LH)", to validate Max's performance in a real manufacturing environment.

1. Introduction

Human–robot interaction (HRI) has been the focus of groundbreaking research for decades (Jost et al., 2020; Raj et al., 2022). Coupled with the rapid development of Artificial Intelligence (AI), various advanced technologies such as real-time object detection (Buhl et al., 2019; Cheng et al., 2022), deep reinforcement learning (Arana-Arexolaleiba et al., 2019), and natural language processing (NLP), are introduced to enhance HRI for industrial robots (Lithoxoidou et al., 2020). However, as one of the latest articles of WEIRD magazine mentioned, "As Robots Fill the Workplace, They Must Learn to Get Along" (Knight, 2021). The presence of advanced technologies alone, do not suffice for natural communication and interaction with multiple types of robots that are designed for different purposes and used by various operators (Villani et al., 2018). For example, the Mobile Industrial Robot (MiR) focuses on internal logistics technologies such as navigation and obstacle avoidance. However, it only communicates with the user based on light signals. At the same time, many industrial robot manipulators are branded as collaborative, often due to marketing reasons and primarily based on their control and safety strategies. However, while they revolutionized the industry enabling human–robot collaboration (HRC) outside of safety fences, they barely incorporate ways to engage users into a natural language dialogue to enhance the communication and interaction with them (Hjorth and Chrysostomou, 2022). Therefore, it is necessary to introduce a flexible and scalable NLP-enabled interface for HRI and HRC, which works with various industrial robots on the shop floor based on a well-designed architecture that provides a centralized way for robot management and maintenance.

There are many mature language-enabled VAs, e.g., Alexa,[1] Google Assistant[2] and Siri[3] available commercially and used by millions of users worldwide. Their key feature is their impressive capacity for handling robust NLP and continuous natural dialogues. Nevertheless, most of those products are designed in the context of entertainment and remain unsuitable for direct use in robotics, especially in an industrial context.

☆ Editor: W. Eric Wong.

* Corresponding author.
E-mail addresses: cl@mp.aau.dk (C. Li), dimi@mp.aau.dk (D. Chrysostomou), hongji.yang@leicester.ac.uk (H. Yang).

1 https://www.amazon.com/b?ie=UTF8&node=17934671011
2 https://developers.google.com/assistant
3 https://developer.apple.com/siri/

https://doi.org/10.1016/j.jss.2023.111818

On the other hand, many research efforts utilize verbal cues to enhance HRI in industrial environments (Mavridis, 2015). Specifically, Li et al. developed a language-enabled virtual assistant, Bot-X, to control a production line composed of eight Festo CP modules and a KUKA robot for product assembly task (Li and Yang, 2021). Maksymova et al. proved that numerous models could be used for voice control of an industrial robot such as logical, semantic networks, and Petri Nets in the context of collaborative assembly (Maksymova et al., 2017). Additionally, Bingol and Aydogmus evaluated the capabilities of deep neural networks for the classification of a set of commands in a natural speech recognition system for the interactive control of an industrial manipulator in various industrial tasks (Bingol and Aydogmus, 2020). González-Docasal et al. progressed even further and integrated a semantic interpreter able to extract semantic information from transcribed spoken content to enable an industrial robot to understand the intention of the operator and execute a collaborative task (González-Docasal et al., 2020).

Naturally, since industrial robots mainly assist the users with manufacturing tasks (Kumar et al., 2021), task completion experience is usually set as the primary evaluation goal of robots' performance in most of the aforementioned scenarios (González-Docasal et al., 2020). However, an operator remains the most flexible entity on the shop floor which needs to handle a versatile range of tasks and tools with robots' cooperation where task-completion is not the only requirement, e.g., collaborative products assembly, and resource management and logistics assisted by mobile robots (Aceto et al., 2019).

This work is motivated by multiple studies from social, service, and lately, industrial robotics, which have proven that creating a pleasant and symbiotic human–robot collaboration often improves the user's engagement and leads to increased productivity (Pérez et al., 2020). Key elements of such successful HRI usually are the strong sense of commitment from the operator (Székely et al., 2019) and the enhanced user experience (Prati et al., 2021).

Based on our previous prototype (Li et al., 2021), we developed an intelligent VA to enable these elements in industrial use cases. We call it Max, and it utilizes a Client–Server (CS) style architecture and RESTful APIs to provide a scalable and flexible NLP solution for intuitive HRI. It supports various industrial robots on the shop floor by maintaining industrial robot services on the server-side alone while the robot control agent lies on the Max client. Furthermore, powered by the state-of-the-art (SOTA) model and inspired by human–human communication, Max can understand the operator's intent, track the dialogue history and enhance the user experience by generating humanized responses.

We summarize our contributions as follows:

- **From architecture perspective**, we propose the design of a natural language-enabled VA, fine-tuned for the needs of industrial tasks. It is equipped with a scalable and flexible Client–Server architecture with RESTful APIs enabling a modular, plug-and-play ability for various robot services. Therefore, the VA is not tightly bound with a specific robotic system when integrating with various shop floor industrial robots.
- **From the interaction perspective**, this is the first study that integrates the human-to-human conversation strategy into HRI for industrial robots to bootstrap the shop floor workers' engagement. Furthermore, we fine-tuned, pre-trained Bidirectional Encoder Representations from Transformers (BERT) (Devlin et al., 2018) model on our dialogue dataset in the industrial robots domain with a high inference accuracy.
- **From the performance perspective,** six performance metrics, with the target of measuring the accuracy of language understanding, the success rate of task completion, and parallel request handling ability, are used to evaluate the proposed VA. The result shows its robustness in three diverse industrial scenarios with realistic noise levels, usually present in a factory.

In Section 2 of this paper, we describe the proposed intelligent VA presenting its design specifications, system architecture, and core components. We present the experiments and evaluate Max's performance in Section 3, and we finalize the paper with reflections and concluding remarks in Section 4.

2. System description

The main objective of Max is to enable industrial users to easily maintain and control their industrial robots and enhance the user experience in a natural and user-friendly way. However, such flexibility introduces certain challenges:

- Driven by the fast-changing market demands, manufacturers need to adjust the shop floor quickly, e.g., relocate robots to another production line, install a new robot. Therefore, the design of a VA should be able to adapt to the dynamically changing working environment and support a rapid expansion of the shop floor.
- To optimize HRI, it is essential to add specific features, e.g., natural responses to dialogue and intent prediction, that will add extra value to the VA and enhance the overall performance. These enhancements of user experience are currently lacking in HRI with industrial robots and are introduced with the proposed framework.

2.1. Design principles

To address the challenges mentioned above, we built Max based on two design principles.

- Achieve a flexible and scalable HRI. We wanted Max's architecture to be simple and easily maintainable. To achieve flexible and scalable HRI, Max should enable a scalable shop floor environment and well-defined interfaces to support the communication for Human-to-Max and Max-to-robots.
- Support natural and humanized communication. We wanted Max to support natural and humanized communication to enable natural communication and allow novice operators to interact with it easily. Max should be able to improve user engagement through a natural dialogue environment. Besides, it should invite the operator to a manufacturing task and initiate an activity related to the context of the conversation. Moreover, Max needs to be situation-aware and switch or end the conversation topic if the current task cannot be executed.

2.2. Architecture overview

Following the CS style architecture, two services, language service and an industrial robot service, are hosted on the Max server-side (see Fig. 1). Max client is mainly composed of a voice interface and robot control agent. RESTful HTTP requests support the communication between the Max server and client. Max client provides a voice interface (i.e., microphone and speaker) to interact with the human operator and leverages the different protocols (e.g., OPC UA, TCP/IP) to communicate with the shop floor robots.

Max is designed and developed as a CS-based web application using Python Flask Framework (Grinberg, 2018). NGINX (Reese, 2008) is serving as a web server due to its high performance,

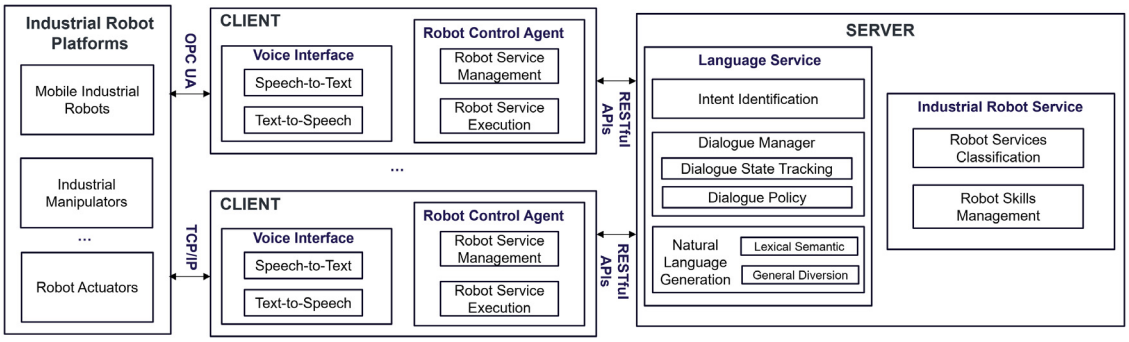

Fig. 1. The proposed architecture of the natural language-enabled VA.

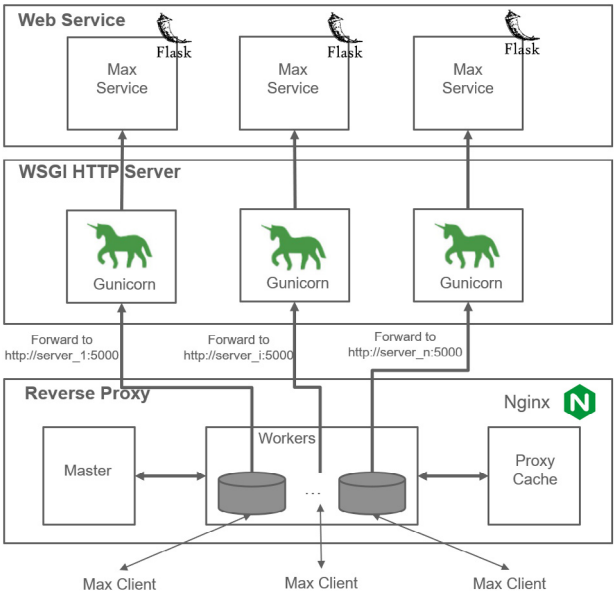

Fig. 2. Deployment of Max server on AAU Cloud.

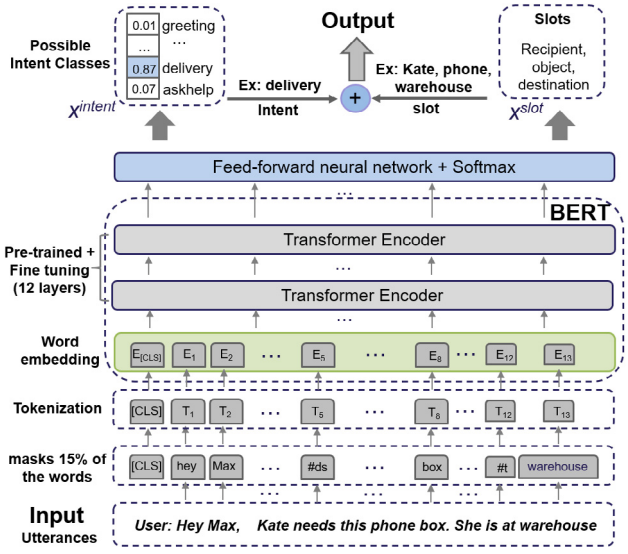

Fig. 3. An overview of pre-trained and fine-tuned BERT model. User utterance is split into multiple tokens and masked (15% of words) and fed to the BERT model. The outputs are the predicted user's intent with slots.

stability, and low resource consumption. To handle concurrent HTTPS connections and recover from crashes, Gunicorn (Chesneau et al., 2020) is chosen as a web server gateway interface (WSGI) application server (see Fig. 2). Max server is deployed both on Aalborg University (AAU) Cloud, a cluster of 32 servers, and a local Ubuntu 18.04 webserver in AAU learning factory (Nardello et al., 2017). All servers have the same configuration with Intel(R) Xeon(R) Silver 4110 CPU @ 2.10 GHz (32 cores) and 128 GB of memory. In general, Max clients send requests to the local web server directly. If the number of requests reaches the upper limit, the server will forward the further requests to the AAU Cloud to balance the workload and avoid the traffic congestion problem. The Max client is deployed on a Raspberry Pi 4 since the client's main responsibility is to provide voice support and invoke the robot control scripts.

2.3. Language service

Designing a VA tailored to industrial requirements is a challenging task. The VA should comprehend natural language conversations, perceive the working environment and task-related context, and actively interact with operators to provide advice and suggestions to assist the manufacturing task.

2.3.1. Human intent recognition

The input of the language service is a transcript transformed from operators' utterances sent by the Max client. In general, to understand the operator's intent (e.g., asking a mobile robot to perform a transportation task), the keywords extraction approach is widely used for mapping operator's utterance to ontology action instances (Mavridis and Roy, 2006). However, such an approach requires a predefined list of keywords for every robot action. Moreover, operators need to remember all keywords and say the right one to trigger an action. While such an approach is simple, it is also less flexible, especially for the workers with no or limited knowledge of the available robot operation skills.

In our work, we built an intent identifier by fine-tuning the base BERT (Devlin et al., 2018) model (a neural network architecture with 12 layers, 768 hidden units, 12 attention heads, and 110M parameters.) with one additional feed-forward neural network (FFNN) layer and Softmax as the activation function to normalize the output of the model (see Fig. 3). A series of sequential operations, Masking (15% of the words), Tokenization, and Embeddings are applied to the user's utterance to generate the input representation. A special symbol, [CLS], is added as the first token to indicate the beginning of the user's utterance. The embedded utterances are passed through 12 Transformer encoders. The last FFNN receives the results of the BERT model as input and outputs a possible user's intent and slots. The first hidden state of [CLS], h^0, and the rest of the hidden states, (h^1,

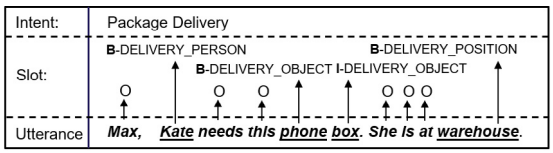

Fig. 4. An example utterance with annotations of slots in IOB tags, and the intent is Package Delivery. DELIVERY_PERSON denotes the recipient, DELIVERY_OBJECT stands for an object that needs to be delivered, DELIVERY_POSITION means destination of the delivery task.

```
{
   "Intent": "create_map",
   "slot": { "location":"yes", "battery":"optional", "schedule":"yes"}
},
{
   "Intent": "deliver_package",
   "slot": {...}
}
```

Fig. 5. A JSON file is defined to specify the required slots for each intent. The value of slot can be 'Yes' (means the slot value cannot be empty) or 'Optional' (means this slot can be an option but not necessary).

$h^2, \ldots, h^n$), of other tokens are feed into softmax layer to predict the intent, X^{intent}, (see (1)) and classify over the slot filling labels, X^{slot}, (see (2)).

$$X^{intent} = Softmax(W^{intent} h_0 + b^{inetnt}) \quad (1)$$

$$X^{slot} = Softmax(W^{slot} h_i + b^{slot}), 1 \leq i \leq N \quad (2)$$

where N represents the total tokens of the user utterance and h^i stands for the hidden state of the word X^i. We trained the fine-tuned BERT model with 0.977 and 0.968 on intent accuracy and slot F1 score, respectively.

The pre-trained BERT model provides contextualized sentence representation and is able to learn the meaning of the words in the given context. Therefore, operators may send the same command to a robot in various ways. For example, to mark a position on a digital map, an operator may say: "*please mark this position on the map*" or "*I need you to remember this location, Max.*". Therefore, our model encodes the operator's utterance (i.e., intents, slots annotated with Inside–outside–beginning (IOB) tags and slots values), and predicts the requested intent (i.e., intent requested by the operator for a task) and requested slots (i.e., parameters requested by the operator in the utterance), as shown in Fig. 4.

2.3.2. Dialogue manager

The predicted human intent and slot values are sent as input to the dialogue manager. It helps to maintain the dialogue history through the dialogue state tracking component and choose the corresponding actions or responses by the dialogue policy component.

Dialogue state tracking In general, a manufacturing task can take several steps to complete. Therefore, a VA should be able to maintain entire states of the task-related dialogue history. For example, requesting a mobile robot to create a 2D digital map of the shop floor may need two steps, i.e., confirming the location and scheduling the task. Such request, *(location=warehouse, schedule=(2 pm, 28 Jan 2021))* may take a 2-turn dialogue with the operator and interact with a database to obtain the required information to fulfil the tasks. Due to the major manufacturing task is composed of straightforward and atomic manipulations, our dialogue state tracking component sequentially takes the output (i.e., intent and slot values) of the BERT model after each sentence. The states will be updated based on the current user's utterance if the slot values are changed.

Dialogue policy It takes the all user's utterance, $User_0$, …, $User_i$, including intent ($intent_j$) and the slots ($slots_j$) of dialogue state tracking results, and the previous system actions ($SysAct_0$, …, $SysAct_{i-1}$) as input to compute the next corresponding system action ($SysAct_i$).

As already mentioned, a manufacturing task can only be performed when a robot obtains all the necessary information. In this work, a pre-defined dialogue policy file (DPF) in JSON format (see Fig. 5) is hosted on the Max server to specify all the required slots of each intent (i.e., task). Dialogue policy needs to verify the dialogue states according to the requested slots in DPF. The dialogue will continue until Max receives all the required slots. The algorithm 1 shows the system action computation process of dialogue policy. The output of the dialogue policy are computed *SysAct*, i.e., *requested_slots*, and response references, $res\_reference_i$, associated with the corresponding response template (see the following section.).

Algorithm 1 Dialogue Policy for system action computation

```
Input: the dialogue history, User_0, SysAct_0,...,User_i, and DPF
Output: the SysAct_i and res_ponse_i
extract the current user intent, intent_i, and previous intent, intent_{i-1}, from User_i and User_{i-1} respectively
if intent_i ≠ intent_{i-1} then
    Assign null to the SysAct_i and inconsistent_intent to res_reference_i
    return
else
    for Every dialogue D ∈ (User_0, ..., User_{i-1}) do
        save the obtained slots of each turn to obt_slots
    end for
end if
extract the slots, slots_i, from current user utterance, User_i
for Every slot ∈ slots_i do
    if slot ∈ obt_slots then
        update the obt_slots with the new value of the slot
    else
        save the slot to obt_slots
    end if
end for
extract all the required slots (req_slots) from DPF regarding the requested intent_i from User_i
for Every slot ∈ req_slots do
    if slot ∉ obt_slots then
        Assign requested_slot to the SysAct_i and res_reference_i
        return
    end if
end for
```

2.3.3. Natural language generation

The natural language generation component of Max helps to produce the system's utterance to the user. Max's response generation follows the frame-based dialogue architecture (Bobrow et al., 1977); that is, Max uses the pre-defined questions/answer templates associated with the slot of each frame as a response. The slots retrieved from the user's utterance will be filled into the templates (van Deemter et al., 2005). Fig. 6 shows one of the system response templates regarding the user's request of Fig. 4. The placeholders (e.g., [DELIVERY_OBJECT]) of the response template are replaced by the predicted slots (e.g., phone box) when it generates the final response.

One way to enhance the user experience is to improve user engagement by including features that the operators need from

Intent:	Package Delivery
Slot:	**B**-DELIVERY_PERSON: **Kate** **B**-DELIVERY_OBJECT: **phone** **I**-DELIVERY_OBJECT: **box** **B**-DELIVERY_POSITION: **warehouse**
System	Sure, I will delivery [**DELIVERY_OBJECT**] to [**DELIVERY_POSITION**] and hand it over to [**DELIVERY_PERSON**]

Fig. 6. An example of a system response template for Fig. 4 with annotations of slots in IOB tags.

```
{"services":[
        {
          "service name":"MiR",
          "skills":{ "delivery", "mark_position", "change_state", "report_location",
                    "report_battery", "report_mission"}
        },
        {
          "service name":"Franka",
          "skills": {…}
        }, …
      ]
}
```

Fig. 7. Exemplary schema of industrial robot services.

a VA integrated with an industrial robot (Lindblom et al., 2020). In our case, we focus on the communication between the operator and Max. Inspired by other work in human-to-human communication (Warren, 2006; Littlejohn and Foss, 2010) and dialogue systems (Banchs and Li, 2012; Yu et al., 2016) two dialogue strategies, namely lexical-semantic strategy, and general diversion strategy, are introduced so Max can improve the user engagement by providing a dynamic and humanized conversation environment while maintaining a high task-completion rate.

Lexical semantic strategy. Research in human–human conversation experience describes that people rarely repeat the same response even when asked the same question. For example, people may respond: "I am good, thanks". or "Fine, thank you". when someone greets them. This is defined as *Do not repeat* in the lexical-semantic strategy (Yu et al., 2016). We applied this strategy to Max to provide diverse responses for creating a humanized dialogue environment and increase the user favorability.

General diversion strategy. High engagement is also observed from active participation in human–human collaboration, e.g., questing each other, taking the initiative in a task, giving each other suggestions. To enable such abilities in Max, two general diversion strategies, namely *initial activities* and *switch a topic*, are introduced (Yu et al., 2016). *Initial activities* strategy enables Max to initiate a request to the operator to start working with manufacturing tasks at the appropriate time. For example, Max may say: "Three delivery tasks are scheduled today. Would you like me to do them now?". Max can also provide suggestions to the operators if the requested task cannot continue, based on the *switch a topic* strategy. For example, Max may say: "Sorry, I cannot identify this location on the map. Do you want to mark it in the system?".

Therefore, Max performs tasks as required by operators and is also able to support active dialogues for an improved user experience.

2.4. Industrial robot service

To verify whether Max supports the requested robot and skill (extracted from the dialogue state tracking results), an industrial robot service is provided by Max server. The CS-style architecture benefits from centralized management, which helps create a more flexible and scalable robot working environment. Two main functions are defined in industrial robot service, robot services classification and robot skills management. Max's server maintains a separate category for robot services on the server-side to interact with various industrial robots on the shop floor using the robot services classification function. It specifies the type of robots supported by Max.

In general, production line reconfiguration or the introduction of new production processes may often happen in industrial environments. Such changes may require the integration of new robots or the update of the existing robotic systems. Max's server CS style architecture provides robot skills management functionality to maintain the robot control scripts in a centralized way. Therefore, a developer needs to maintain those scripts (e.g., creating a new control script for a new robot) only on the server-side. A JSON style schema is leveraged to maintain the industrial robot services, as depicted in Fig. 7.

2.5. Voice interface

A significant difference between text-based and voice-based VA is the operators' medium to transfer their commands. A keyboard for the former case and verbal commands for the latter. However, an average professional typist usually types at speeds of 50 to 80 words per minute, while speech can reach 150–160 per minute (Williams, 1998; Ayres, 2005). Furthermore, it is impractical for shop floor workers to use the keyboard and monitor to interact with a VA during HRI, especially if the task requires a bimanual operation (Sheikholeslami et al., 2017).

Therefore, a voice-enabled interface for Max's client is designed to enable more efficient communication with low hardware requirements. Software-wise, many mature Speech-to-Text, and Text-to-Speech services are available on the market, e.g., Google Speech-to-Text API,[4] IBM Watson API,[5] and Amazon Polly Text-to-Speech.[6] They can recognize the human voice, translate it into text and synthesize natural sounding human speech while providing light APIs and libraries to invoke services. Hardware-wise, Max's client voice interface requires a pair of headphones with a built-in microphone and Bluetooth wireless connection. Such voice interface frees the hands of the operator while provides a natural communication for HRI in the shop floor.

In our work, we chose the Google Speech-to-Text and Amazon Polly Text-to-Speech services to enable Max's client voice interface. The operator's utterance transcripts are organized into a RESTful style HTTP request and sent to Max's server. The response message is then extracted from the server's reply (a JSON string), and a natural-sounding voice is generated.

2.6. Robot control agent

A vital aspect of the application is to use Max's client to control the shop floor's robots. A robot control agent is set to manipulate the robots according to the operator's verbal commands. Two sub-components, a robot service execution (RSE) component, and a robot service management (RSM) component, are implemented to achieve that goal.

According to the design principle, Max is designed to be robot-agnostic and aims to support various kinds of industrial robots such as mobile robots and industrial manipulators. RSE scripts specify robot control functionalities (e.g., mark position) and communication protocols (e.g., TCP/IP, OPC-UA) for each robot type. As Fig. 8 shows, RSM will invoke the corresponding RSE to control a robot after receiving the response. Furthermore, RSM maintains a local registry of all the local RSEs. It will require a new or updated RSE script if the local RSEs are not compatible with the target robot.

4 GoogleSpeech-to-Text.

5 IBMWatson.

6 AmazonPollyText-to-Speech.

Table 1
The measurement result of ambient noise in AAU learning factory.

Peaks: MAX	Peaks: MIN	Average Level (LAeq)
80.5 dB(A)	44.6 dB(A)	69.0 dB(A)

Request from Max Client:

```
requests.get(host + 'get_task/', params=
{'message':'Max, can you mark this
location?'}, headers= headers)
```

Response from Max Server:

```
{
   'service_name': 'MiR',
   'intent': 'POSITIONUPDATE',
   'slot': {
      'locat_name': 'warehouse'
    }
   'parameters': {
       'message': 'Sure, I will do that. ',
       'dialogue_state': 'Done',
       'result': 'good_request'
    }
}
```

Fig. 8. An example of a HTTP GET request from Max's client (asking LH to mark a position on the digital map), and a JSON reply from Max's server (including identified intent, slot, message, dialogue state).

3. Pilot study

In order to explore the capabilities of our proposed language-enabled VA, we design three scenarios. Here our focus is to evaluate the performance of the VA when it comes to assisting the operators' daily work, specifically internal logistics in our study.

In general, internal logistics focus on internal supply processes, transportation of materials and tools, and cargo distribution. Such tasks may require a mobile robot to deliver the goods to a location within the organization. A fleet manager or web application for robot control typically comes as a package with the chosen mobile robot to assist in scheduling and tracking the tasks. However, it could have a steep learning curve for inexperienced or new operators.

To this end, we tested the VA on two practical industrial cases, including *environment exploration* and *package delivery*. Furthermore, another three cases are tested with the focus on humanized response by applying *embedded conversation strategy*. These scenarios explore and test Max's capabilities to handle typical, everyday tasks from an industrial shop floor while satisfying the design considerations state in Section 2.1.

Based on the needs of the manufacturing environment, real-time assistance of manufacturing tasks requires that the VA can understand the operator's utterance with high accuracy. Therefore, the VA needs to consider the context of the dialogue and pay attention to the meaning of the words in the given context. Consequently, we fine-tuned the SOTA pre-trained BERT model and trained it on our dialogue dataset to predict the operator's intent and slots. Though the model theoretically achieves high performance on prediction on a pure text-based dataset, the prediction results based on the voice command in the real manufacturing environment may be lower due to the ambient noise (e.g., machine operating sounds, ventilation noises, human chatter). We tested Max in the aforementioned scenarios at our learning factory (Nardello et al., 2017), a smart production facility equipped with cyber–physical modules and autonomous robots, where the expected production-related activities happen daily. The ambient noise levels are measured by Smarter Noise[7] when experiments were conducted. The minimum, maximum and average ambient noise levels are shown in Table 1.

[7] http://smarternoise.com/en

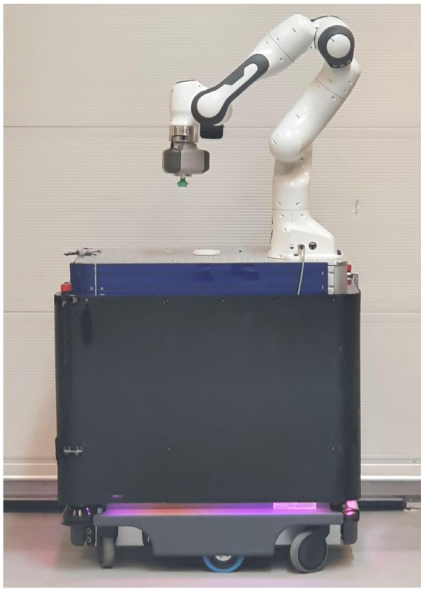

Fig. 9. The autonomous industrial mobile manipulator, Little Helper (LH), where our proposed virtual assistant is integrated.

3.1. Experimental setup

To demonstrate the effectiveness of our approach, we chose one of our autonomous industrial mobile manipulators, namely LH8 (Schou et al., 2018). In this iteration, LH combines a MiR 200 (on the bottom) with a Franka Emika Robot (on the top), as shown in Fig. 9. The 2D map of the AAU learning factory is generated by LH as shown in Fig. 10.

The VA is expected to obtain the core information (e.g., a package delivery task may require the destination of the delivery and a recipient) through a dialogue with the operator. In such a case, a standard wireless headset is the only extra device we need to communicate between the operator and VA. The operator's verbal command is transmitted through the headset to the VA to control the robot's action.

An team of six composed of computer scientists, a robotics engineer, a mechanical engineer, a lab engineer and robotics student designed, implemented and tested the experiments. The conversation dataset collected for LH crosses 9 intents, *BATTERYCHECK*, *ASKHELP*, *STATESTOP*, *STATERUN*, *DELIVERY*, *GREETING*, *MISSIONCHECK*, *POSITIONCHECK*, *POSITIONUPDATE* according to the type of tasks. The selected tasks and respective intents are presented in Table 2. Each task was reproduced 30 times to verify Max's accuracy and validate its overall performance. The evaluation involves two main levels, language level, devoted to evaluating how well Max can recognize the operator's intent and slot value during the dialogue (Schatzmann et al., 2007), and task level, including task success rate and dialogue time costs (Deriu et al., 2021; Ni et al., 2021). Furthermore, simultaneous task handling ability is also considered to evaluate the scalability of the proposed architecture.

The following are six metrics used to evaluate the performance of Max in these scenarios.

- **Intent Error Rate (IER)**: the model cannot correctly predict the requested intent of operator, for example, the requested intent is *POSITIONCHECK*, but the predicted intent from model might be *POSITIONUPDATE*. IER is obtained by calculating the proportion of it occurring in all experiments.

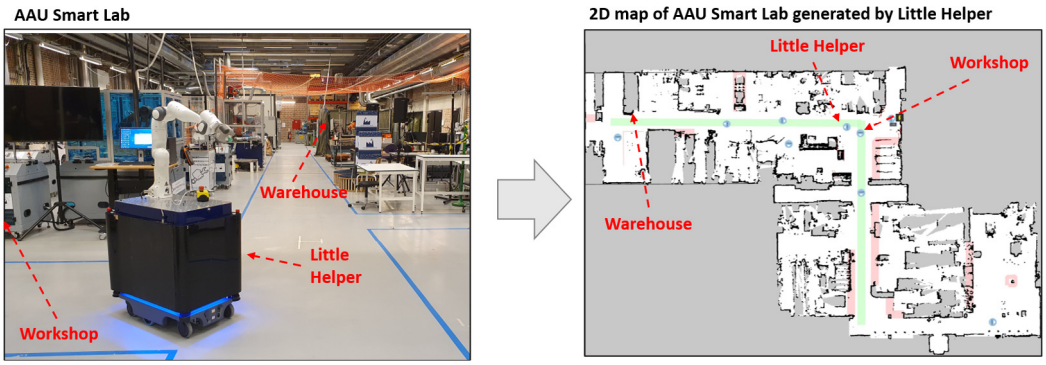

Fig. 10. AAU learning factory and the corresponding 2D map generated by LH. Three locations: Little Helper, workshop and warehouse, are marked on the map.

Table 2
Selected tasks and intents for the three scenarios.

Task ID	Task description	Intent
1	Update the location	*POSITIONUPDATE*
2	Check the location	*POSITIONCHECK*
3	Deliver a package	*DELIVERY*
4	Initial activities	*GREETING*
5	Switch a topic	*ASKHELP*
6	Do not repeat	*MISSIONCHECK*

- **Slot Error Rate (SER)**: including incorrect slot value (the model selects an incorrect value for the slot) and slot detection failed (the model cannot recognize the slot). Similar to IER, the number of occurrences of failing to recognize slot values is used to calculate the SER.
- **Task-Success Rate (TSR)**: measuring how well Max can retrieve all the required information to complete a task without encountering any intent or slot error problem.
- **Average Communication Time (ACT)**: the task completion time is decided by both the robot operation time and the communication time through Max. Since the robot operation time is typically fixed for the same manufacturing task, we use average communication time to evaluate Max's performance. The average communication time is equal to the mean time from the beginning of the conversation to its end for 30 experiments of each task. In general, high intent/slot error rate requires more communication time.
- **Parallel Requests Handling (PRH)**: CS style architecture brings flexibility and scalability to the Max, but it also suffers from traffic congestion. Max's client may face a considerable delay due to Max's server lacking the ability to handle massive simultaneous requests. Thus, parallel request handling is selected to measure Max's parallel processing capacity.
- **Average Service Updating Time (ASUT)**: measures the average time cost for updating the local robot control scripts. ASUT is equal to the meantime from sending a synchronization request to completing an update of the local robot control scripts.

The intent and slot error rates of each Task are reported in Table 4. The overview of the required communication time of the 30 experiments performed for each Task is illustrated in Fig. 11.

3.2. Environment exploration

Exploring the working environment is essential for industrial companies to plan the internal logistics, locate machines/equipment on the shop floor, and calculate the robot's operational capacity.

In this scenario, the operator navigates LH inside the AAU learning factory to explore the working environment and collect information about the positions of shelves and available equipment. Updating and checking the position of the robot on the digital map are identified as two major tasks (see Tasks 1 and 2 in Table 2). The corresponding intents to be tested are *POSITIONUPDATE* and *POSITIONCHECK*.

Task 1: The operator requests Max to add two positions, *warehouse* and *workshop* to the map (see Fig. 10). Max needs 2 turns to obtain the requested intents and slot values as shown in Table 3. After receiving the JSON response from Max's Server, RSM calls LH's RSE to send two HTTP POST requests to add the positions *warehouse* and *workshop* on the map.

Task 2: Max is asked to report LH's location. Max will use its current location to verify the requested slot if the operator provides it in the dialogue. If not, Max will report its location directly in one dialogue turn. Table 3 shows an example of conversations used in Task 2. After receiving the JSON response from Max Server, RSM calls LH's RSE to send an HTTP GET request to retrieve the location.

Max failed to recognize the operator's intent and slot values in eight experiments and five experiments, respectively, for task one. For task two, the corresponding occurrences numbers are six and two separately. Max completed the above two tasks with a rate (TSR) of 0.60 and 0.76, respectively, while requiring 24.70 s and 15.28 s on average to perform the communication. Table 4 reports the corresponding IER, SER, TSR and ACT.

3.3. Package delivery

Internal transportation plays an important role in handling various materials in internal factory logistics. Relocation of goods takes place daily in a warehouse. Therefore, we select package delivery as our second scenario.

In Task 3, LH needs to deliver a box from one place to another according to the operator's verbal instruction.[8] The box pickup place *workshop* and destination *warehouse* are marked on the map generated already in Task 1. The intent *DELIVERY* is tested here, and slot values *recipient*, *to be delivered object*, *object size*, *object colour* and *destination*. Table 3 shows a sample dialogue used during the experiments.

Similar to Tasks 1 and 2, the dialogue finished when Max predicted all the required information (e.g., service name, slot values) for performing the delivery task. RSM then extracts parameters from the JSON response and invokes RSE for LH to send an HTTP POST request (i.e., calling *mission_queue* API) to LH's internal webserver.

[8] https://www.youtube.com/watch?v=XTk7bNCRm94

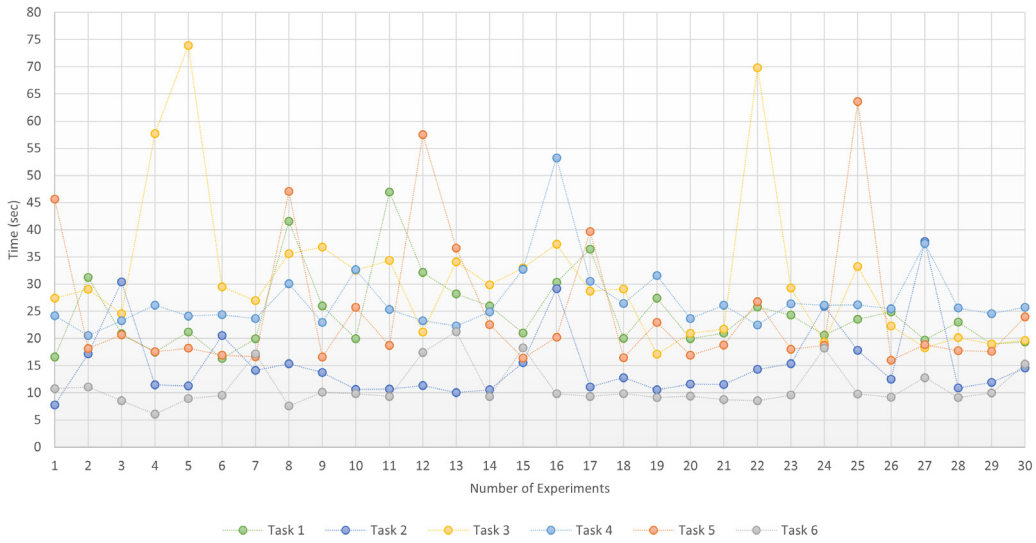

Fig. 11. An overview of the total communication time for all performed experiments.

Table 3
Example dialogues used for Tasks 1, 2 and 3 including dialogue turns, example dialogues and respective slot values.

Task ID	Turn	Dialogue	Slot value
1	1	**Operator**: Can you add this location to the system? Max: Sure, but I need a name for that.	location_name:'None'
	2	**Operator**: warehouse Max: Sure, I will do that.	location_name:'warehouse'
2	1	**Operator**: Max, is this workshop? Max: No, here is the warehouse	location_name:'workshop'
	2	**Operator**: I see. Thanks! Max: Happy to help!	location_name:'workshop'
3	1	**Operator**: Max, would you please deliver this small black box to the warehouse and give it to Kate? Max: Sorry, I did not hear clearly. Who should I give it to?	delivery_object:'box', delivery_object_size:'small', delivery_object_color:'black', delivery_position:'warehouse', delivery_person:'None'
	2	**Operator**: Please give it to Kate. Max: Got it! I will add it on my schedule	delivery_object:'box' delivery_object_size:'small', delivery_object_color:'black', delivery_position:'warehouse', delivery_person:'Kate'

Table 4
Intent error rate, slot error rate, task-success rate and average communication time (in seconds) for each task.

Task ID	IER	SER	TSR	ACT
1	0.26	0.16	0.60	24.70
2	0.2	0.06	0.76	15.28
3	0.1	0.43	0.50	31.08
4	0.2	None	0.80	27.07
5	0.3	None	0.70	25.04
6	0.17	None	0.83	11.12

Max failed to recognize the operator's intent three times and the slot value 13 times. Due to the complexity of the task and the ambient noise, Max achieved a completion rate of only 0.50 and required the highest communication time of 31.08 s on average (see Table 4).

3.4. Embedded conversation strategy

In this scenario, we evaluate Max's performance focusing on conversation strategies (Tasks 4, 5, and 6).[9] These tasks test specifically whether the embedded human–human conversation strategies can bootstrap the user experience by improving the user's engagement.

In Task 4, Max can initialize activities (e.g., report the currently scheduled tasks) when it detects the operators' greetings. For Task 5, Max provides task-related options to the operator when it cannot perform the requested tasks. Task 6 tests if Max can provide different responses for the same request. Table 5 shows the dialogues used for these three tasks.

In these experiments, Max's client remains in standby mode until it receives confirmation from the operator. The intent error rate varies depending on the complexity of the dialogue, and the task completion rate is 0.80, 0.70, and 0.83 per task. Similarly, the average communication time varies noticeably from 27.07 to 25.04 and 11.12 s, respectively (see Table 4).

3.5. Parallel request handling

Additionally, to evaluate the overall parallel processing capacity, we run stress tests on Max's local server and the AAU Cloud using Siege,[10] an HTTP load testing and benchmarking utility.

9 https://www.youtube.com/watch?v=lYnh2cOeeE0

10 Siegelink.

Table 5
Example dialogues used for Tasks 4, 5 and 6.

Task ID	Dialogue
4	**Operator**: Max, what's up? Max: I am good, thanks. By the way, three delivery tasks are on my list. Would you like me to do it now?
5	**Operator**: Hey Max, can you give me a hand? Max: Well, my battery level is less than 20%. I can only help you to do some position updating tasks. What do you say?
6	**Operator**: Max, are you working on something? Max: Well, I am totally free now **Operator**: Max, are you busy? Max: Em, nothing on my schedule now

Table 6
The stress test results of Max server including local server and AAU Cloud.

Server type	Transactions	Elapsed time	Transaction rate	Concurrency	Throughput
Local Server	2000	10.09 s	198.22 trans/s	2.01	0.16 MB/s
AAU Cloud	100,000	7.04 s	14204.55 trans/s	289.79	11.65 MB/s

The tests focus on the total elapsed time for the given number of transactions, the transaction rate, the actual maximum concurrent number of the connections, and the throughput. Table 6 shows the test results of the local server and the AAU Cloud.

3.6. Service updating time

Finally, to evaluate the ASUT, another 30 experiments were conducted. The tests focus on calling the functions which are not defined in the local robot control script. For the 30 experiments, the minimum, maximum, and average service updating times were 5.6, 6.3, and 5.7 s respectively, measured on the local server.

4. Discussion & conclusions

The proposed natural language-enabled VA, Max, benefits from the CS-style architecture, RESTful style APIs, and centralized management, enabling high efficiency in multiple HRI scenarios in industrial robots. With the addition of the industrial robot service, the robot control agent can also interact efficiently with various industrial robots. Though our model can reach a high inference accuracy, the response latency is observed. The inaccurate interpretation of the operators' command may lead to serious safety issues, especially when robots share the same workspace. Therefore, emergency control (e.g., stop movement) is also needed in the real industrial environment.

We observe that the AAU Cloud maintains a high concurrent processing capability comparing to the local server. Therefore, industrial parties who wish to incorporate Max's abilities into their industrial shop floor should consider the need to dedicate significant processing power for handling many parallel requests.

Moreover, a well-designed security strategy is needed for communication between the Max client and server. In these small-scale experiments, communication is based on unencrypted HTTP protocols. Naturally, the situation in a real manufacturing scenario will be different and potentially affect Max's performance.

Additionally, we can observe that Max's performance is highly relevant to the intent/slot error rate and ambient noise levels. The high intent/slot error rate directly influences the overall task completion time by increasing the communication time. Although our model achieves a high intent/slot accuracy theoretically, the experiments conducted in the AAU learning factory, under ambient noise levels of 69 dB(A) on average, demonstrated a relatively high intent/slot error rate. As an indication, experiments for Task 3 in an office environment, with an average ambient noise level of 35.8 dB(A), had only 0.03 intent/slot error rates.

Furthermore, other factors, e.g., the operator's accent and voice volume, also influence the intent/slot error rate. Future work will focus on ways to suppress the ambient noise and enhance speech so as the VA and, consequently the robot, can effectively communicate with the workers with fewer interruptions and errors.

The encouraging results based on the two embedded conversation strategies prove the Max can support an active interaction during various manufacturing tasks. It provides task-related suggestions, successfully attracts the operator's attention, and forms a diverse and thoughtful dialogue to improve user engagement in HRI for industrial robots. An extensive user study was postponed due to COVID-19 restrictions; however, it remains a central part of our future work to collect feedback on the naturalness and coherence of Max's generated dialogue and responses.

CRediT authorship contribution statement

Chen Li: Methodology, Conceptualization, Investigation, Project administration, Software, Writing – original draft, Writing – review & editing. **Dimitris Chrysostomou:** Methodology, Conceptualization, Investigation, Resources, Writing – original draft, Writing – review & editing. **Hongji Yang:** Resources, Writing – review & editing.

Declaration of competing interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests: Hongji Yang reports financial support was provided by University of Leicester. Hongji Yang reports a relationship with University of Leicester that includes: employment.

Data availability

We have uploaded the data/code to a GitHub repository, https://github.com/lcroy/Virtual-Assistant-Max.

Acknowledgements

The authors would like to acknowledge support by Aalborg University Bridging Project (A Multimodal Attention Tracking in Human-Robot Collaboration for Manufacturing Tasks), EU's SMART EUREKA programme under grant agreement S0218-ch-ARmER and the H2020-WIDESPREAD project no. 857061 "Networking for Research and Development of Human Interactive and Sensitive Robotics Taking Advantage of Additive Manufacturing – R2P2".

References

Aceto, G., Persico, V., Pescapé, A., 2019. A survey on information and communication technologies for industry 4.0: State-of-the-art, taxonomies, perspectives, and challenges. IEEE Commun. Surv. Tutor. 21 (4), 3467–3501. http://dx.doi.org/10.1109/COMST.2019.2938259.

Arana-Arexolaleiba, N., Urrestilla-Anguiozar, N., Chrysostomou, D., Bøgh, S., 2019. Transferring human manipulation knowledge to industrial robots using reinforcement learning. Procedia Manuf. 38, 1508–1515. http://dx.doi.org/10.1016/j.promfg.2020.01.136.

Ayres, R.U., 2005. On the Reappraisal of Microeconomics: Economic Growth and Change in a Material World. Edward Elgar Publishing.

Banchs, R.E., Li, H., 2012. IRIS: A chat-oriented dialogue system based on the vector space model. In: Proceedings of the ACL 2012 System Demonstrations. Association for Computational Linguistics, pp. 37–42, URL https://www.aclweb.org/anthology/P12-3007.

Bingol, M.C., Aydogmus, O., 2020. Performing predefined tasks using the human–robot interaction on speech recognition for an industrial robot. Eng. Appl. Artif. Intell. 95, 103903. http://dx.doi.org/10.1016/j.engappai.2020.103903.

Bobrow, D.G., Kaplan, R.M., Kay, M., Norman, D.A., Thompson, H., Winograd, T., 1977. GUS, a frame-driven dialog system. Artif. Intell. 8 (2), 155–173. http://dx.doi.org/10.1016/0004-3702(77)90018-2.

Buhl, J.F., Grønhøj, R., Jørgensen, J.K., Mateus, G., Pinto, D., Sørensen, J.K., Bøgh, S., Chrysostomou, D., 2019. A dual-arm collaborative robot system for the smart factories of the future. Procedia Manuf. 38, 333–340. http://dx.doi.org/10.1016/j.promfg.2020.01.043.

Cheng, Y., Li, D., Wong, W.E., Zhao, M., Mo, D., 2022. Multi-UAV collaborative path planning using hierarchical reinforcement learning and simulated annealing. Int. J. Performabil. Eng. 18 (7).

Chesneau, B., Davis, P., Peksag, B., Leeds, R., 2020. Gunicorn. Obtenido de http://gunicorn.org.

Deriu, J., Rodrigo, A., Otegi, A., Echegoyen, G., Rosset, S., Agirre, E., Cieliebak, M., 2021. Survey on evaluation methods for dialogue systems. Artif. Intell. Rev. 54 (1), 755–810. http://dx.doi.org/10.1007/s10462-020-09866-x.

Devlin, J., Chang, M.-W., Lee, K., Toutanova, K., 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805.

González-Docasal, A., Aceta, C., Arzelus, H., Álvarez, A., Fernández, I., Kildal, J., 2020. Towards a natural human-robot interaction in an industrial environment. In: Conversational Dialogue Systems for the Next Decade. Springer, pp. 243–255. http://dx.doi.org/10.1007/978-981-15-8395-7_18.

Grinberg, M., 2018. Flask Web Development: Developing Web Applications with Python. " O'Reilly Media, Inc.".

Hjorth, S., Chrysostomou, D., 2022. Human–robot collaboration in industrial environments: A literature review on non-destructive disassembly. Robot. Comput.-Integr. Manuf. 73, 102208. http://dx.doi.org/10.1016/j.rcim.2021.102208.

Jost, C., Le Pévédic, B., Belpaeme, T., Bethel, C., Chrysostomou, D., Crook, N., Grandgeorge, M., Mirnig, N. (Eds.), 2020. Human-robot interaction : evaluation methods and their standardization, Vol. 12. Springer, p. 385. http://dx.doi.org/10.1007/978-3-030-42307-0.

Knight, W., 2021. In: Weird.com (Ed.), As Robots Fill the Workplace, They Must Learn to Get Along. URL https://www.wired.com/story/robots-fill-workplace-must-learn-get-along/. (Accessed 02 Feb 2021).

Kumar, S., Savur, C., Sahin, F., 2021. Survey of human–robot collaboration in industrial settings: Awareness, intelligence, and compliance. IEEE Trans. Syst. Man Cybern.: Syst. 51 (1), 280–297. http://dx.doi.org/10.1109/TSMC.2020.3041231.

Li, C., Park, J., Kim, H., Chrysostomou, D., 2021. How can I help you? An intelligent virtual assistant for industrial robots. In: HRI '21 Companion, Association for Computing Machinery, New York, NY, USA, pp. 220–224. http://dx.doi.org/10.1145/3434074.3447163.

Li, C., Yang, H.J., 2021. Bot-x: An AI-based virtual assistant for intelligent manufacturing. Multiagent Grid Syst. 17 (1), 1–14. http://dx.doi.org/10.3233/MGS-210340.

Lindblom, J., Alenljung, B., Billing, E., 2020. Evaluating the user experience of human–robot interaction. In: Human-Robot Interaction. Springer, pp. 231–256. http://dx.doi.org/10.1007/978-3-030-42307-0_9.

Lithoxoidou, E., Doumpoulakis, S., Tsakiris, A., Ziogou, C., Krinidis, S., Paliokas, I., Ioannidis, D., Votis, K., Voutetakis, S., Elmasllari, E., et al., 2020. A novel social gamified collaboration platform enriched with shop-floor data and feedback for the improvement of the productivity, safety and engagement in factories. Comput. Ind. Eng. 139, 105691. http://dx.doi.org/10.1016/j.cie.2019.02.005.

Littlejohn, S.W., Foss, K.A., 2010. Theories of Human Communication. Waveland Press.

Maksymova, S., Matarneh, R., Lyashenko, V., Belova, N., 2017. Voice control for an industrial robot as a combination of various robotic assembly process models. J. Comput. Commun. http://dx.doi.org/10.4236/jcc.2017.511001.

Mavridis, N., 2015. A review of verbal and non-verbal human–robot interactive communication. Robot. Auton. Syst. 63, 22–35. http://dx.doi.org/10.1016/j.robot.2014.09.031.

Mavridis, N., Roy, D., 2006. Grounded situation models for robots: Where words and percepts meet. In: 2006 IEEE/RSJ International Conference on Intelligent Robots and Systems. IEEE, pp. 4690–4697. http://dx.doi.org/10.1109/IROS.2006.282258.

Nardello, M., Madsen, O., Møller, C., 2017. The smart production laboratory: A learning factory for industry 4.0 concepts. In: CEUR Workshop Proceedings, Vol. 1898. CEUR Workshop Proceedings.

Ni, J., Young, T., Pandelea, V., Xue, F., Adiga, V., Cambria, E., 2021. Recent advances in deep learning based dialogue systems: A systematic survey. arXiv preprint arXiv:2105.04387.

Pérez, L., Rodríguez-Jiménez, S., Rodríguez, N., Usamentiaga, R., García, D.F., Wang, L., 2020. Symbiotic human–robot collaborative approach for increased productivity and enhanced safety in the aerospace manufacturing industry. Int. J. Adv. Manuf. Technol. 106 (3), 851–863. http://dx.doi.org/10.1007/s00170-019-04638-6.

Prati, E., Peruzzini, M., Pellicciari, M., Raffaeli, R., 2021. How to include user experience in the design of human-robot interaction. Robot. Comput.-Integr. Manuf. 68, 102072. http://dx.doi.org/10.1016/j.rcim.2020.102072.

Raj, S., Kalia, T., Aggarwal, S., Jaglan, S., Nijhawan, N., Sharma, M., 2022. Human computer interaction using virtual user computer interaction system. Int. J. Performabil. Eng. 18 (6).

Reese, W., 2008. Nginx: The high-performance web server and reverse proxy. Linux J. 2008 (173), 2.

Schatzmann, J., Thomson, B., Young, S., 2007. Error simulation for training statistical dialogue systems. In: 2007 IEEE Workshop on Automatic Speech Recognition & Understanding. ASRU, IEEE, pp. 526–531. http://dx.doi.org/10.1109/ASRU.2007.4430167.

Schou, C., Andersen, R.S., Chrysostomou, D., Bøgh, S., Madsen, O., 2018. Skill-based instruction of collaborative robots in industrial settings. Robot. Comput.-Integr. Manuf. 53 (June 2016), 72–80. http://dx.doi.org/10.1016/j.rcim.2018.03.008.

Sheikholeslami, S., Moon, A., Croft, E.A., 2017. Cooperative gestures for industry: Exploring the efficacy of robot hand configurations in expression of instructional gestures for human–robot interaction. Int. J. Robot. Res. 36 (5–7), 699–720. http://dx.doi.org/10.1177/0278364917709941.

Székely, M., Powell, H., Vannucci, F., Rea, F., Sciutti, A., Michael, J., 2019. The perception of a robot partner's effort elicits a sense of commitment to human-robot interaction. Interact. Stud. 20 (2), 234–255. http://dx.doi.org/10.1075/is.18001.sze.

van Deemter, K., Krahmer, E., Theune, M., 2005. Squibs and discussions: Real versus template-based natural language generation: A false opposition? Comput. Linguist. 31 (1), 15–24. http://dx.doi.org/10.1162/0891201053630291, URL https://www.aclweb.org/anthology/J05-1002.

Villani, V., Pini, F., Leali, F., Secchi, C., 2018. Survey on human–robot collaboration in industrial settings: Safety, intuitive interfaces and applications. Mechatronics 55, 248–266. http://dx.doi.org/10.1016/j.mechatronics.2018.02.009.

Warren, M., 2006. Features of Naturalness in Conversation, Vol. 152. John Benjamins Publishing, http://dx.doi.org/10.1075/pbns.152.

Williams, J.R., 1998. Guidelines for the use of multimedia in instruction. Proc. Hum. Factors Ergon. Soc. Ann. Meet. 42 (20), 1447–1451. http://dx.doi.org/10.1177/154193129804202019.

Yu, Z., Xu, Z., Black, A.W., Rudnicky, A., 2016. Strategy and policy learning for non-task-oriented conversational systems. In: Proceedings of the 17th Annual Meeting of the Special Interest Group on Discourse and Dialogue. Association for Computational Linguistics, pp. 404–412. http://dx.doi.org/10.18653/v1/W16-3649, URL https://www.aclweb.org/anthology/W16-3649.

Chen li received the M.S. degree in computer application from the University of Shanghai for Science and Technology, Shanghai, China, in 2010, and Ph.D. degree in computer science and technology from the Shanghai Jiao Tong University, Shanghai, China, in 2015.

From 2015 to 2016, he was a Research Assistant in Centre for Creative Computing at Bath Spa University. From 2016 to 2018, he was a Research Associate in Computer Science department at Imperial College London. From

2018 to 2020, he held a postdoctoral position at Department of Materials and Production, Aalborg University, where he has been an Assistant Professor since 2020. His research interests include Natural Language Processing, Human-Robot Interaction and System Modelling.

Dimitris Chrysostomou received his Diploma degree in production engineering, in 2006, and the Ph.D. degree in robot vision from Democritus University of Thrace, Greece, in 2013. Since 2013 he has been working with the Robotics and Automation Group of the Department of Materials and Production, Aalborg University, Denmark as a postdoctoral researcher from 2013–2016, an Assistant Professor from 2016–2019 and since 2020 as an Associate Professor. His research interests include robot vision, skill-based programming and human–robot interaction for intelligent robot assistants.

Hongji yang received the B.S. and M.S. degrees in computer science from Jilin University, Changchun, China, in 1982 and 1985, respectively, and the Ph.D. degree in computer science from Durham University, Durham, U.K., in 1994. He is working at the School of Computing and Mathematica Sciences, University of Leicester. His main research interests include knowledge modelling and creative computing. He has published over 500 papers. He became a Golden Core Member of the IEEE Computer Society in 2010.