HERO: An Artificial Conversational Assistant to Support Humans in Industrial Scenarios

Claudia Bonanno[1], Francesco Ragusa[1,2], Rosario Leonardi[1], Antonino Furnari[1,2] and Giovanni Maria Farinella[1,2]

[1]*FPV@IPLAB, DMI - University of Catania, Italy*

[2]*Next Vision s.r.l. - Spinoff of the University of Catania, Italy*

Keywords: First Person Vision, Object Detection, Chatbot, Conversational Agent, Visual Question Answering.

Abstract: We present HERO, a Conversational Intelligent Assistant to support workers in industrial domains. The proposed system is able to interact with humans using natural language and observing the surrounding world in order to avoid the language ambiguity. HERO is composed of four modules: 1) the input module to process both text and visual signals, 2) the NLP module to predict user intent and extract relevant entities from text, 3) the object detection module to extract entities by analyzing images captured by the user and 4) the output module which is responsible for choosing the best answer to send to the user. To assess its usefulness in a real scenario, the proposed system is implemented and evaluated in an industrial laboratory setting. Preliminary experiments show that HERO achieves good performance in predicting intents and entities exploiting both text and visual signals.

1 INTRODUCTION

Artificial Intelligence (AI) allows to build systems which enable automatic conversation between humans and machines using natural language. These conversational agents could be integrated in more complex systems with the aim to support humans where they live and work. Generally, conversational AI systems can be grouped into three categories (Gao et al., 2018): 1) task-oriented systems, 2) chat-oriented systems and 3) question answering systems. The first category comprises all systems that aim to address specific tasks such as booking a table at a restaurant or ordering a pizza, while chatbots are designed to handle common queries (e.g., "tell me a joke") or to simulate conversation with a virtual friend (Fedorenko et al., 2017). Given a user's query, question answering bots are able to exploit previously collected knowledge and resources to find the most appropriate answer.

Nowadays, well-known hybrid systems, belonging to both chatbot and question answering categories, are exploited in daily-life applications such as Apple's Siri (Apple, 2012) (e.g., to ask her to call someone), Microsoft's Cortana (Microsoft, 2014) (e.g., to open a specific app in our personal computer) or Amazon's Alexa (Amazon, 2014) (e.g., to receive information about the weather or to listen to your favourite songs). Conversational agents are used also in the context of marketing or customer care, providing several functions such as answering frequently asked questions, resolving customer queries or recommending new offers in an automatic way. Conversational agents are used as well in the kitchens domain, where they can support humans for maintenance of the coffee machine (Mleczko, 2021).

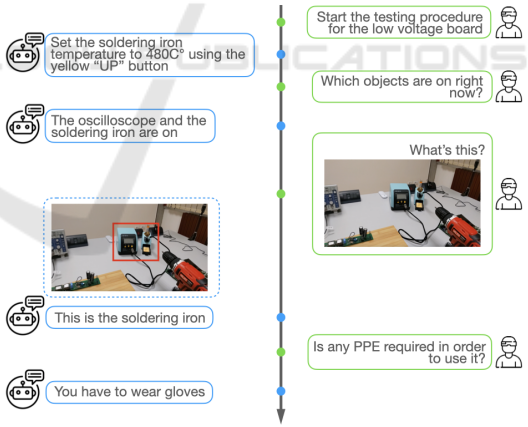

Figure 1: Concept of the proposed AI Assistant HERO.

Bonanno, C., Ragusa, F., Leonardi, R., Furnari, A. and Farinella, G.
HERO: An Artificial Conversational Assistant to Support Humans in Industrial Scenarios.
DOI: 10.5220/0011318600003289
In *Proceedings of the 19th International Conference on Signal Processing and Multimedia Applications (SIGMAP 2022)*, pages 86-93
ISBN: 978-989-758-591-3; ISSN: 2184-9471

While these solutions are widely employed, they suffer from natural language ambiguity (or diversity) due to the fact that they cannot see the surrounding environment. Recently Amazon's Alexa has a new available feature called "Show and Tell" (Amazon, 2019) which allows visually impaired users to exploit the camera of the smartphone/tablet to recognize the objects in the hands of the user. Also in the context of kitchens, a conversational agent which can see through a mobile phone's camera has been developed to assist visually impaired people to understand how long to cook a microwave meal or if it contains an ingredient they are allergic to (Brick et al., 2021). Despite the progress made in this field, there are not solutions able to support workers in the industrial domain, e.g., assisting them during the maintenance operations "What is the next step?") or providing continuous training ("How can I use this tool?").

In this paper, we present HERO, an AI Assistant that, through the interaction with a Conversational Agent equipped with artificial vision allows the user to request information about the environment or the objects (e.g., How can I use this object?) using natural language, as well as images acquired from the user's point of view using wearable devices (see Figure 1). HERO is based on the Egocentric Vision paradigm, which offers a convenient setting to collect visual information about the user's activities and the interactions with objects and can be useful to remove the ambiguity present in the human's questions. Specifically, we assume that the user interacting with HERO is equipped with a wearable camera able to collect visual information from their point of view. By means of its direct access to the egocentric camera, HERO can exploit both language and vision to interact with the user. HERO is composed of four main modules: 1) the input module which takes both text/audio and visual signals, 2) the NLP module which analyzes the text/audio signal to understand the user's questions, 3) the object detection module which processes the input images acquired from the user's point of view and recognizes all the objects present in the surrounding environment and 4) the output module which combines the obtained information to generate the most reasonable answer to the human's question. The proposed system has been tested in a real industrial laboratory (Leonardi et al., 2021) where workers performed different maintenance operations with 2 electrical boards and 21 industrial tools. To implement HERO in this domain, we considered 26 different worker's intents such as "Which objects are there?" or "Turn off the oscilloscope", which HERO should exploit to generate the correct answer and give the needed support. Preliminary experiments show that the proposed system achieves good performance in the tasks of object detection and natural language understanding in the considered industrial domain. The remainder of the paper is organized as follows. Section 2 reports the related work. Section 3 presents the collected datasets in the industrial laboratory. The architecture of HERO and its services are discussed in Section 4, while the experimental results are reported in Section 5. Section 6 concludes the paper.

2 RELATED WORK

Natural Language Understanding (NLU). Several works investigated how to perform text classification (Joachims, 1998; Fan et al., 2008; Zhang et al., 2015). In particular, different word embedding approaches have been proposed in literature for classification purposes. The authors of (Mikolov et al., 2013) suggested *Word2Vec*, a vectorial representation of words for learning high-quality text representations considering huge datasets. The authors of (Joulin et al., 2017) proposed a simple and efficient baseline called *fastText* able to classify millions of sentences represented as bags of n-grams. *GloVe* (Global Vectors) has been proposed in (Pennington et al., 2014) with the aim to produce a vector space which is generic with respect to the variations in phrasing.

Text classification is a key task to build a Dialog Management Control system which is able to understand the human's natural language (Young et al., 2013). Most task-oriented systems are able to perform end-to-end learning to build natural conversations in specific domains (Wen et al., 2015; Bordes et al., 2017). To overcome this limitation, the authors of (Williams et al., 2017) proposed a end-to-end learning RNN which reduced the amount of training data required to adapt the dialog control system in specific domains. Also the authors of (Williams and Liden, 2017) proposed an interactive "teaching" systems which allows a developer to teach the network by interacting with the system and providing on-the-spot corrections. Recently, the authors of (Bunk et al., 2020) proposed the Dual Intent and Entity Transformer (DIET), which is a multi-task architecture for intent classification and entity recognition. DIET has the ability to generalize to different domains by incorporating pre-trained word embeddings from language models and combining these with sparse word- and character-level n-gram features. To allow HERO to understand the human's language and to answer the human's questions, we designed a NLU module which contains a DIET module, considering a fixed set of possible requests which the user can ask.

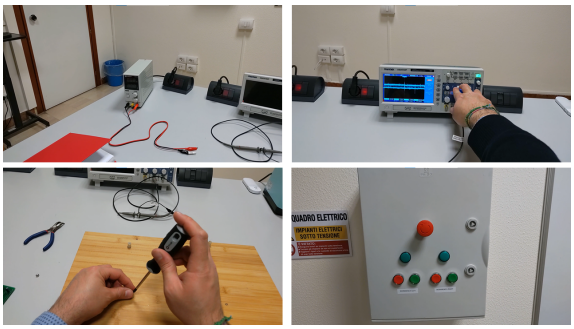

Figure 2: Sample images of the industrial laboratory with bounding box annotations for the objects present in the scene.

NLU Software Libraries. There are different software libraries publicly available to build high-quality conversational agents. PyDeal (Ultes et al., 2017) is an open source end-to-end statistical spoken dialogue system toolkit which offers easy configurations and domain-independent implementations of several dialogue system modules. To avoid the need of specialist software developers, several frameworks have been released with the aim to build conversational agents in a easy way such as OpenDial (Lison and Kennington, 2016), RavenClaw (Bohus and Rudnicky, 2009), AT&T (Williams et al., 2010), InproTKs (Kennington et al., 2014) and IrisTK(Skantze and Moubayed, 2012). In particular, RASA NLU and RASA Core have been proposed in (Bocklisch et al., 2017) and are open source Python libraries for building conversational software. The main difference with respect to the other works is the architecture which has been implemented as common APIs, which can be easily integrated in other systems. We considered the RASA framework to develop our NLP module.

Object Detection and Recognition. Given an image, the aim of an object detector is to detect and recognize all the objects present in the input image. In the literature, there are two main groups of object detectors: 1) Two-stages detectors and 2) One-stage detectors. Two-stage methods predict a set of candidate bounding boxes in the first step, while, in a second step, they extract visual features from each box and use them to classify boxes into one of the considered object classes and regress accurate bounding box coordinates. These detectors (Girshick et al., 2014; Girshick, 2015; Ren et al., 2015) favor the accuracy of the predictions at the expenses of computation speed. Differently, one-stage methods perform object localization and classification in a single step, allowing for a faster computation, but often with a reduced accuracy (Liu et al., 2016; Redmon and Farhadi, 2018; Bochkovskiy et al., 2020). Due to its more accurate results, we choose the two-stage detector Faster R-CNN (Ren et al., 2015) to build our Object Detection module.

- **intent:** *object_warnings*
  examples:
    - show me the warnings related to the [oscilloscope]**(object)**
    - what are the alerts for the [power supply]**(object)**?
    - warnings for the [welder]**(object)**?
    - are there warnings for this object?
    - what warnings are there for this?
    - has the [electric screwdriver]**(object)** any warnings?

- **intent:** start
  examples:
    - start [low voltage]**(board)** electronic board [repair]**(procedure)** procedure
    - start [high voltage]**(board)** electronic board [test]**(procedure)** procedure
    - start [test]**(procedure)** [low voltage]**(board)** electronic board
    - begin the [repair]**(procedure)** of the [high voltage]**(board)** electronic board

Figure 3: Samples of the structured data used to train the NLU module. For each intent (blue) we designed different examples including different entities (red, green and purple).

3 INDUSTRIAL LABORATORY

We considered the realistic industrial context proposed in (Leonardi et al., 2021), where an industrial laboratory has been set up. The authors acquired 8 real egocentric videos using a Microsoft Hololens 2 device in which subjects interact with 23 objects following different sequences of operations related to *test* and *repair* procedures on two electrical boards (e.g., turning on the power supply or pushing the red button on the electrical panel). Figure 2 shows sample images of the industrial laboratory where HERO is able to support workers performing several operations.

3.1 Dataset for the NLP Module

We built a dataset to train our NLP module considering the industrial laboratory. We structured information about the user messages following the *intent-entity* concepts. The intent is what the user plans to do or accomplish (e.g., greeting, turn on objects, etc.), while the entities are the key pieces of information that can be extracted from the message (e.g., phone number, location, objects, etc.). Considering the videos and the operations performed by the users, we defined 26 different intents and 4 entities. The 4 entities are the following:

- *object*: is related to the object's class of the industrial laboratory;
- *electronic board*: it can be high or low voltage electronic board;
- *component*: several objects are composed of different components (e.g., the high voltage elec-

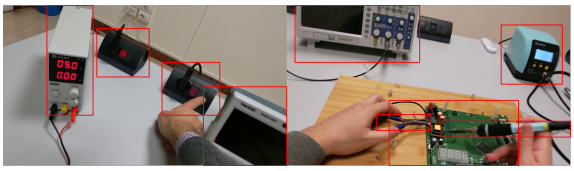

Figure 4: Some images with bounding box annotations for the objects present in the scene.

tronic board has a display);

- *procedure*: indicates a procedure that the subject can perform. There are two kinds of procedures: *testing* and *repair*.

For each intent, we defined several examples which represent different questions that the user can make referring to the same intent. For example, the sentences *How can I use the oscilloscope?* and *I can't use the power supply* are different examples belonging to the same intent (i.e., *object's instructions*). Since HERO is able to see what the user is looking at, the user does not need to ask questions explicitly including the name of the objects, for example when he does not know its name (e.g., *How can I use this object?*). At the end, we obtained 136 examples related to the 26 intents. Figure 3 reports some intents with the related examples.

3.2 Dataset for the Object Detection Module

We used the dataset proposed in (Leonardi et al., 2021), which includes 8 videos acquired using a Microsoft Hololens 2 wearable device at a resolution of 2272×1278 pixels and with a framerate of $30 fps$. For each of these videos, the authors selected the first frame where the hand touches an object and the frame after the hand releases it. For each extracted frame, the objects visible in the image have been annotated with (x, y, w, h, c) tuples, where c indicates the object class among the 19 object categories and (x, y, w, h) are the bounding box coordinates. Following this procedure, we extended the set of annotations from 19 to 23 object classes, manually annotating all the object instances belonging to the following 4 object categories: *power supply cables, ground clip, battery charger connector* and *panel A*. With this procedure, we obtained 20000 labeled objects belonging to 23 objects categories. Figure 4 shows some examples of the annotated frames.

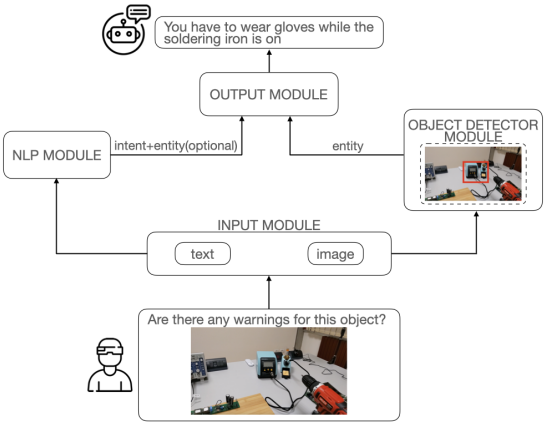

Figure 5: The architecture of the HERO system.

4 PROPOSED SYSTEM

Figure 5 illustrates the architecture of the HERO system. It is composed of four main components: 1) input module, 2) NLP module, 3) object detection module and 4) output module. We describe the modules of HERO in the following sections.

4.1 Input Module

The input module is responsible for taking different inputs from the user and sending them to the corresponding modules. In particular, this module takes in input the text which will be processed by the NLP module and the images acquired by a camera from the first point of view which will be sent to the object detection module.

4.2 NLP Module

This module is based on the RASA framework (Bocklisch et al., 2017) and it is responsible for managing conversations between humans and the AI assistant. In the considered scenario, the user follows a procedure (i.e., testing or repairing) composed of several sequential operations. At any point of the procedure the AI assistant must be able to change context and answer other questions related to the different intents (e.g., *object's instructions*). In RASA, a general conversation can be represented by *stories* or *rules*, a type of data, which are used to train the assistant's dialogue management module. Since in real cases, such as in the considered industrial context, a complex conversation between a worker and the AI agent can be composed of different question-answer couples, it would be too expensive to add a story for each different question-answer instance. For

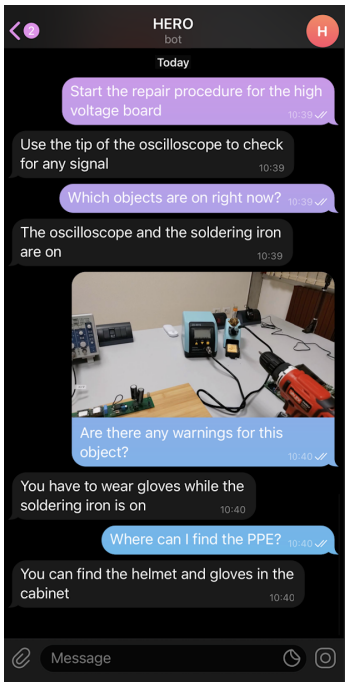

Figure 6: A sample conversation with our HERO system through a Telegram bot.

this reason, we represented the data as *rules* which describe short pieces of conversations that should always follow the same path. The output of the NLP module is made of the predicted *intent* and the *entities*. The entities can sometimes be absent (e.g., if the user does not know the name of an object, the NLP module will not able to extract this information from the text). The intent represents the main intention of the user when they ask a question. The intent information will be combined with the extracted entities in order to choose the correct response. In this industrial scenario, 4 of the 26 considered intents (i.e., *present_objects, can_use_objects, powered_objects, where_PPE*) do not need any entities nor any context in order to generate the answer. These 4 intents have been grouped into a macro-intent called *FAQ*. Using this scheme, the DIETClassifier (Bunk et al., 2020) predicts whether each intent is a regular intent or a FAQ. In the latter case, the ResponseSelector further classifies the FAQ macro-intent into one of the four related intents (*present_objects, can_use_objects, powered_objects, where_PPE*).

4.2.1 NLU Pipeline

Our NLU Pipeline is composed of several components which are described in the following:

- *SpacyNLP*: This component is needed in order to initialize Spacy structures[1];
- *SpacyTokenizer*: This component segments each message into words, punctuation and so on using specific rules related to the used language model, thus obtaining tokens;
- *CountVectorsFeaturizer (CVF)*: This component creates bag-of-words representation of user messages, intents, and responses. Two instances of CVF are used. The former analyzes messages at word-level, whereas the latter analyzes messages at character-level. We consider n-grams of lengths ranging from 1 to 4.
- *SpacyFeaturizer*: This component creates a vector representation of user messages and responses.
- *DIETClassifier (Dual Intent Entity Transformer) (Bunk et al., 2020)*: This component is a multi-task architecture for intent classification and entity recognition. The architecture is based on a transformer which is shared for both tasks;
- *EntitySynonymMapper*: This component is responsible of the synonymous entities. If the training data contains defined synonyms, this component will make sure that detected entity values will be mapped to the same value;
- *ResponseSelector*: This component chooses the response from a set of possible responses for those intents that were grouped in a macro-intent;

4.3 Object Detection Module

Our Object Detection module relies on the two-stage object detector Faster-RCNN (Ren et al., 2015) to address object detection and recognition. Given an image from the Input module, the object detector predicts a (x, y, w, h, c) tuple for each detected object, where c indicates the class belonging to the 23 object categories and (x, y, w, h) are the bounding box coordinates. Since humans usually focus their attention on the center of the image when they are interacting with an object, we filter the predictions choosing only the object closest to the center of the image. The class of the predicted object represents an *entity* and it will be used by the output module.

4.4 Output Module

This module chooses the final answer to send to the user considering the intent and the entities predicted by the NLP module and the entity detected by the object detection module.

[1]The RASA framework is based on the Spacy library: https://spacy.io/

Table 1: Results obtained by the NLP module considering the intent and entity classification of the DIETClassifier (first and second row) and the response selection performed by the ResponseSelector (last row).

	Precision	**Accuracy**	**F_1-score**
Entity classification	0.944	0.959	0.866
Intent classification	0.729	0.735	0.700
Response Selection	0.800	0.867	0.822

5 PRELIMINARY EXPERIMENTS AND RESULTS

To evaluate HERO, we assess the performance the NLP and object detection modules. In particular, we consider two tasks: 1) intent-entity recognition and 2) object detection and recognition. For a preliminary evaluation we designed channel connectors where the user can send and receive messages. In particular, we integrated a channel connector through the use of a Telegram bot where the input signals are the text written in a chat by the user and the images is acquired by the camera phone (see Figure 6).

5.1 NLP Module

We split the dataset into training and test sets with an 80 : 20 ratio. Considering the small dimension of the dataset, we find the optimal model's parameters performing a 5-fold cross-valuation on the training set. We trained the final model on the whole training set considering the mean of the parameters of each split. The model has been trained on an Intel Core i5 CPU for 100 epochs with learning rate of 0.001 and a variable batch size which linearly increases for each epoch from 64 to 256. We evaluate our model on the test set using precision, accuracy and F_1-score measures. Table 1 reports the results obtained on the entity classification (first row) and intent classification (second row) using the DIETClassifier module, as well as the response selection performed by the ResponseSelector module (last row). The DIETClassifier recognizes very well the entities considering all the evaluation measures with a precision of 0.944, an accuracy of 0.959 and an F_1-score of 0.866. The performances of the intent classification are lower, with a precision of 0.729, an accuracy of 0.735 and a F_1-score of 0.700. The ResponseSelector module obtains good performance considering the classification of the 4 micro-intents contained in the FAQ group, achieving a precision of 0.800, an accuracy of 0.867 and an F_1-score of 0.822.

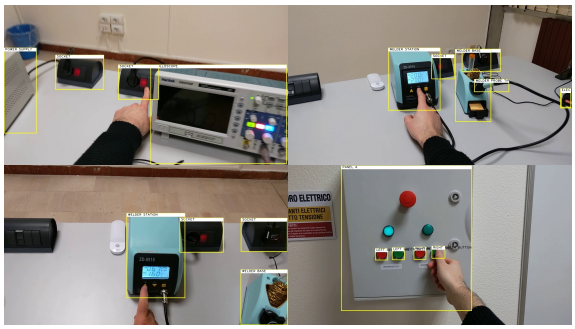

Figure 7: Qualitative results of the object detector.

Table 2: Splits of the dataset reporting the number of videos, images and objects for each split.

	Train	**Validation**	**Test**
Videos	4	2	2
Images	1,367	832	857
Objects instances	7,981	6,009	6,010

5.2 Object Detection Module

We split the object detection dataset into training, validation and test sets being careful to include all the frames extracted from a video in a single split. Table 2 reports statistics about these splits considering the number of videos, images and objects included in each split.

As object detector we used Faster R-CNN (Ren et al., 2015) with a ResNet-101 backbone (He et al., 2016). We used the implementation provided in the Detectron2 (Wu et al., 2019) framework. The experiments were conducted on a Nvidia V100 GPU. We trained the model for 30,000 iterations with a batch size of 2, a learning rate of 0.001 and 1000 warmup iterations. We decreased the learning rate by a factor of 10 after 15,000 and 20,000 iterations. We evaluated the model using the COCO mean Average Precision metric (*mAP*) [2] with an *Intersection over Union (IoU)* of 0.5 (*mAP@50*). The mAP measures the ability of the model to both localize and classify the objects in the image. We tested the model on the Test set obtaining a mAP of 73.41%, which indicates that the module is able to correctly localize and recognize the objects present in the images. Table 3 reports the individual APs for each object class. The results suggest that the network recognizes well large objects such as the *oscilloscope* (90.12%) or the *welder station* (89.87%), whereas it has some difficulty in recognizing small objects such as the *battery charger connector*, obtaining an AP of 15.91%. Figure 7 shows some qualitative results of our object detector.

[2] https://github.com/cocodataset/cocoapi

Table 3: Average Precision *AP* per class.

Category	AP	Category	AP
power supply	80.18	working area	90.18
oscilloscope	90.12	welder base	88.82
welder station	89.87	socket	90.27
electric screwdriver	81.45	left red button	100
screwdriver	58.73	left green button	100
pliers	79.18	right red button	81.82
welder probe tip	50.63	right green button	90.91
oscilloscope probe tip	51.72	power supply cables	41.34
low voltage board	88.53	ground clip	44.84
high voltage board	61.44	battery charger connector	15.91
register	71.07	panel A	89.77
electric screwdriver battery	51.72		

6 CONCLUSION

We presented HERO, a Conversational Intelligent Assistant able to support workers using the natural language and observing the surrounding environment to avoid natural language ambiguity. Experiments highlight the good performance on the considered industrial laboratory used to evaluate HERO's ability of *intent-entity* prediction considering both text and visual inputs. Future works will consider the integration of wearable devices such as Microsoft Hololens2 and a speech-to-text module to leave the human hands-free during their work.

ACKNOWLEDGEMENTS

This research is supported by Next Vision[3] s.r.l. and by the project MEGABIT - PIAno di inCEntivi per la RIcerca di Ateneo 2020/2022 (PIACERI) – linea di intervento 2, DMI - University of Catania.

REFERENCES

Amazon (2014). Amazon's alexa. https://developer.amazon.com/alexa.

Amazon (2019). Amazon's show and tell. https://www.amazon.com/b?ie=UTF8&node=21213731011.

Apple (2012). Apple's siri. https://www.apple.com/siri/.

Bochkovskiy, A., Wang, C., and Liao, H. M. (2020). Yolov4: Optimal speed and accuracy of object detection. *CoRR*, abs/2004.10934.

Bocklisch, T., Faulkner, J., Pawlowski, N., and Nichol, A. (2017). Rasa: Open source language understanding and dialogue management. *CoRR*, abs/1712.05181.

Bohus, D. and Rudnicky, A. I. (2009). The ravenclaw dialog management framework: Architecture and systems. *Comput. Speech Lang.*, 23.

Bordes, A., Boureau, Y.-L., and Weston, J. (2017). Learning end-to-end goal-oriented dialog. In *5th International Conference on Learning Representations (ICLR)*.

Brick, E., Alonso, V., O'Brien, C., Tong, S., Tavernier, E., Parekh, A., Addlesee, J.-A., and Lemon, O. (2021). Am i allergic to this? assisting sight impaired people in the kitchen. *ICMI '21: Proceedings of the 2021 International Conference on Multimodal Interaction*.

Bunk, T., Varshneya, D., Vlasov, V., and Nichol, A. (2020). DIET: lightweight language understanding for dialogue systems. *CoRR*, abs/2004.09936.

Fan, R.-E., Chang, K.-W., Hsieh, C.-J., Wang, X.-R., and Lin, C.-J. (2008). Liblinear: A library for large linear classification. *J. Mach. Learn. Res.*, 9.

Fedorenko, D. G., Smetanin, N., and Rodichev, A. (2017). Avoiding echo-responses in a retrieval-based conversation system. *ArXiv*, abs/1712.05626.

Gao, J., Galley, M., and Li, L. (2018). Neural approaches to conversational AI. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics: Tutorial Abstracts*.

Girshick, R. (2015). Fast r-cnn. In *Proceedings of the IEEE international conference on computer vision*.

Girshick, R., Donahue, J., Darrell, T., and Malik, J. (2014). Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*.

He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep residual learning for image recognition. In *Conference on Computer Vision and Pattern Recognition*.

Joachims, T. (1998). Text categorization with support vector machines: learning with many relevant features. In Nédellec, C. and Rouveirol, C., editors, *Proceedings of ECML-98, 10th European Conference on Machine Learning*, number 1398.

Joulin, A., Grave, E., Bojanowski, P., and Mikolov, T. (2017). Bag of tricks for efficient text classification. In *EACL*.

Kennington, C. R., Kousidis, S., and Schlangen, D. (2014). Inprotks: A toolkit for incremental situated processing. In *SIGDIAL Conference*.

Leonardi, R., Ragusa, F., Furnari, A., and Farinella, G. M. (2021). Egocentric human-object interaction detection exploiting synthetic data. In *International Conference on Image Analysis and Processing*.

Lison, P. and Kennington, C. (2016). OpenDial: A toolkit for developing spoken dialogue systems with probabilistic rules. In *Proceedings of ACL-2016 System Demonstrations*.

Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.-Y., and Berg, A. C. (2016). Ssd: Single shot multibox detector. In *Computer Vision – ECCV 2016*.

Microsoft (2014). Microsoft's cortana. https://support.microsoft.com/en-us/topic/what-is-cortana-953e648d-5668-e017-1341-7f26f7d0f825.

Mikolov, T., Chen, K., Corrado, G. S., and Dean, J. (2013). Efficient estimation of word representations in vector space.

[3]Next Vision: https://www.nextvisionlab.it/

Mleczko, K. (2021). Chatbot as a tool for knowledge sharing in the maintenance and repair processes. *Multidisciplinary Aspects of Production Engineering*, 4.

Pennington, J., Socher, R., and Manning, C. D. (2014). Glove: Global vectors for word representation. In *EMNLP*.

Redmon, J. and Farhadi, A. (2018). Yolov3: An incremental improvement. *CoRR*, abs/1804.02767.

Ren, S., He, K., Girshick, R., and Sun, J. (2015). Faster R-CNN: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*.

Skantze, G. and Moubayed, S. A. (2012). Iristk: a statechart-based toolkit for multi-party face-to-face interaction. In *ICMI '12*.

Ultes, S., Rojas-Barahona, L. M., Su, P.-H., Vandyke, D., Kim, D., Casanueva, I., Budzianowski, P., Mrkšić, N., Wen, T.-H., Gašić, M., and Young, S. (2017). PyDial: A multi-domain statistical dialogue system toolkit. In *Proceedings of ACL 2017, System Demonstrations*.

Wen, T.-H., Gašić, M., Mrkšić, N., Su, P.-H., Vandyke, D., and Young, S. (2015). Semantically conditioned LSTM-based natural language generation for spoken dialogue systems. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*.

Williams, J., Asadi, K., and Zweig, G. (2017). Hybrid code networks: practical and efficient end-to-end dialog control with supervised and reinforcement learning. In *ACL*.

Williams, J. D., Arizmendi, I., and Conkie, A. (2010). Demonstration of at amp;t "let's go": A production-grade statistical spoken dialog system. In *2010 IEEE Spoken Language Technology Workshop*.

Williams, J. D. and Liden, L. (2017). Demonstration of interactive teaching for end-to-end dialog control with hybrid code networks. In *Proceedings of the 18th Annual SIGdial Meeting on Discourse and Dialogue*.

Wu, Y., Kirillov, A., Massa, F., Lo, W.-Y., and Girshick, R. (2019). Detectron2. https://github.com/facebookresearch/detectron2.

Young, S., Gašić, M., Thomson, B., and Williams, J. D. (2013). Pomdp-based statistical spoken dialog systems: A review. *Proceedings of the IEEE*, 101(5).

Zhang, X., Zhao, J., and LeCun, Y. (2015). Character-level convolutional networks for text classification. In *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 1*, NIPS'15.