IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received April 16, 2021, accepted May 3, 2021, date of publication May 25, 2021, date of current version June 14, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3083518

Multi-Modal Chatbot in Intelligent Manufacturing

TZU-YU CHEN[1], YU-CHING CHIU[1], NANYI BI[2], AND RICHARD TZONG-HAN TSAI[1,2,3,4]

[1]Department of Computer Science and Information Engineering, National Central University, Taoyuan City 320317, Taiwan
[2]IoX Center, National Taiwan University, Taipei 10617, Taiwan
[3]Center for Geographic Information Science, Academia Sinica, Taipei 115201, Taiwan
[4]Research Center for Humanities and Social Sciences, Academia Sinica, Taipei 11529, Taiwan

Corresponding author: Richard Tzong-Han Tsai (thtsai@g.ncu.edu.tw)

This work was supported in part by the Ministry of Science and Technology of Taiwan under Grant MOST 108-2633-E-002-001, and in part by the National Taiwan University under Grant NTU-108L104039.

ABSTRACT Artificial intelligence (AI) has been widely used in various industries. In this work, we concentrate on what AI is capable of doing in manufacturing, in the form of a chatbot. We designed a chatbot that helps users complete an assembly task that simulates those in manufacturing settings. In order to recreate this setting, we have users assemble a Meccanoid robot through multiple stages, with the help of an interactive dialogue system. Based on classifying users' intent, the chatbot is able to provide answers or instructions to the user when the user encounters problems during the assembly process. Our goal is to improve our system so that it can capture users' needs by detecting their intent and therefore provide relevant and helpful information to the user. However, in a multiple-step task, we cannot rely on intent classification with user question utterance as the only input, as user questions raised from different steps may share the same intent but require different responses. In this paper, we proposed two methods to address this problem. One is that we capture not only textual features but also visual features through the YOLO-based Masker with CNN (YMC) model. Another is the usage of an Autoencoder to encode multi-modal features for user intent classification. By incorporating visual information, we have significantly improved the chatbot's performance from the experiments conducted on different dataset.

INDEX TERMS Chatbot, human-robot interaction, multi-modal intent classification.

I. INTRODUCTION

A. ARTIFICIAL INTELLIGENCE IN MANUFACTURING

Artificial intelligence (AI) has risen rapidly and has been widely used in the society, and is causing a huge impact on how people work, behave, and live [1], [2]. By providing a fast, efficient, and customized service, AI truly enhances our daily lives with its lowered costs, higher efficiency, fewer human errors and greater output, making the technology more accessible among various industries. In this work, we focus on AI in intelligent manufacturing.

AI has already transformed manufacturing in many ways. For instance, we have robotic arms for robotized assembly [3], which is useful when conducting highly repetitive and high risk tasks to reduce time cost and protect humans from danger. Another case is where computer vision takes place for defect detection, especially when the flaws in products are too small for inspectors to notice with the naked eye [4], [5]. Cases above reveal the advantages that AI is capable of offering, including rapid manufacturing conduction, operational costs minimization, danger prevention, etc. Without question, AI is making its way into manufacturing.

The associate editor coordinating the review of this manuscript and approving it for publication was Her-Terng Yau.

B. CHATBOT IN MANUFACTURING

Although AI has achieved enormous success in manufacturing, chatbot is not much involved in this field. The possible reason is that when we speak of a chatbot, we usually refer to a text-based chatbot, which faces challenges when it comes to multiple-step tasks common in manufacturing, as it cannot fully capture the status of the user with mere textual information, potentially leading to irrelevant response. Inspired by [6] which responds to users' queries (images and speech questions) in order to assist people in different situations, we introduce a multi-modal chatbot which combines visual and textual information to better capture user's status. In this section, we will discuss what roles a multi-modal chatbot can play under the manufacturing environment. We came up with three settings: early stage for orientation training, mid stage for working on an assembly line, and late stage for repair and

maintenance, all of which represent common scenarios that a worker has to go through in manufacturing settings. The details of each stage are as follows.

1) EARLY STAGE: ORIENTATION TRAINING

Chatbot can be used in the field of education [7] where the chatbot is able to deliver knowledge to students. Similarly, chatbot can also deliver techniques to novices. In this case, chatbot can be utilized in orientation training. Imagine you are a supervisor in a factory and you are facing a bunch of novices who are going to work on an assembly line. Your job is to deliver a lecture to these novices to familiarize them with what they are going to do. In this case, chatbot can come to save the day. A chatbot with domain knowledge can serve as an expert and therefore provide instructions to the newbies.

2) MID STAGE: ASSEMBLY LINE

In [8], a chatbot combined with visual management for the purpose of monitoring and recording issues of a production line can be used in industry. In our case, a visual-aid QA based chatbot can be deployed in the assembly process. In this stage, although workers are usually able to complete the assembly process by themselves, it is still possible that they get confused when it comes to complex steps in the process, especially when the workers are new to the workplace. During the assembly process, chatbot can turn itself into a QA or dialogue system. The worker can simply ask the chatbot for help when encountering difficulties. Chatbot in this stage is required to recognize which step of the assembly process the worker is in and give a proper response.

3) LATER STAGE: REPAIR AND MAINTENANCE

Similar to the previous stage, chatbot in this stage aims to provide assistance by giving detailed information on how to repair or maintain the product whenever the worker gets stuck. The main difference between this stage and the previous stage is the order of the steps. Unlike the fixed order of steps in the previous stage, this stage usually does not have a prescribed set of steps to follow, the steps are dependent on what component is being repaired. Therefore, it is vital for the chatbot to identify the current step and thus provide appropriate responses.

C. MECCANOID ROBOT FINAL ASSEMBLY TASK

To model the manufacturing environment, we designed an assembly task using a Meccanoid robot to represent a self-assembly task. Shown in Figure 1, Meccanoid is a personal robot which users can build in the way they want and write programs to make it move.

The robot contains 497 pieces and it takes 90 steps to complete. Since we are not measuring the performance of the user in the task, as is not the focus of our study, we reduced this task to main component assembly task, or final assembly task, in which the users needed to join the main body parts, including head, neck, body, left and right arms, both legs, and the feet, together, instead of assembling the finer parts.

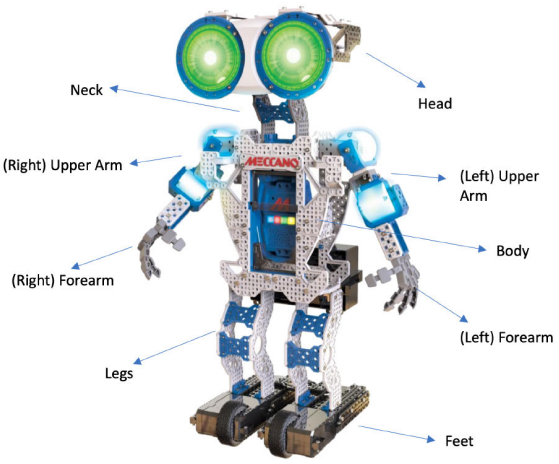


FIGURE 1. Meccanoid robot.

We then designed a task-oriented dialogue system, or in other words, a chatbot for this task. Users can complete the robot assembly under the guidance of our system. If the user has any questions, he or she can ask our system for help. Since the system focuses only on solving users' problems during the assembly process, we simplified our dialogue system as a QA system. That is, our system only provides the solution to the questions initiated by the user. Here we take advantage of the versatility of Frequently Asked Question (FAQ). Basically, FAQ can solve most of the users' question. We first collected a FAQ list as the class of the user question by pilot study and designed the response for each question in the list. Our system can then provide the solution by classifying the input question to the correct FAQ. However, since our task is multi-step by nature, we noticed that the system failed to predict user intent when the assembly step should be taken into consideration. Specifically, the chatbot may give a relevant answer for step two when the user is still in step one. The fact that the intent classification based solely on users' utterance inputs motivates us to come up with the solution to incorporate visual data into our user intent classification model.

In this work, we proposed YOLO-based Masker with CNN (YMC) model to extract information from video clips which can help discriminate between different user intents. We adopted the idea of object detection to highlight the components of robots shown in the video frames and mask out the background. After integrating YMC model into our user intent classifier, we can see a significant improvement on accuracy of user intent classification. For a more detailed illustration, please see our demo here.[1]

II. RELATED WORK

A. DIALOGUE SYSTEM

A dialogue system [9] is a computer system which can converse with a human either in text, speech or other kinds of modalities. It can be categorized into two groups according

[1] Demonstration Video - `https://youtu.be/jpjaQ9UnGD0`

to its purpose: task-oriented dialogue system and chit-chat dialogue system. A task-oriented dialogue system is built to assist people with some certain tasks, while a chit-chat dialogue system is built to interact with people for no specific goals.

Generally speaking, a dialogue system includes the following three parts: Natural Language Understanding (NLU), Dialogue Management (DM) and Natural Language Generation (NLG). First, the user utterance is fed into the NLU module. NLU aims to understand what the purpose of the user utterance is. User intent detection and slot-filling are two essential tasks of NLU. We then need a DM module to manage the dialogue state and action. Dialogue state tracker tracks the dialogue state by the current and the past turns, while dialogue policy learning generates the next dialogue action according to the current dialogue state. Finally, NLG module gives the response after receiving the selected dialogue action. The responses from NLG module can either be fixed according to hand-crafted rules or be generated from a well-trained NLG models [9].

We aim to build an interactive dialogue system to solve user problems in the Meccanoid robot final assembly task, so our system would be a task-oriented dialogue system. In our system, we do not need a DM module since the interaction of question answering would only be done in one turn. Also, we have fixed possible questions, so we only need hand-crafted answers for system responses. Therefore, we focus on improving the user intent classifier in order to give accurate answers to the user questions.

B. VISUAL AND VIDEO QUESTION ANSWERING

Visual Question Answering (VQA) [10] is an interesting research topic which involves natural language processing and computer vision. Given an image and a natural language question, the task is to choose the correct answer from a list of answer candidates. How to generate a joint representation of the two different modalities is the main issue in this task. Some attention-based models have achieved great performance by applying some powerful deep learning methods. MUTAN [11], for instance, provides a fusion method to extract information between the question and the image. MLAN [12] adds multi-level attention in order to capture the region of the image that is related to the question. DFAF [13] focuses not only on the information of inter-modality attention but also on the information of intra-modality attention.

Instead of answering questions according to an image, video question answering task [14], [15] [16] is to answer the questions according to a video clip. Unlike a single image, a video clip contains a series of frames and audio. To correctly answer the question, the model has to first select the part of the video which is related to the question, extract the information from it and then generate a joint representation for answer prediction.

C. OBJECT DETECTION

Object detection is the task related to computer vision and image processing. Models included in R-CNN series such as R-CNN [17], Fast R-CNN [18], Faster R-CNN [19] do the detection in two stages. The candidate regions are first selected, and a CNN detector then classifies the objects in these regions. Unlike R-CNN based models, models like SSD [20] and YOLO [21], [22] [23], [24] handle localization and classification simultaneously. It significantly reduces the detection process time, and can therefore practice the real-time object detection.

III. SYSTEM DESIGN

A. DATASET

1) PRE-DATA COLLECTION: WIZARD OF OZ PILOT STUDY

To collect possible questions user may ask during the assembly, we first conducted a pilot study. Fifteen participants without any Meccanoid robot assembly experience were asked to complete the final assembly task. They could ask question during their assembly process when they encountered any problem. Additionally, we trained a research assistant by having him practice this assembly task for many times and made sure that this student was familiar with the whole process. This assistant, as an expert, later served as our conversational agent, which gave instructions and answered participants' questions. The participant and the expert would be sitting in the same room. They were separated by a wall to make sure they cannot see or hear each other. The communication between them was done through a laptop. Figure 2 illustrates how this pilot study is done.

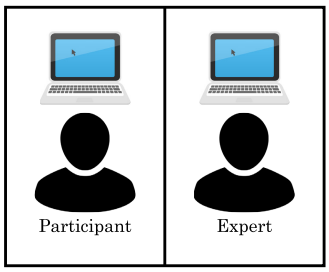


FIGURE 2. Illustration of our wizard of Oz experiment setting.

Throughout the whole task, the participant had no idea he or she was interacting with a real person rather than a pre-programmed QA system. We used this Wizard of Oz experiment design because, on one hand, it was flexible and intelligent enough so as to insure the ongoing of the interaction with the participant [25]. On the other hand, with this kind of experiment setting and participants believing they were interacting with a QA system, they tended to ask questions more formally to ensure that the question they ask can be clearly understood by the conversational agent. For example, when user wants to know how the neck should be attached, the participant would ask the question "How should I connect the neck and the body?" rather than "How to connect these two?" to clarify the problem he encounters.

TABLE 1. FormalQ list of 21 classes.

Class	User Intent	Assemble Step	FormalQ
0	screw direction	Leg to Body	Is there a right direction to lock the screws?
1	connection	Leg to Body	Which holes should I lock the screws?
2	nut position	Leg to Body	Should I put the nuts on top or bottom?
3	screw direction	Feet to Leg	Does the direction I lock the screws matter?
4	connection	Feet to Leg	Do you have anymore detail?
5	screw direction	Neck to Body	Is the screw direction from outside to inside?
6	connection	Neck to Body	Can I get a closer picture?
7	neck direction	Neck to Body	Does the neck have front side or back side?
8	connection	Head to Neck	Where do I put the screws? Can you zoom?
9	S2	Forearm to Upper arm	Doesn't matter which one I put S2?
10	connection	Forearm to Upper arm	Where does the S2 go and where does the M2 go?
11	embedding	Forearm to Upper arm	Should I embed it?
12	S2	Arm to Body	Again only one screw?
13	connection	Arm to Body	Show me more details about the joint part?
14	check	Arm to Body	I want to check to make sure I didn't make any mistake.
15	nut direction	*-	Does hex nut have front side or back side?
16	tightness	-	Should I lock the screws tightly?
17	assembly skills	-	Is there a better way for locking screws?
18	holes	-	Is it necessary to lock all the holes?
19	operation way	-	Can I lay it down to install?
20	wires	-	Should I care about the wires?

*The step does not matter when classifying the user intent.

TABLE 2. Example of questions collected from AMT.

FormalQ	Collected Question
Is there a right direction to lock the screws?	Which direction am I supposed to turn the screws in?
Should I lock the screws tightly?	Does it matter if I lock it tightly?
Should I put the nuts on top or bottom?	Where do the nuts go?
Is the screw direction from outside to inside?	Should the screw come in from the outside?

As our focus is to build a FAQ list, we need to categorize these collected questions. After looking into them, we found that there were only 13 kinds of problem user may encounter when completing Meccanoid final assembly task. Some of these problems are step-independent, which means that system response to these problems would be different according to the current step, while the others are not. These step-independent question categories were subdivided into several question categories according to the step. By this, we expanded the question categories from 13 classes into 21 classes. Finally, we chose the most representative question for each problem as formal question (FormalQ), which will also be the user intent. FormalQ list of 21 user intents and assembly scenarios are shown in Table 1.

2) DATA COLLECTION

After we defined the classes and formal questions of each class, the next step was to enlarge the dataset. We decided to collect data from Amazon Mechanical Turk (MTurk), an online crowd-sourcing platform. On this platform, requesters can post works as Human Intelligence Tasks (HITs), and workers can get rewards by completing these HITs. It also provides flexible API and UI for requesters to create and manage their HITs.

We first built up a website as our data collecting system with the provided services of Amazon Mechanical Turk. As we knew that there were only 21 possible classes of questions, what we needed to do was to generate more questions in every single category so that we could get enough data for the following training process. We provided several video clips of each assembly step where the questions may be asked and the FormalQ of each class. After watching the video clips, workers had to generate question which is similar to FormalQ in a different way. To avoid the generated question being too similar to FormalQ, we set an evaluation process before the workers submitted their questions. The evaluation process was to calculate the cosine similarity of the generated question and the target FormalQ. The cosine similarity of the two questions should pass the threshold we set to be accepted by our system.

In the end, we recruited 180 workers at total and 3780 questions were collected. We later checked the questions manually and deleted question which was either meaningless or too short, e.g., "I cannot tell". Questions which were misclassified were also moved to the correct classes. After the manual evaluation, we eventually got 3690 questions as the dataset. Examples of collected questions can be seen in Table 2.

TABLE 3. Dataset statistics.

Dataset	# of data	Dataset	# of data
TD_{train}	2,901	MM_TD_{train}	11,604
TD_{test}	789	MM_TD_{test}	3,156
SVD_{train}	2,901	MM_SVD_{train}	11,604
SVD_{test}	789	MM_SVD_{test}	3,156
$TDSVD_{train}$	5,802	MM_TDSVD_{train}	23,208
RVD_{test}	789	MM_RVD_{test}	3,156

3) POST-DATA COLLECTION: VOICE DATA AND VISUAL DATA COLLECTION

We aim to build a conversational agent which user can interact with the system orally, so we also need voice dataset. After questions in text form were collected, we then converted them into synthesized voice data by Google Text-To-Speech service. The language and accent is set as English and American accent. We also recruited 8 people with different nationality and gender to collect their voice data as real data in order to test how robust the system is. They were asked to read the selected questions in their own accent and pace.

We also recorded the assembly process of user as the visual data. The camera was placed over the working table so that the whole assembly process can be clearly captured. We filmed 3 to 4 videos to present the start, the middle and the end of each step, and each video has the length of 12 to 23 seconds. According to the research [26], mean rate of speech in conversation is 208.7 words per minute. In our dataset, the longest question is consisted of 30 words, which can be finished in 10 seconds with the speech rate. Therefore, we manually cropped these videos into several 10-second clips. With this process, we got 4 informative clips for each step.

B. MULTI-MODAL DATASET

We separated the collected textual question dataset (TD) into training set and testing set with 2,901 questions and 789 questions. We also converted them into synthesized voice data (SVD) in the same split of training and testing set. As for recorded voice data (RVD), we recorded 789 questions as testing set. Also, we combined TD and SVD to form a larger dataset (TDSVD), which will only be used as training set. Every video clips were then paired with each dataset according to the corresponding step to form the final multi-modal dataset, which makes the training and testing dataset four times larger than those without video data. The name of these multi-modal dataset would be added a prefix "MM_". Table3 provides the statistic of these datasets.

C. SYSTEM ARCHITECTURE

1) YMC MODEL

Since the steps of our assembly task highly influence the result of user intent detection, what we want to know from the visual data is the current assembly step.We are able to provide proper response by being aware of user's status. Intuitively, the easiest way to detect the step from the image is to see which components show on the working table. For example, if there are only the body part and the neck part in the working space, we can indicate that the user is connecting the neck to the body of the robot. In that case, we adopted the idea of object recognition and object detection.

First, we have to collect training data. We took several photos of each Meccanoid robot component from different angles. Figure 4 shows some examples of photos of the body part. We collected 447 images in total for training the YOLOv4 object detection model.

After the object detection model was trained, we then integrated it into the proposed YMC model. YMC model contains two parts: YOLO-based Masker and CNN model. After a 10-second video is fed into the YMC model, the YOLO-based Masker randomly chooses 1 frame from the video. The image will then go through the pre-trained YOLOv4 object detection model to detect the presence of the robot components.

Finally, the YOLO-based Masker generates masked images according to the object detection results. The YOLOv4 object detection model plots the bounding boxes of the detected components, and we then mask out the area which is not covered in the bounding boxes. This masked image is then sent into the CNN model. A 3-layer CNN model extracts visual features from the single image and finally outputs a vector as the overall visual representation.

2) MULTI-MODAL INTENT CLASSIFICATION MODEL

Figure 3 presents the overall architecture of our multi-modal user intent classification model. The user question utterance is first converted to text form by Google ASR system. The question is then transformed into a representation vector through a pre-trained language representation model, Bidirectional Encoder Representation from Transformers (BERT) [27]. We take the output vector of [CLS] token from BERT as the question embedding, i.e., the textual feature. Meanwhile, the video clip is fed into the proposed YMC model to obtain the video feature. The two features extracted from user utterance and video are concatenated to form an overall feature. Finally, we pass the overall feature into the classifier to predict user intent.

IV. EXPERIMENTS AND RESULTS

To see how the visual data can help, We separated the experiments into two parts: intent classification without video data and multi-modal intent classification. The analysis of different test datasets would also be presented at the end of this section.

A. WITHOUT-VIDEO INTENT CLASSIFICATION

We used datasets without video features to train the baseline intent classifier. We tried different experimental settings in order to find out the best setting with the highest accuracy, which will then be used to train multi-modal intent classifier.

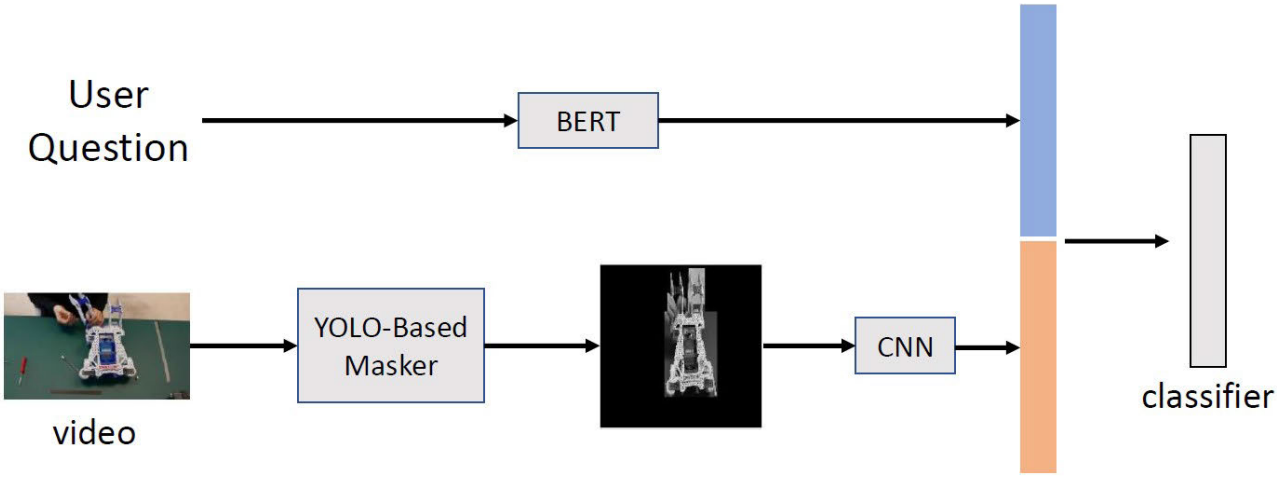


FIGURE 3. Model that integrates textual context and visual context to form new representation which is later fed into the intent classifier to predict user intent.

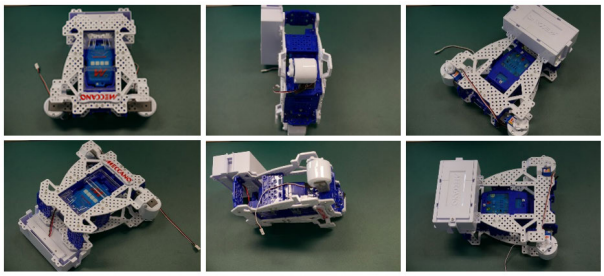


FIGURE 4. Photos of the body part.

1) TRAINING DATASET

We first trained the model on different training datasets. From Table 4 we can see that the highest accuracy can be achieved by the model trained on TDSVD_{train} dataset. The possible reason is that TDSVD_{train} dataset contains the most data. Also, since perfectly written text data and text data converted from voice data are both included in TDSVD_{train} dataset, the trained model is robust enough to handle data with or without noises.

TABLE 4. Accuracy of models trained on different training dataset.

Training Dataset	Testing Dataset			
	TD_{test}	SVD_{test}	RVD_{test}	Average
TD_{train}	76.81	67.43	57.41	67.22
SVD_{train}	75.16	70.34	59.82	68.44
TDSVD_{train}	76.17	70.22	60.46	**68.95**

2) MAX SEQUENCE LENGTH OF INPUT OF BERT MODEL

Next, we trained our model with different sequence length of BERT input. Before a sentence is sent into the BERT model, it has to be tokenized, and two tokens, [CLS] and [SEP], are then appended to it. That is, if we set the max sequence length as L, only $L-2$ tokens of the sentence would be input. We have tried to set the max sequence length to 20 in order to reduce the computing resources and training time. Since the longest question in our datasets contains 30 words, here we set the max length to 32 to see if it really influences the results. Table 5 shows that we can get the highest accuracy by setting the max length to 32.

TABLE 5. Accuracy of models with different max sequence length.

Max Sequence Length	Testing Dataset			
	TD_{test}	SVD_{test}	RVD_{test}	Average
20	75.92	69.46	59.95	68.44
32	76.17	70.22	60.46	**68.95**

3) SENTENCE REPRESENTATION METHOD

Finally, we tried two different methods to obtain question embedding. The output vector of [CLS] token, which is a special token in BERT model, can be considered as an overall embedding of the input sentence. Since our task is a classification problem, it is reasonable to take the output of [CLS] token as the sentence representation. We also tried to add an RNN model at the end of BERT model to obtain the embedding of the input sentence. The results are shown in Table 6. We can find a slight improvement in the model with CLS method.

From the experiment results above we can conclude that: (1)model trained with dataset TDSVD_{train}, (2)max sequence length set to 32, and (3)[CLS] token set as sentence representation, are the best text model settings.

B. MULITMODAL INTENT CLASSIFICATION

We integrated textual features from BERT and visual features from our YOLO-based Masker with CNN to form a new representation which is then be fed into the intent classifier. In this section, we implemented two ways of fusing the different modalities. One is simply concatenate the textual

TABLE 6. Accuracy of models with different sentence embedding methods.

Method	Testing Dataset			
	TD_{test}	SVD_{test}	RVD_{test}	Average
CLS	76.17	70.22	60.46	**68.95**
RNN	75.79	70.72	60.08	68.86

and visual features. The other is feeding the concatenated features into an Autoencoder and obtain the encoded features as the input of our intent classifier.

1) CONCATENATION

We chose TDSVD_32_CLS as the text model settings to be paired with our YMC model in order to form the final multi-modal intent classifier. For visual data, we also tried different methods to extract features.

The baseline method is that we capture the video features by a simple CNN model without any other pre-processing.

We also conducted experiments on applying YOLOv3 and YOLOv4 to see whether we can get any improvement with different object detection model. These model names will have the suffix "_v3" and "_v4" to show which object detection model is used.

Table 7 shows the results of these models. The baseline model gets an average of 81.48% on accuracy.

TABLE 7. Accuracy of different visual feature extraction model.

Model	Testing Dataset			
	MM_TD_{test}	MM_SVD_{test}	MM_RVD_{test}	Average
Baseline	87.61	82.83	73.99	81.48
YMC_v3	90.27	85.71	76.93	84.30
YMC_v4	90.68	86.03	76.90	**84.54**

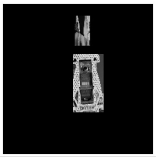
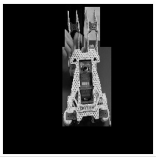
Still, without any pre-processing, the raw images may contains noises and thus leads to a huge impact on the following user intent classifier. By masking, YMC model only focus on the important parts of the image, and thus outperforms the others with the accuracy of 84.54%.

In Table 7, we can also see that the model which applies YOLOv4 object detection has a better result.

As for YMC model, it would mask out the area where no component is detected. From here we can realize how important the object detection model is. We take some of the masked images processed by Yolo-base Masker to see the difference between the results of YOLOv3 and YOLOv4.

Table 8 shows the comparison between the masked images in different steps. In step 1: Legs and the body, we can see that YOLOv4 model correctly detects the two legs in the image, while YOLOv3 model misses one of the leg. Also, the detected area of YOLOv4 covers completely over the components in the image. YOLOv3 covers only a part of each detected components. For step 5: Forearm to upper arm, we can clearly see the difference. In the original image, the user covers most of the arm component with his hands.

TABLE 8. Masked images generated by YOLOv3 and YOLOv4.

Step	Original Image	Masked (YOLOV3)	Masked (YOLOv4)
1			
5			

YOLOv3 fails to detect and thus masks out the whole image. In contrast, YOLOv4 correctly points out the area of the arm.

2) AUTOENCODER

In addition to concatenating the modalities, we also did another experiment to see if the different modalities influence each other. Since our chatbot aims to be used in a manufacturing factory, we expect our model to be quickly trained and used in such a setting. Therefore, in this section, we only took a single frame from the video as the visual feature and compared the result with the YMC model which also obtained the visual feature from randomly picked frame from the video. In this section, we add Autoencoder [28]into our model architecture. Since Autoencoder is an unsupervised-learning method, we simply took the concatenated textual and visual features as the training data. We encoded the concatenated feature into different dimensions D, which D ranged between 100 and 2300 dimension.

In Figure 5 and Table 9, among all dimension D, we find that when $D = 1300$, the accuracy of autoencoder is slightly higher than YMC model. When $D > 1300$, the accuracy shows a decreasing trend.

We can also look into how much time is spent during inference. Figure 10 shows the time differences between different models. when $D = 100$, the inference time is slightly smaller than the YMC model. Although the average accuracy of the Autoencoder_100D model is about 3% lower than the YMC model, it indeed executes faster. Therefore, a feature that is encoded by an Autoencoder with a low dimension can be considered for use when quick inference is needed, especially under the manufacturing setting.

The main difference between the Autoencoder model and the YMC model is that an Autoencoder needs to be trained in advance when implementing the Autoencoder model. On the other hand, the YMC model does not require such prior preparation since it is an end-to-end model.

Since in the manufacturing factory setting, a high performance and high efficiency method is required. Therefore, we believe that the Autoencoder method which executes faster by encoding features into smaller dimensions and

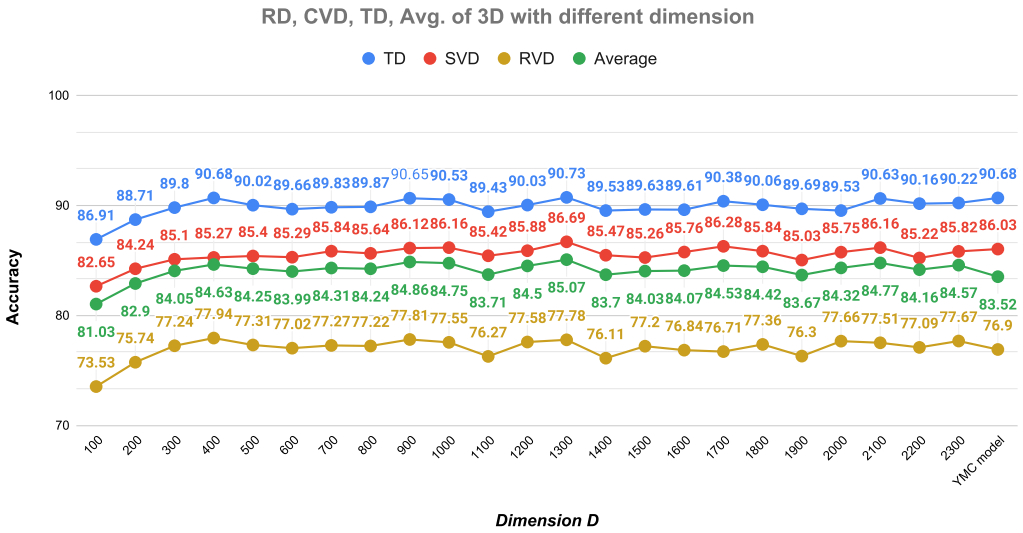


FIGURE 5. Autoencoder with different dimensions.

TABLE 9. Accuracy of different feature fusion model.

Model	Testing Dataset			
	MM_TD_{test}	MM_SVD_{test}	MM_RVD_{test}	Average
YMC_v3	90.27	85.71	76.93	84.30
YMC_v4	90.68	86.03	76.90	84.54
Autoencoder_100D	86.91	82.65	73.53	81.03
Autoencoder_1300D	**90.73**	**86.69**	**77.78**	**85.07**

TABLE 10. Inference time between different models.

Model	Inference time / per data
Autoencoder_100D	1.92ms
Autoemcoder_1300D	1.99ms
YMC_v4	2.13ms

TABLE 11. Accuracy of different visual information choosing model.

Model	Testing Dataset			
	MM_TD_{test}	MM_SVD_{test}	MM_RVD_{test}	Average
Baseline	87.61	82.83	73.99	81.48
RandImg	89.96	85.27	76.74	83.99
YMC_v3	90.27	85.71	76.93	84.30
YMC_v4	90.68	86.03	76.90	84.54

remains high accuracy can be considered to be a suitable method for the scenario of a manufacturing factory.

C. ANALYSIS AND DISCUSSION

1) ERROR ANALYSIS

Confusion matrix is a common way to visualize the performance of a classification model. From a confusion matrix, we can see how many data are incorrectly classified and which class they are classified. Figure 6 and Figure 7 show the classification results of model using text only and model incorporating visual context, respectively. Class 15 to Class 20 are step-independent classes, and we can see that text-only model does a great job on these classes. But for those step-dependent classes, the classification accuracy drops dramatically. As for the model incorporating visual context, it does better than text-only model on step-dependent classes, but it does not maintain the high performance on step-independent classes. The possible reason is that for those step-independent classes, visual context can be a noise since they can be correctly classified by text only.

We then check how well the models do on step-dependent classes, which is the core problem we want to tackle in this work. We choose the classes with the user intent ''connection'' which appears in all the steps. The classification performance of two models are show in Figure 8 and Figure 9. We can see that text-only model correctly classifies the questions asked in forearm-to-upper-arm and arm-to-body step. For the rest of the steps, the performance of text-only model significantly drops. The proposed model gets better performance than that of the text-only model except for head-to-neck and arm-to-body step. In these two steps, user has to stand the robot up to assemble these two components. Since the camera is placed above the working table, some of the robot components may be entirely covered. This can lead to the problem of incorrect detection of our YOLO-based Masker, which would later influence the classifier.

2) ANALYSIS ON VISUAL CONTEXT CAPTURE METHOD

Here we discuss different methods of capturing visual information. The baseline method captures the video features by a simple CNN model. RandImg model randomly chooses a frame from the video as the visual data instead of using the whole video data. YMC model randomly chooses a frame

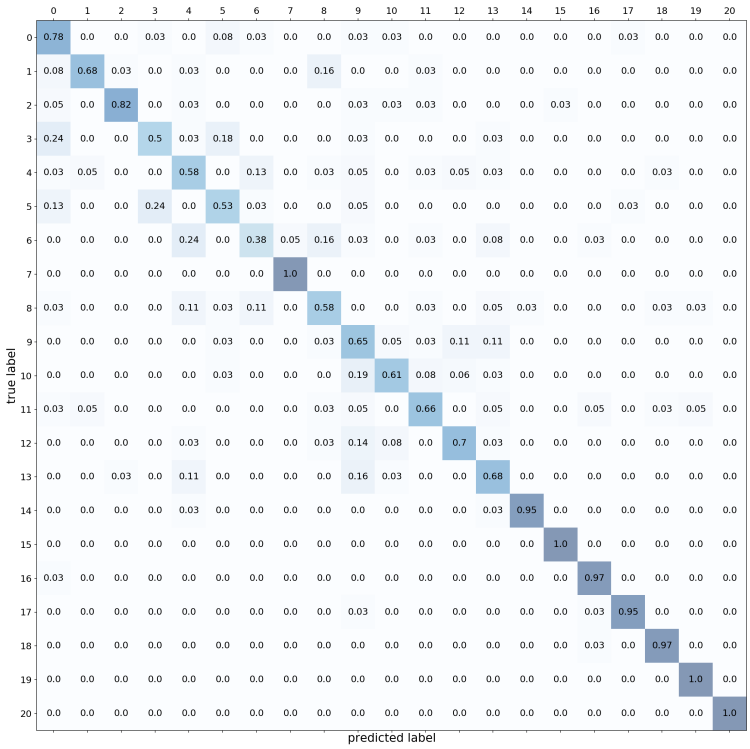


FIGURE 6. Confusion matrix of the text-only model.

from the video first and the frame would then go through the YOLO-based Masker. We also tried to apply YOLOv3 and YOLOv4 in this experiment.

Table 11 shows the experiment results. We can surprisingly find that by randomly choosing a frame, the model can achieve higher accuracy than the model utilizes the whole video data when no pre-processing stage is included. The possible cause is that without any pre-processing, more noise is carried in the video than that be carried in an image.

By applying our YOLO-based Masker with CNN model, we can top the rest of the models even with taking only one random frame as the visual data.

The results prove that the pre-processing of the image is important to our task.

3) ANALYSIS ON DIFFERENT TEST DATA

Since this work focuses on building a speech dialogue system, we want to know how the voice could influence the experiment results. In this section, we show the detailed experiment results of MM_SVD and MM_RVD$_{test}$.

First, we need to measure the quality of the collected voice dataset. We decided to evaluate the collected voice dataset according to Word Error Rate (WER) [29]. WER is a common metric used to evaluate the performance of a speech recognition system or machine translation system. The core idea of WER is simple: we want to know how many words in the hypothesized sentence we should delete, insert or substitute to exactly match the reference sentence. WER is calculated as below:

$$Word\ Error\ Rate\ (WER) = \frac{S+D+I}{S+D+C},$$

where S stands for the number of substitutions, D for deletion, I for insertion and C for correct words.

Since we applied the powerful ASR system of Google in our system, we're not using WER to evaluate the performance of ASR. Instead, we consider WER as how accurate the voice data is compared to the text data. That is, if one voice dataset has higher WER, we can say that there are more noises in it. Background noises and accent can be one of the causes of high WER. Table 12 shows the WER of the collected voice dataset and the accuracy of our model test on these datasets. We can see that the WER of MM_RVD$_{test}$ is much higher than that of MM_SVD$_{test}$. The accuracy, as can be expected, drops to 79.85% when testing on MM_RVD$_{test}$.

We then look into MM_RVD$_{test}$ to see if we can have the same finding. Table 13 shows simple personal information, the WER and the experiment results of every single subject in MM_RVD$_{test}$. We separate these subjects into three groups according to the WER and discuss them respectively.

The accuracy of data in the first group, which has the WER under 22.56%, both achieve over 85%. The data with the lowest WER even gets 96.25% of accuracy. The second group of data, with the WER between 22.56% and 28.55%, gets the accuracy around 74% to 84%. Interestingly, the data

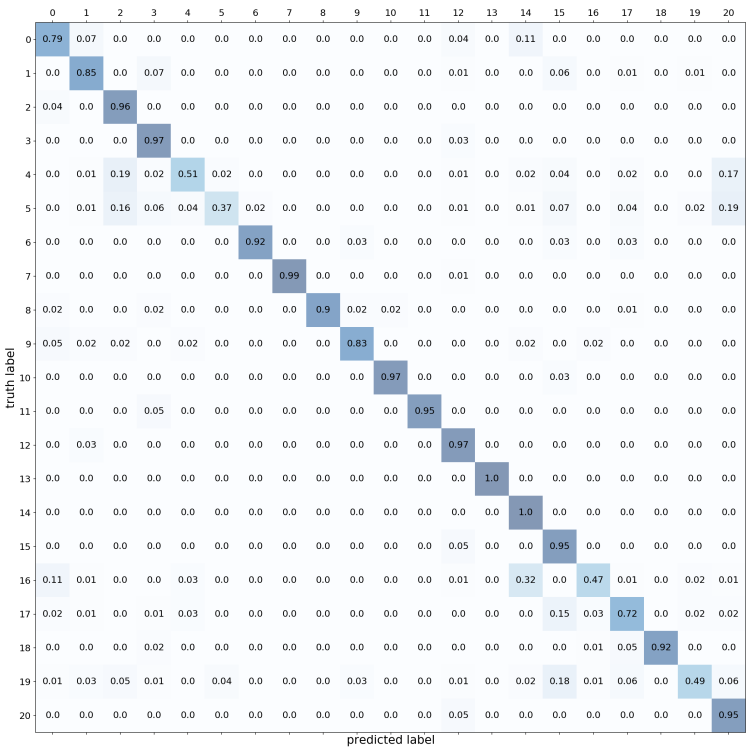


FIGURE 7. Confusion matrix of the YMC model.

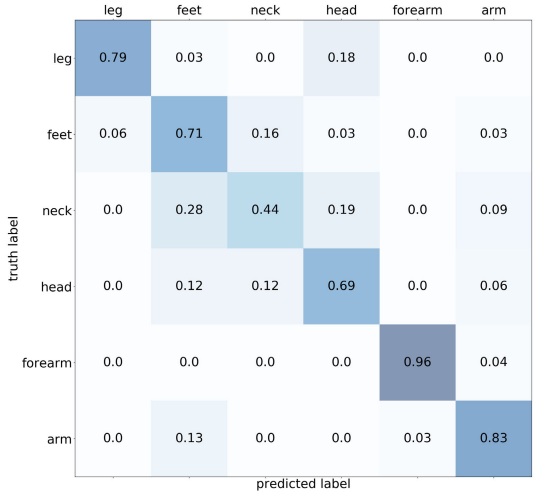


FIGURE 8. Confusion matrix on class with same user intent of the text-only model.

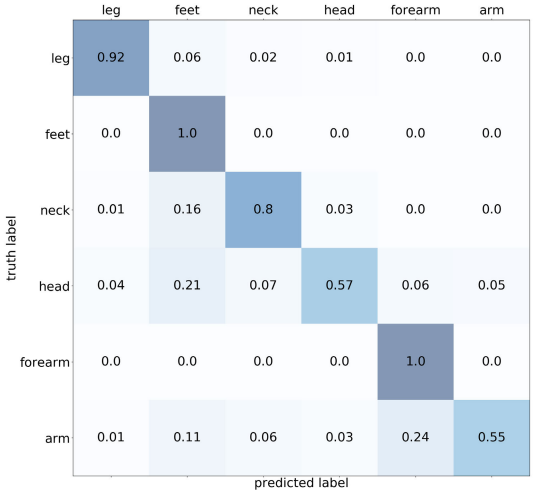


FIGURE 9. Confusion matrix on class with same user intent of the YMC model.

with WER of 28.27% gets the best performance in this group. The possible explanation is that the important words of this subject's utterance are correctly recognized. As for the last group, we can see that the accuracy barely reach over 70% and 65%.

The results roughly match the finding we previous got that the model would get poorer performance on the data with higher WER. A custom ASR system or incorporating acoustic features can be the solutions to reduce the impact of high WER.

4) IMPORTANCE OF THE POSITION OF THE CAMERA

In our proposed model, we extract visual feature by applying object detection with YOLOv4 to indicate the user status, i.e. the step which the user is at. The step information is

TABLE 12. WER of the collected voice dataset and the performance.

Data	WER	Accuracy
MM_SVD$_{all}$	22.64%	N/A
MM_SVD$_{train}$	22.67%	N/A
MM_SVD$_{test}$	22.56%	88.31%
MM_RVD$_{test}$	28.55%	79.85%

TABLE 13. WER of MM_RVD$_{test}$ and the performance.

Data	Gender	Nationality	# of data	WER	Accuracy
subject_1	Female	Taiwanese	400	17.33%	96.25%
subject_2	Male	Taiwanese	400	22.18%	85.25%
subject_3	Female	Taiwanese	396	23.80%	81.57%
subject_4	Male	French	396	27.80%	74.24%
subject_5	Female	Indonesian	400	28.27%	83.50%
subject_6	Female	Indonesian	392	28.55%	79.85%
subject_7	Female	Taiwanese	384	37.47%	71.35%
subject_8	Male	Taiwanese	388	47.69%	65.98%

crucial to our task, which means the result of object detection plays an important role. Therefore, we expect the position we place the camera displays the view of the workbench clear enough to see all the robot components and avoids robot components be covered by user. After trying different positions of the camera, we found that we can get the best view with the camera on top of the workbench.

V. CONCLUSION

In this work, we present the YMC model, a simple yet efficient way to capture video features for user intent classification task. The core concept of the model is to mask out the unrelated area according to the object detection result, which forces the following classifier to focus on the crucial parts in the image. Also, we implement an unsupervised method, Autoencoder, to encode multi-modal features, i.e. the concatenation of textual and visual features, into smaller dimensions. With these smaller dimensions, we not only execute faster during inference but still remain high performance with high accuracy. The results show that by applying YOLOv4 as the object detection model, we can achieve slightly better performance than the model that applies YOLOv3. From the confusion matrix we can observe how visual context helps the classification of step-dependent classes. Experiment results confirmed that with the integration of textual and masked visual information, we achieved significant improvement on the accuracy of user intent classification.

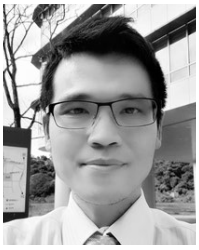
REFERENCES

[1] F. Meziane, S. Vadera, K. Kobbacy, and N. Proudlove, "Intelligent systems in manufacturing: Current developments and future prospects," *Integr. Manuf. Syst.*, vol. 11, no. 4, pp. 218–238, Jul. 2000.
[2] X. Yao, J. Zhou, J. Zhang, and C. R. Boër, "From intelligent manufacturing to smart manufacturing for industry 4.0 driven by next generation artificial intelligence and further on," in *Proc. 5th Int. Conf. Enterprise Syst. (ES)*, Sep. 2017, pp. 311–318.
[3] L. Lattanzi, C. Cristalli, D. Massa, S. Boria, P. Lépine, and M. Pellicciari, "Geometrical calibration of a 6-axis robotic arm for high accuracy manufacturing task," *Int. J. Adv. Manuf. Technol.*, vol. 111, nos. 7–8, pp. 1813–1829, Dec. 2020.
[4] P. R. Jeyaraj and E. R. S. Nadar, "Computer vision for automatic detection and classification of fabric defect employing deep learning algorithm," *Int. J. Clothing Sci. Technol.*, vol. 31, no. 4, pp. 510–521, Aug. 2019.
[5] O. Badmos, A. Kopp, T. Bernthaler, and G. Schneider, "Image-based defect detection in lithium-ion battery electrode using convolutional neural networks," *J. Intell. Manuf.*, vol. 31, no. 4, pp. 885–897, Apr. 2020.
[6] S. Amendaño-Murrillo, C. Dután-Gómez, E. Lema-Condo, and V. Robles-Bykbaev, "Personal robotic assistants: A proposal based on the intelligent services of the IBM cloud and additive manufacturing," in *Proc. IEEE ANDESCON*, Oct. 2020, pp. 1–6.
[7] D. Griol and Z. Callejas, "An architecture to develop multimodal educative applications with chatbots," *Int. J. Adv. Robot. Syst.*, vol. 10, no. 3, p. 175, Mar. 2013.
[8] A. Dersingh, P. Srisakulpinyo, S. Rakkarn, and P. Boonkanit, "Chatbot and visual management in production process," in *Proc. Int. Conf. Electron., Inf., Commun.*, 2017, pp. 274–277.
[9] H. Chen, X. Liu, D. Yin, and J. Tang, "A survey on dialogue systems: Recent advances and new frontiers," *ACM SIGKDD Explor. Newslett.*, vol. 19, no. 2, pp. 25–35, Nov. 2017.
[10] S. Antol, A. Agrawal, J. Lu, M. Mitchell, D. Batra, C. L. Zitnick, and D. Parikh, "VQA: Visual question answering," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2015, pp. 2425–2433.
[11] H. Ben-Younes, R. Cadene, M. Cord, and N. Thome, "MUTAN: Multimodal tucker fusion for visual question answering," in *Proc. IEEE Int. Conf. Comput. Vis.*, Oct. 2017, pp. 2612–2620.
[12] D. Yu, J. Fu, T. Mei, and Y. Rui, "Multi-level attention networks for visual question answering," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jul. 2017, pp. 4709–4717.
[13] P. Gao, Z. Jiang, H. You, P. Lu, S. C. H. Hoi, X. Wang, and H. Li, "Dynamic fusion with intra- and inter-modality attention flow for visual question answering," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2019, pp. 6639–6648.
[14] L. Zhu, Z. Xu, Y. Yang, and A. G. Hauptmann, "Uncovering the temporal context for video question answering," *Int. J. Comput. Vis.*, vol. 124, no. 3, pp. 409–421, Sep. 2017.
[15] M. Tapaswi, Y. Zhu, R. Stiefelhagen, A. Torralba, R. Urtasun, and S. Fidler, "MovieQA: Understanding stories in movies through question-answering," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2016, pp. 4631–4640.
[16] Y. Jang, Y. Song, Y. Yu, Y. Kim, and G. Kim, "TGIF-QA: Toward spatio-temporal reasoning in visual question answering," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jul. 2017, pp. 2758–2766.
[17] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2014, pp. 580–587.
[18] R. Girshick, "Fast R-CNN," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2015, pp. 1440–1448.
[19] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2015, pp. 91–99.
[20] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "SSD: Single shot multibox detector," in *Proc. Eur. Conf. Comput. Vis.*, 2016, pp. 21–37.
[21] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2016, pp. 7263–7271.
[22] J. Redmon and A. Farhadi, "YOLO9000: Better, faster, stronger," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jul. 2017, pp. 7263–7271.
[23] J. Redmon and A. Farhadi, "YOLOv3: An incremental improvement," 2018, *arXiv:1804.02767*. [Online]. Available: http://arxiv.org/abs/1804.02767
[24] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, "YOLOv4: Optimal speed and accuracy of object detection," 2020, *arXiv:2004.10934*. [Online]. Available: http://arxiv.org/abs/2004.10934
[25] N. Dahlbäck, A. Jönsson, and L. Ahrenberg, "Wizard of Oz studies: Why and how," in *Proc. 1st Int. Conf. Intell. User Interfaces*, 1993, pp. 193–200.
[26] S. Tauroza and D. Allison, "Speech rates in British English," *Appl. Linguistics*, vol. 11, no. 1, pp. 90–105, Mar. 1990.
[27] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," 2018, *arXiv:1810.04805*. [Online]. Available: http://arxiv.org/abs/1810.04805
[28] P. Baldi, "Autoencoders, unsupervised learning, and deep architectures," in *Proc. ICML Workshop Unsupervised Transf. Learn.*, 2012, pp. 37–49.
[29] M. Popović and H. Ney, "Word error rates: Decomposition over pos classes and applications for error analysis," in *Proc. 2nd Workshop Stat. Mach. Transl.*, 2007, pp. 48–55.

TZU-YU CHEN received the B.S. and M.S. degrees in computer science and information engineering from National Central University, Taiwan. Her research interests include natural language processing, deep learning, and machine learning.

YU-CHING CHIU received the B.S. degree in computer science and information engineering from Chung Yuan Christian University, Taiwan. Her research interests include natural language processing, deep learning, and machine learning.

NANYI BI received the Ph.D. degree from the Department of Communication, Cornell University. She is currently a Postdoctoral Researcher with National Taiwan University, Taiwan. Her research interests include human–computer interaction, computer-supported cooperative work, and user experience.



RICHARD TZONG-HAN TSAI received the Ph.D. degree in computer science and information engineering from National Taiwan University, Taiwan. He is currently a Professor of computer science and information engineering with National Central University. He is also a Researcher with the Center for Geographic Information Science and the Research Center for Humanities and Social Sciences, Academia Sinica, Taiwan. His primary research interests include natural language processing, deep learning, dialogue systems, cross-lingual information access, sentiment analysis, and digital humanities.