

國立臺灣大學電機資訊學院電機工程學系



碩士論文

Department of Electrical Engineering  
College of Electrical Engineering and Computer Science

National Taiwan University

Master Thesis

具有情緒的自主社交陪伴機器人

Autonomous Social Companion Robot with Moods

吳宗澤

Zong-Ze Wu

指導教授：傅立成 博士

Advisor: Li-Chen Fu, Ph.D.

中華民國 107 年 7 月

July 2018

國立臺灣大學碩士學位論文  
口試委員會審定書

具有情緒的自主社交陪伴機器人  
Autonomous Social Companion Robot with Moods

本論文係吳宗澤君（學號 R05921012）在國立臺灣大學電機工程學系、所完成之碩士學位論文，於民國 107 年 7 月 24 日承下列考試委員審查通過及口試及格，特此證明

口試委員：

傅立成

（簽名）

（指導教授）

李宏毅

李淑亭

陳永銘

蘇木君

系主任、所長

劉志文

（簽名）

## 誌謝



能完成這項研究承蒙許多人的幫助與鼓勵，首先想感謝我的指導教授 傅立成老師。在研究路上總是適時給予我許多的激勵及啟發，讓我能夠不斷地思考改進目前的問題，不斷挖掘自己的潛能讓，面對各式挑戰時永不放棄。同時也感謝口試委員賴金鑫老師、陸哲駒老師、顏炳郎老師及盧璐醫師，老師們所提供專業而具體的建議讓此研究更臻於完善，在此對所有的老師表示深摯的感謝。

在碩士生涯中，機器人組成員的情誼將是值得且難忘的回憶。謝謝所有之前學長姐的經驗傳承與教導，士桓學長的對一開始還是碩一的我的幫助，或是 4c、沛淮、宜倫、曉蓓等等人讓我在碩一的時候可以向他們學習到很多做研究的技巧和精神，這些都是支撐我繼續走下去的支柱。

謝謝 ACL 實驗室所有的夥伴們，謝謝俊偉學長、明理學長與安陞學長在我求學過程中不吝提供學術指導，謝謝助理懿萱、小寧與郁璇在行政事務莫大的協助，謝謝一同努力的 Cesar、Vicente、竣棠、秉蒼、嘉梁、彥程、子駿、侑震、佐薪、孟浩的互相打氣和經驗交流，感謝學弟妹啟維、羽庭平日的關懷。也感謝更多未列名於此的朋友，你們的友情支援讓我的碩士生涯能順利的完成。

特別感謝我最親愛的家人一直以來的無私付出與包容，讓我能夠無後顧之憂的進行求學與研究，並讓我從低潮中再次振作起來。最後再次感謝所有關心著我的人。

吳宗澤 謹致於 2018 年 8 月



## 中文摘要



隨著慢慢高齡化的社會跟勞力的不足，人們對社交陪伴型的機器人能陪伴老人或小孩的認知、意識逐漸增加，這是一個非常困難的問題，因為不論是老人或小孩都會更喜歡擁有一個更人性化的機器人而不是一個冷酷的機器人作為陪伴者。除此之外，為了使社交陪伴機器人能達到更高的自主性，機器人應該要能在沒有使用者的命令下自己做出決定。因此，我們利用穩態理論跟馬斯洛的五大需求來模擬我們的機器人，並架構了架構出自動系統來讓機器人知道每個時刻該做什麼。為了追求人與機器人能有更好的互動，在機器人與人在互動時某種程度上機器人該擁有自己的個性和情緒。與此同時，對於下一個階段的人機互動來說，人機互動不僅僅需要考慮機器人基本問答能力，也要將閒聊能力列入考量。因此，在本篇論文中，我們架構了一個能夠增進機器人與人之間的關係的聊天系統。另外，我們也開發了文本風格轉換的模型，此模型能根據機器人的情緒來轉換閒聊的神經對話系統的輸出句子。值得一提的是，好奇心是人的重要的特性之一，因此我們提出了一個視覺問題生成模型來讓機器人可以針對自己觀察到的景象提出多個問題。最後為了更好的去分析我們的系統，我們會去針對裡面各個提出的模組去做評估，例如我們的自動系統、文本風格轉換模型、視覺問題生成模型。

在實驗結果可以看出我們自動系統可以很好的讓機器人滿足自己的內在需求，並且在 RSR 指標達到 99.98%，而在文本風格轉換模型中，我們的模型可以在 Yelp 的數據集中達到 84.65%成功率來轉換文本的風格，除此之外，我們提出了幾種不同針對機器人的視覺問題生成模型的評估方式，而我們的模型經過評估後，是可以讓機器人有一定水準的互動。

關鍵字：社交陪伴機器人，動態平衡理論，馬斯洛的五大需求，機器人自主系統，文本風格翻譯，視覺問題生成



## ABSTRACT

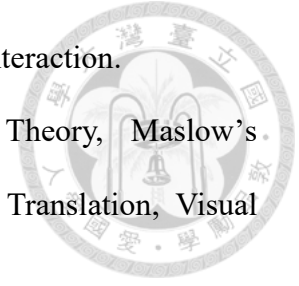


With the gradually aging society and lack of labor, the awareness of social companion robots accompanying elders and children rises. It is a very challenging problem, since the elders and children might prefer having a more humanlike robot rather than having a cold bloodless robot as friends. Moreover, for a social companion robot to reach high autonomy, it should make its own decisions without users' command. Therefore, homeostatic theory and Maslow's hierarchy of needs have been adopted to model the robot, and we build the autonomous system for robot to know what to do at every time moment. In order to pursue a better interaction between robot and human, it might be better for robot to possess personality and moods to some extent while interacting with human. Meanwhile, not only basic Question-Answering abilities but also abilities for Chit-Chatting should be considered crucial for the next level interactions. Therefore, in this research work, we build up a dialogue system for improving the relationship between robot and human. What's more, we develop a text style translation model to translate the style of the output sentence from chit-chat bot based on robot's moods. It is quite worthy to mention curiosity, which is important characteristic of human, and thus we develop visual question generation capability such that robot knows how to propose diverse questions to human for what it has observed. Finally, we evaluate our system separately for the three proposed modules, such as the autonomous system, text style translation module, and the visual question generation module.

The results show that the autonomous system can make the robot satisfy the internal needs well and reach 99.98% in RSR index. In text style translation, the transferred sentiment accuracy in the Yelp dataset can reach 84.65%, which is an improvement. With the proposed evaluation metrics, which is designed for the visual question generation of

our robot, we can say that robot will have a certain quality for this interaction.

Keywords: Social Companion Robot, Homeostasis Theory, Maslow's Hierarchy of Needs, Autonomous System for robot, Text Style Translation, Visual Question Generation



# CONTENTS



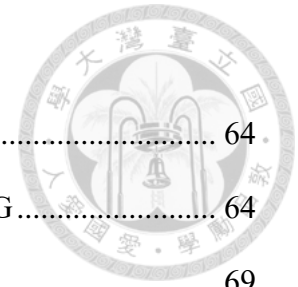
口試委員會審定書 .....	#
誌謝 .....	i
中文摘要 .....	ii
ABSTRACT .....	iv
CONTENTS .....	vi
LIST OF FIGURES .....	ix
LIST OF TABLES .....	xii
<b>Chapter 1 Introduction .....</b>	<b>1</b>
1.1 Background.....	1
1.2 Motivation .....	1
1.3 Related Work .....	3
1.3.1 Decision Making Systems .....	3
1.3.2 Text Style Translation .....	6
1.3.3 Visual Understanding and Question Generation .....	9
1.4 Objective and Contribution .....	10
1.5 Thesis Organization .....	12
<b>Chapter 2 Preliminaries .....</b>	<b>13</b>
2.1 Markov Decision Process .....	13
2.1.1 Reinforcement Learning.....	15
2.1.2 Deep Learning .....	17
2.1.3 Deep Reinforcement Learning.....	17
2.2 Convolutional Neural Network .....	22





2.2.1	Convolutional Layer .....	22
2.2.2	Pooling Layer .....	23
2.2.3	Fully Connected Layer .....	24
2.2.4	VGG-16 Net .....	24
2.3	Recurrent Neural Network.....	25
2.3.1	Long Short-Term Memory.....	26
2.3.2	Gated Recurrent Unit.....	27
2.4	Generative Model .....	28
2.4.1	Auto-Encoder.....	29
2.4.2	Variational Auto-Encoder .....	30
2.4.3	Generative Adversarial Network .....	31
<b>Chapter 3</b>	<b>Methodology .....</b>	<b>34</b>
3.1	System Overview.....	34
3.2	Autonomous System.....	36
3.2.1	Drive .....	37
3.2.2	Stimulus and Environment .....	39
3.2.3	Motivation .....	40
3.2.4	Action .....	41
3.2.5	Mechanism for Moods.....	44
3.2.6	Decision Making System.....	45
3.3	Dialogue System.....	49
3.3.1	Text Style Translation.....	54
3.3.2	Formulation and Model Architecture of Text Style Translation.....	54
3.3.3	Gumbel-Softmax and Professor Forcing for Text Generation.....	60
3.3.4	N+1 Discriminator with Adversarial Training and Cycle Consistency	

Loss	62
3.4 Visual Question Generation.....	64
3.4.1 Problem Formulation and Model Architecture of VQG.....	64
3.4.2 The Loss Function and Adversarial Training .....	69
<b>Chapter 4 Experiment.....</b>	<b>73</b>
4.1 Autonomous System.....	73
4.2 Dialogue System.....	80
4.3 Visual Question Generation.....	83
<b>Chapter 5 Conclusion .....</b>	<b>88</b>
<b>REFERENCE .....</b>	<b>91</b>



# LIST OF FIGURES



Fig. 2. 1 The graph representation of the Markov Decision Process.....14

Fig. 2. 2 Q-Learning algorithm with epsilon greedy exploration strategy .....16

Fig. 2. 3 Three main method of reinforcement learning.....18

Fig. 2. 4 Deep Q-Learning algorithm .....20

Fig. 2. 5 The architecture of dueling network. The upper one-dimension red block before the green process is the state-value function, and the lower red block is the action advantage function with the dimension being the number of actions. The green process is defined as in the equation (2. 14).....21

Fig. 2. 6 The architecture of ZF Net. ....22

Fig. 2. 7 The example of the process of the convolution. In the first row, the filter in blue will do the dot product with the image map in the yellow color. Therefore, after the dot product operation, the output the value is 3, and the same process for the second and third row. ....23

Fig. 2. 8 The example of how the max-pooling with  $2 \times 2$  mask and the stride 2. ....24

Fig. 2. 9 The architecture of VGG-16 net. C is the number of the classes.....24

Fig. 2. 10 The operation process and unfold process of the RNN.....25

Fig. 2. 11 The mechanism of LSTM and the operation process will be shown in (2. 16).....26

Fig. 2. 12 The mechanism of GRU and the operation process is shown in the (2. 17). ...27

Fig. 2. 13 The idea of auto-encoder. ....29

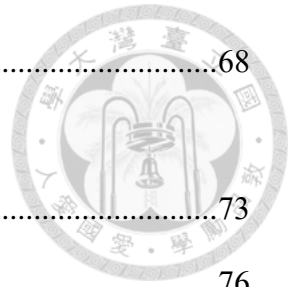
Fig. 2. 14 The example result of the auto-encoder. The upper row is the testing input image, and the lower row is the reconstruction result. ....30

Fig. 2. 15 The idea of Variational Auto-Encoder (VAE) .....31



Fig. 2. 16 The data flow of the GAN.....	32
Fig. 2. 17 The algorithm of the Generative Adversarial Network.....	33
Fig. 3. 1 The proposed autonomous system. ....	34
Fig. 3. 2 The decision making system in the autonomous system.....	35
Fig. 3. 3 The dialogue system of social chat in the robot actions.....	36
Fig. 3. 4 The operational flow within Homeostasis.....	37
Fig. 3. 5 Maslow’s hierarchy of needs.....	38
Fig. 3. 6 The model architecture of the DDQN in this work for decision making system. All the green blocks mean the fully connected layers, and the number next to it is the number of neurons in the layer.....	48
Fig. 3. 7 The algorithm for DDQN in the decision making system. ....	49
Fig. 3. 8 The webpage interface on the pad of the robot. ....	50
Fig. 3. 9 The flow of the dialogue management.....	51
Fig. 3. 10 The idea of the text style translation in this work. ....	55
Fig. 3. 11 The learning flow of the text style translation module in the work.....	56
Fig. 3. 12 The model architecture of the auto-encoder $E_y$ . The 200 dimensions vector is the latent variable $y$ .....	56
Fig. 3. 13 This is the model architecture of the encoder $E_z$ . ....	57
Fig. 3. 14 This is the model architecture of the generator $G$ . ....	58
Fig. 3. 15 The idea of aligning the sentence population.....	61
Fig. 3. 16 The concept of the VQG in this work. ....	65
Fig. 3. 17 The model architecture of the image encoder $E_I$ . ....	66
Fig. 3. 18 The model architecture of the encoder $E_z$ .....	67
Fig. 3. 19 The model architecture of the generator $G$ . ....	67

Fig. 3. 20 The model architecture for the discriminator D2.....	68
Fig. 4. 1 The Pepper robot.....	73
Fig. 4. 2 The reward of dueling deep q network.....	76
Fig. 4. 3 The secure rate in the training process.....	77
Fig. 4. 4 The intensity of motivations changes through time. The x-axis is the time step, and the y-axis is the intensity value.....	78
Fig. 4. 5 The sequence of actions in this epoch.....	79
Fig. 4. 6 The value for moods.....	79



# LIST OF TABLES

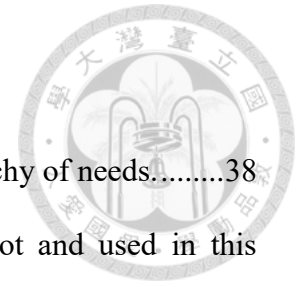


Table. 3. 1 The difference of drives compared with Maslow’s hierarchy of needs.....	38
Table. 3. 2 The stimuli can be sensed by the sensors of the robot and used in this work. ....	40
Table. 3. 3 The motivations which is correlated to each drive in the system. ....	41
Table. 3. 4 The categories of Questions and how the functions react correspondingly...	52
Table 4. 1 The table of drive change through time. ....	74
Table 4. 2 The table of the decreased value of drive due to the action, and the execution time of the action. ....	74
Table 4. 3 The values in the table are $fst(\cdot)$ which affect the intensity of the motivation .....	75
Table 4. 4 The sentiment accuracy of transferred sentences evaluated by a pre-trained classifier. ....	81
Table 4. 5 The positive sentences are transferred to the negative sentences. ....	82
Table 4. 6 The negative sentences are transferred to the positive sentences. ....	82

# Chapter 1 Introduction



## 1.1 Background

In the past decades, the robotic researches and applications have received significant attention in fields of service robots, as compared with those of the industrial robots [1]. More and more commercial robots targeting to provide personal services have been released to the market. Despite such developments have always been humans' vision, and even more unrealized ones seen only in current scientific fiction movies are on the way, the endeavors made so far for service robotics can achieve usually single and simple purpose, like the most famous example being Roomba [2]. These simple-goal robots have their own specific goal, and all the actions of the robot are planned around that purpose.

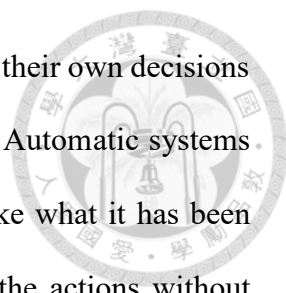
More recently, more general purpose service robots have been developed more widely, mainly to be deployed in the domestic environment like, nursing home, regular home, and expected to be involved in our daily life. There are numerous examples, say, personal robots [3][4], office robots [5][6], healthcare robots [7][8], home service robots [9][10], etc. However, those robots are unlike task-specific robot, having a specific goal. Lacking a specific goal makes them hard to act and behave on their own, and they can only wait for human's commands, which lower the autonomy of the robot.

Moreover, people regard them not only as assistants that remind us of the schedule and take reservations, but also as ones who can accompany and socially interact with the families. For the latter, as a companion, we anticipate that robots can behave in a more human manner, and participate in our life with humanlike interactions.

## 1.2 Motivation

In order to achieve social comfort of the robot, we should raise their autonomy and improve their skills to act in a more humanlike way for having better interactions.





Autonomy which is different from automaticity is the ability to make their own decisions based on what have been perceived and actuate in the environment. Automatic systems only have the ability to follow the human's orders or commands like what it has been programmed, but autonomous systems have their free wills to do the actions without human intervention [11]. Several research works [12][13][14][15] also gave the similar definitions. Therefore, to achieve the higher level of autonomy, the robot should be able to know what to do at every time moment according to the environment status and its own free will rather than waiting for human's command.

For achieving autonomy, a toy robot as an example , Sony's Aibo [16], senses its internal states and shows the ability to tell proprioceptively that it needs to be charged for self-maintenance. As for larger robot like the satellite and spacecraft, they are required to be autonomous to reach their destinations, and they are able to react quickly for what they need and reduce the cost of human supervision significantly [12]. To achieve the autonomy of the robot, one popular method is inspired biologically from animals and human [17]. The idea is to fully study and model the animal or human behaviors, and use the same way for building agents' model, making them fully autonomous. Such model is widely used not only for designing the mechanism of the robots [18][19], but also for controlling of robots, social robots [20][21], etc.

As the aging population and low birth rate are major issues among the developed countries, the companion and caring for the elders are more and more desperate nowadays. Due to the heavy burden of the elder caring, researcher have started trying to bring the social companion robot for it. Being a social companion robot, it should act more humanlike to have a better interaction with human. Otherwise, human will easily get tired of a feeling-less machine. With the advance of AI and various technology, researchers focus not only on safety physical interaction but on social interaction as well. For the

latter, researchers start modeling the robot as a human, and try to build up the emotion in it such that humans tend to feel that robot is not a machine anymore [22][23]. During the interaction, a robot will have different moods based on how it perceives human emotions and its own feeling.

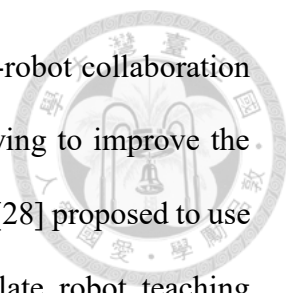
Moreover, if robot has its own curiosity after observing the environment and initiate the interaction by asking some questions for satisfying its own curiosity, it can increase the connections and social interactions between the robot and human. It can also be a good opening topic for starting the chatting, and researchers in the field of human-robot interaction, believe that taking initiative in interactions can result in having more fluent interactions [24][25]. After all, as a social companion robot, chatting is an indispensable skill in interactions. As a result, one can see that different moods and the curiosity of the robot do enrich the content of interaction.

## **1.3 Related Work**

In order to answer the question about how the autonomy works for the decision making of a robot, we turn to research in decision making systems. After that, for embedding the moods in the chatting, how to do text style translation needs to be surveyed as well. Finally, we review literatures on visual understanding and question generation to build up the curiosity-like ability of the robot.

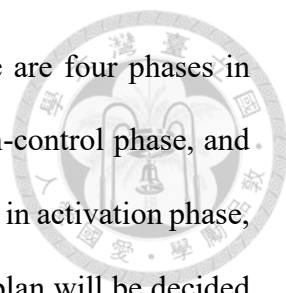
### **1.3.1 Decision Making Systems**

There are various methods in decision making, and the popular ones are Markov models, Bayesian models, and so on. Gillini *et al.* [26] uses Markov Decision Process (MDP) to model the navigation of shopping assistance robot in the shopping area to gain the optimal behavior. Then, it will choose the proper actions like paying bills, collecting items, or navigating at each time moment. Cui *et al.* [27] combines MDP and State-



Action-Reward-State-Action (SARSA) algorithm to solve the multi-robot collaboration in 3D soccer simulation game in Robocup. SARSA algorithm is trying to improve the drawback of slow convergence speed in traditional MDP. Chen *et al.* [28] proposed to use Partial Observable Markov Decision Process (POMDP) to formulate robot teaching problem for industrial robot applications and deal with image features with uncertainties by Successive Approximation of Reachable Space under Optimal Policies (SARSOP) algorithm. Elinas *et al.* [5] modeled the task of a visually guided interactive mobile robot with factored POMDP. In addition, they have presented a heuristic to speed up the process of large POMDP. Omidshafiei *et al.* [29] solved the probabilistic multi-robot coordination problems in continuous space with Decentralized Partially Observable Semi-Markov Decision Process (Dec-POSMDP). They incorporated the macro-action for simplifying the Dec-POMDP to Dec-POSMDP, which can be solved by discrete-space search method. Feyzabadi *et al.* [30] proposed a hierarchical solution of Constrained Markov Decision Process (CMDP) to multi-objective sequential decision problems for robots, and it is implemented in a path planning planar. They partitioned the states into a set of macro-states to guarantee the connectivity preservation, and estimated the relevant parameters like transition probability by Monte Carlo sampling.

As mentioned previously, some approaches are inspired biologically, and researchers try to mimic how mind and behavior of human work neuro-biologically or psychologically. Maes [31] proposed an Agent Network Architecture (ANA), consisting of a distributed set of different competence modules, like action module and belief module, and it is a behavior-based architecture using a spreading network. Bellas *et al.* [32] used artificial neural network (ANN) to represent the models for Multilevel Darwinist Brain (MDB). The world, internal, and satisfaction models will adapt the behavior for trying to maximize the satisfaction. Ando *et al.* [33] modeled self-sufficiency



system for communication robot by using “urge system”, and there are four phases in their system such as activation phase, decision-making phase, action-control phase, and post-mortem phase. An “urge” will be activated by situation cognition in activation phase, and the action plan(s) of that “urge” will start influence. The action plan will be decided by ranking priority in decision-making phase, and the action will be proceeded in action-control phase. Finally, in post-mortem phase, it will try to evaluate the success or failure of this action plan. Hoefinghoff *et al.* [34] proposed a framework for creating robot applications usable for non-experts, and those users are able to create it by a decision making framework for robot companion. The author used the somatic markers, the possible emotions of picking the action according to the stimuli, and aimed to filter out the inappropriate actions. Wilson [35] enabled the robot to be a long-term companion in the home by computation models of affective appraisal, empathy, and moral decision making.

For psychological approaches, homeostasis drive theory based systems are one of the dominant methods. Cannon [36] is the first one who introduced homeostasis, and was described as that a regular system needs to maintain or balance the body in a stable physiological state. In a robotic decision making system, many robots adopted this method, such as Sony’s Aibo [37], MIT Kismet [38], and ROBEE in Cao *et al.* [39], etc. Robot will try to maintain its internal needs by doing different actions. Cao *et al.* [39] proposed ROBEE which is an architecture of which the internal needs generate drive, and a drive only has one satiator for deciding the action to compensate the highest needs. Cañamero *et al.* [40] adopted the homeostatic approach to model the motivations of the robot, and the intensity of motivations are be influenced by the drive and the external stimuli. After obtaining the intensity, the behavior of highest intensity will be chosen. Gandaho *et al.* [41][42] used reinforcement learning algorithm on controller for learning

and adaptation of the robot.

Inspired by both Cañamero and Gandoho, Castro-González [43] modeled the internal needs of robot by drives, and determined the intensity of the motivations along with both drive and stimuli. The correlation between actions and dominant motivation is not predefined, and must be learned through the interaction. Both our previous works [44][45] follow the formulation similar to [43], while introducing the human intention and feedback to the decision making system, which makes the robot serve proactively and more human-aware. However, for improving the robot to behave in a more humanlike way, this work tries to re-define the internal needs inspired by Maslow's hierarchy of needs [46], and we will explain it detailedly in Chapter 3. We also formulate the moods for the robot for having a better and interesting interaction between robot and human. In addition, in order to approach more optimal goal, we will compute the learning process for every intensity of motivations, not limited to a single dominant motivation. Finally, we try to make the social companion robot feasible in the real world.

### 1.3.2 Text Style Translation

In machine translation or summarization, using large amount of parallel data has been important in text generation task. However, there are not many parallel data for text generation problem, such as style transfer, decipherment, and so on. Therefore, researcher focused on generating the text by learning from non-parallel data or mono-lingual data. While generating the text, the content or the meaning of the source sentence need to be preserved, and try to generate it with the desired constraints.

Non-parallel style transfer for vision fields has been extensively researched, and has also received significant attention. The main goal for those papers are going to learn the joint distribution of image data under different domains. Gatys *et al.* [47] generate the

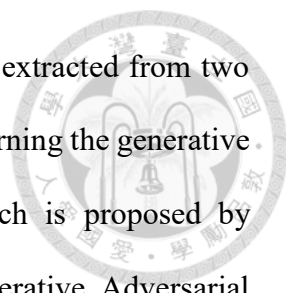
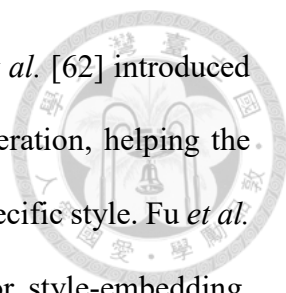


image by combining the content and style representation which are extracted from two different source images. Recently, the approaches are focusing on learning the generative neural network by using the generative adversarial training which is proposed by Goodfellow *et al.* [48]. Liu *et al.* [49] proposed Coupled Generative Adversarial Networks (CoGAN) which enforces weight-sharing constraint on the layers of decoding abstract semantics to learn the representations of cross-domain. Later on, Liu *et al.* [50] proposed new unsupervised image-to-image translation framework based on CoGAN, and they used not only the weight-sharing mechanism but also the shared latent space for the content representation for variational auto-encoders and generative adversarial networks. Zhu *et al.* [51] introduced the cycle consistency loss to enforce the inverse mapping better for preserving the better source content. At the same time, Kim *et al.* [52] and Yi *et al.* [53] proposed DiscoGAN and DualGAN which are very similar as the CycleGAN in Zhu *et al.* [51]. In addition, Zhu *et al.* [54] proposed to learn how to manipulate the latent representation of the image for changing the shapes and colors. Nevertheless, the method in style transfer in vision cannot be directly applied to the style transfer in natural language processing fields. Due to the difficulties of discreteness of natural language while generating, researches on text style translation are still in an early stage.

There are various approaches for doing text style translation. In tradition, Brown *et al.* [55] used a statistical approach to do the translation from French to English. Later, many researchers adopted databases or parallel corpora to facilitate the translation, such as WordNet [56], PPDB [57], and used probabilistic translation models[58], Markov Logic Network [59] or graph theory [60] for inference. Lately, neural machine translation comes out, and it utilizes non-parallel data for learning the sentence generation indirectly. Ficler *et al.* [61] proposed to use conditioned RNN language model, while desired



semantic content and style will serve as conditional contexts. Han *et al.* [62] introduced two switches with seq2seq to control the style of the sentence generation, helping the model to capture the semantic content in a style and to decode in a specific style. Fu *et al.* [63] proposed two models, one for multi-decoder, and another for style-embedding. Adversarial training is used to separate the content from the style. Mueller *et al.* [64] utilized a classifier to guide the modification of the latent representation in variational auto-encoder (VAE) and attained the sentence with desired attribute. Hu *et al.* [65] presented a new neural generative model combining VAE with adversarial training. VAE encodes the sentence into a latent representation and ties it with the style to generate specific style sentence. Shen *et al.* [66] introduced cross-alignment training to align both latent representation and sentence population with Professor-Forcing algorithm [67] and continuous relaxation for discrete sampling process [68], and they assumed the source and target domain sentence share the same latent space. Very similar to Shen *et al.* [66], Lample *et al.* [69] presented the method with alignment to the latent representation, and two decoders are used to generate the sentence with specific styles. Zhao *et al.* [70] proposed style discrepancy loss and cycle consistency loss to handle the arbitrary style in the source domain, but a disadvantage is that it can only have one target style in target domain. Melnyk *et al.* [71] introduced the attention mechanism to the auto-encoder part in Shen *et al.* [66], and reduced the parameters for multiple styles by a collaborative classifier.

This work is largely influenced by Shen *et al.* [66], Zhao *et al.* [70], and Melnyk *et al.* [71]. We utilize a pre-trained language model for speeding up the convergence, and preserve better style information with an auto-encoder. In addition, cycle consistency loss here is also to preserve a better semantic content, and N+1 discriminator [72] similar to collaborative classifier in Melnyk *et al.* [71], which is a good way to handle possible

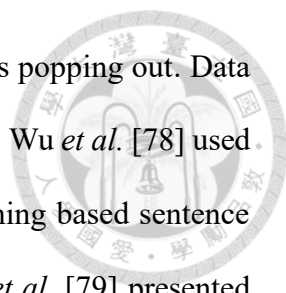


multiple moods for the robot in the future.



### 1.3.3 Visual Understanding and Question Generation

In general, one way to show the ability of understanding of an image is to describe the image, namely, caption generation. The task of caption generation needs to generate a paragraph or a sentence for describing the content in the image or a short clip of the video. There are several approaches for visual understanding and question generation, such as knowledge-based approach, data driven approach, etc. In knowledge-based approach, researchers utilize the information with high-level description in structured representation. Aditya *et al.* [73] integrated deep learning based perception module (object, scene, and scene constituent recognition) and the concept modeling from commonsense knowledge obtain from the text, and proposed scene description graph (SDG) to encode the relations among the entities from perception module. Then, they utilized the template based sentence generation (SimpleNLG [74]) to generate the sentence. Wang *et al.* [75] developed a knowledge-based approach to solve visual question generation (VQA) task. Resource Description Framework (RDF) is utilized for graph construction consisting of visual concepts of the image, and each concept can be linked to DBpedia [76] entities with the same semantic meaning. The system can not only answer the questions using concept not contained in the image but also explain the reason how it develops the answer. Wu [77] also proposed to use deep learning perception module for high level structured information, and integrated the existing commonsense knowledge base (concept net) and the word embedding method to associate the structured information and topic database with different concept nodes for the following conversation. However, the limited database for the sentence generation is a big disadvantage in knowledge-based approach for the social companion robot, because

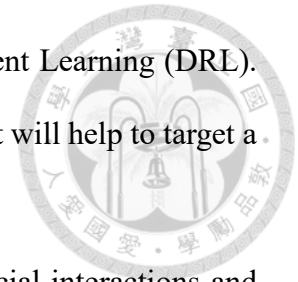


human will quite easily get tired of the same sentence which keeps popping out. Data driven approach with deep learning does not suffer the same problem. Wu *et al.* [78] used external knowledge and encode it into vector for input in deep learning based sentence generation. For improving the various caption generation, Johnson *et al.* [79] presented Dense Captioning (DenseCap) which is similar to Faster R-CNN [80] and combines recognition network, localization network, and sentence generation, and it localizes and describes different regions of an image. Dai *et al.* [81] introduced conditional generative adversarial network with policy gradient for solving the image captioning. The generated sentences are more natural and diverse as compared to those from MLE-based model. Jain *et al.* [82] improved the diversity and creativity of the sentence generation by using variational auto-encoder approach. Based on Jain *et al.* [82], Wang *et al.* [83] proposed an additive Gaussian encoding space to the latent representation in variational auto-encoder. This work is also highly inspired by Dai *et al.* [81] and Jain *et al.* [82], and we utilize variational auto-encoder with Gumbel-Softmax to for encoding phase of the latent representation for a more informative latent representation, compared with using conditional generative adversarial network, and introduce the discriminator for enforcing the association between the generated sentence and the image which is kind of advantage in Jain *et al.* [82].

## 1.4 Objective and Contribution

For the purpose of developing an autonomous social companion robot with moods, the system targets at three main objectives. The first objective is to improve the autonomy of the robot, by letting the robot know what to do at every moment, and also act in humanlike manner. Therefore, inspired by Maslow's hierarchy of needs, we re-define the internal needs in homeostatic modeling part. In order to make autonomous system work

for the robot, decision making system is built by Deep Reinforcement Learning (DRL). DRL is utilized to consider all kinds of intensity of motivation, and it will help to target a more optimal long-term goal and consider more comprehensive.



Secondly, in order to make the robot have more interesting social interactions and behave more in a human way, we build the positive and negative moods for the robot. In addition, we integrate the sequence to sequence with attention chat-bot module with text style translation module, and build this chatting system on our robot. In text style translation module, we utilize the pre-trained language model to speed up the convergence, and train it with cycle consistency loss for preserving a better content from source sentence. For possible multiple moods for robot, we utilize N+1 discriminator to reduce the needed parameters and it is also a better way for adversarial training here. Moreover, other basic interactions are also built up on our robot, such as asking weather, time, map, news, basic knowledge, playing music, taking photo, and so on.

Finally, curiosity is also a significant characteristic of a human, in result, we consider it as one of internal needs of the robot, too. Deep learning based approach for visual question generation is built through the visual perception system from our robot. For the purpose of preventing the human from easily getting tired of the robot, we expect that robot can ask diverse questions through observing a single scene. We utilize variational auto-encoder with Gumbel-softmax for the encoding phase of the latent representation, and Professor-Forcing algorithm and continuous relaxation for discrete sampling process are used to generate the sentence. What's more, adversarial training is used to match the image and the generated sentence better.

## 1.5 Thesis Organization

The rest of this thesis is organized as following. In the Chapter 2, we are going to introduce some preliminaries for this work. First, the basic idea of reinforcement learning, and deep reinforcement learning is given, and it is applied to our decision making system. Then, the convolution neural network, recurrent neural network, and generative models are mentioned.

In Chapter 3, the proposed system for the autonomous social companion robot is been presented. At first, we will give the overview of the proposed system. Then, the autonomous system with homeostasis and the decision making system are introduced. After that, we will explain the mechanism of the moods for the robot, and how we develop the text style translation module for changing to the positive or negative style. Finally, how we utilize and develop the visual question generation is given.

In Chapter 4, the evaluations of the autonomous system, text style translation model, and visual question generation model are shown. At the end, we make the conclusion in Chapter 5.



## Chapter 2 Preliminaries

### 2.1 Markov Decision Process

For each time step in Markov Decision Process (MDP), the process or the agent is in some state  $s$ , and the decision maker of the MDP will choose the action  $a$  which is available in the state  $s$ . Then, the process will move to a new state  $s'$ , and get a reward  $r_a(s, s')$ . The probability for state  $s$  moving to state  $s'$  is decided by state transition  $t_a(s, s')$ . Formally speaking, MDP is a discrete time stochastic control process, and it is a 5-tuple  $(S, A, T, R, \gamma)$ .  $S$  is the finite set of states,  $A$  is the finite set of action,  $T$  is the state transition function mapping from state and action to next state with probability like in (2. 1).

$$\begin{aligned} T: S \times A &\rightarrow S \\ T(s, a, s') &= P(s'|s, a) \end{aligned} \tag{2. 1}$$

$s, s' \in S, a \in A$ , and  $P(s'|s, a)$  is the probability of moving toward  $s'$  given the current  $s$  and action  $a$ .  $R$  is reward function that maps the state-action-state pair to a bounded finite number like in (2. 2).

$$\begin{aligned} R: S \times A \times S &\rightarrow \mathbb{R} \\ |R(s, a, s')| &\leq R_{max} \end{aligned} \tag{2. 2}$$

$R_{max}$  is the maximum absolute number of the reward function, and  $\gamma$  is the discounting factor for the expected accumulated reward bounded in  $0 < \gamma < 1$ . Fig. 2. 1 shows the comprehensive illusion of the relationship. Given the MDP, our goal is to find the optimal policy  $\pi^*: S \rightarrow A$  which chooses the action based on the state  $s \in S$  and tries to maximize the expected accumulated reward. There are several methods for solving the optimal policy, such as policy iteration and value iteration.



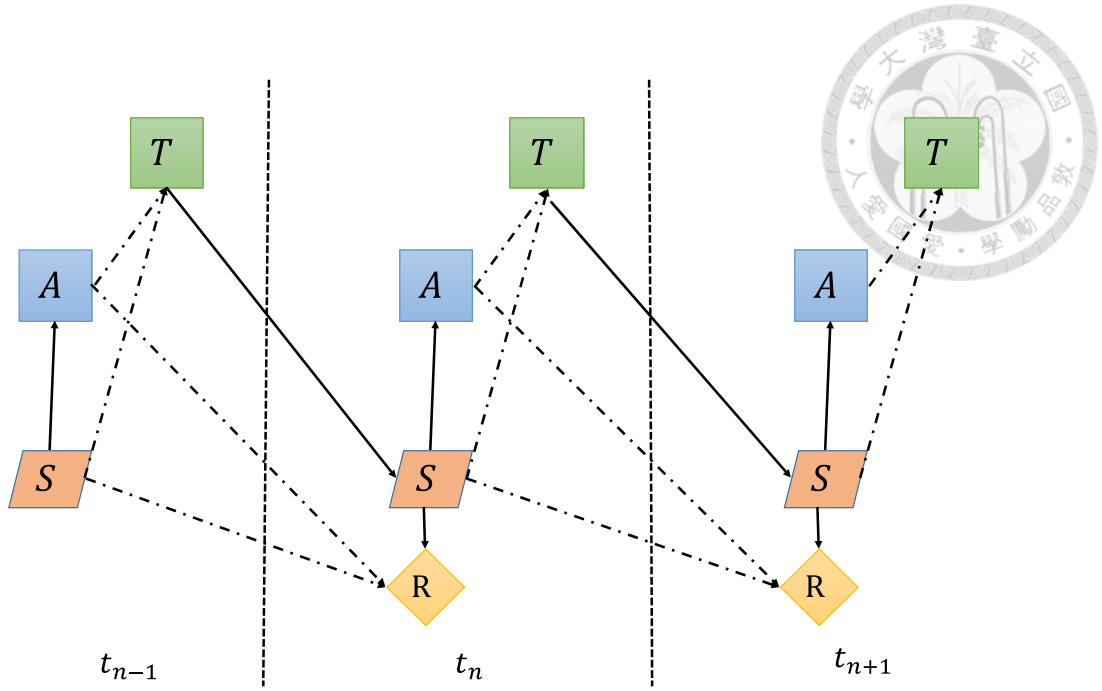


Fig. 2. 1 The graph representation of the Markov Decision Process

In order to explain the method that we utilize in the decision making system, we introduce how the basic value function works to calculate the expected accumulated reward. The expected accumulated reward under the policy  $\pi$  with immediate reward  $R_{t+1}$  and discounting factor  $\gamma$  from the initial state  $s$  will be defined as following:

$$\begin{aligned}
 V_{\pi}(s) &= \mathbb{E}_{\pi}[R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \dots | S_t = s] \\
 &= \mathbb{E}_{\pi}[R_{t+1} + \gamma(R_{t+2} + \gamma R_{t+3} + \dots) | S_t = s]
 \end{aligned} \tag{2.3}$$

The future rewards are less important than the recent rewards, therefore, the future rewards are discounted by the discounting factor  $\gamma$ . The optimal value function can be determined in an iterative manner. We can rewrite the equation (2.3) into:

$$V_{\pi}(s) = \mathbb{E}_{\pi}[R_{t+1} + \gamma V_{\pi}(s') | S_t = s] \tag{2.4}$$

Assuming that the reward and transition function are given, we can reformulate the equation (2.4) into:

$$V_{\pi}(s) = R(s, \pi(s)) + \gamma \sum_{s' \in S} T(s, \pi(s), s') V_{\pi}(s') \quad (2.5)$$

The optimal state-value function  $V^*$  is the maximum value function over all policies, and it is defined as:

$$V^*(s) = \max_a (R(s, \pi(s)) + \gamma \sum_{s' \in S} T(s, \pi(s), s') V^*(s')) \quad (2.6)$$

Similarly, the optimal expected long-term reward of taking action  $a$  in a state  $s$  by following the policy  $\pi$ , which is defined as the action-value function and also named as the Q-value function.

$$Q^*(s, a) = R(s, a) + \gamma \sum_{s' \in S} T(s, \pi(s), s') V^*(s') \quad (2.7)$$

Due to the Bellman equation, it can be proved that the approximated value function can converge to the optimal value. In addition, the optimal policy always chooses the action with the maximal long-term reward (e.g. the action maximizes the Q-value function), and the optimal policy  $\pi^*(s)$  will be formulated as:

$$\pi^*(s) = \arg \max_a Q^*(s, a) \quad (2.8)$$

## 2.1.1 Reinforcement Learning

Given the MDP, our goal is to find the optimal policy for reinforcement learning. The iteration-based method which we mentioned previously can converge eventually, nonetheless, there are still others better algorithm for speed up the convergence of the reinforcement learning, such as Q-learning, SARSA, etc.

In Q-learning algorithm, there is a table for  $Q(s, a)$  values for each different discrete state-action pairs, which is named as Q-table. It can also be proven that the Q table will converge to the optimal one  $Q^*(s, a)$ . Assuming that the current state of the



agent is  $s_t$ , and the agent takes the action  $a_t$  and observes the next state  $s_{t+1}$ . The agent will also get immediate the reward  $r_{t+1}$ , and the update mechanism will be defined as following:

$$Q_t(s_t, a_t) \leftarrow Q_t(s_t, a_t) + \alpha(r_{t+1} + \gamma \max_a Q_t(s_{t+1}, a_t) - Q_t(s_t, a_t))$$

(2. 9)

where  $\gamma$  is the discounting factor, and  $\alpha$  is the learning rate.

In addition, we will utilize the  $\epsilon - greedy$  method to improve the exploration of the Q-learning for not stuck at the local minimum. To sum up, the Q-Learning algorithm is shown in the Fig. 2. 2.

---

**Algorithm 1** Q-Learning

---

**Require:**

State  $S = \{1, \dots, n_s\}$   
 Action  $A = \{1, \dots, n_a\}, A : S \Rightarrow A$   
 Reward function:  $R(s, a), R : S \times A \rightarrow \mathbb{R}$   
 (Probabilistic) Transition function  $T : S \times A \rightarrow S$   
 Learning function  $Q, Q : S \times A \rightarrow \mathbb{R}$   
 Learning rate:  $\alpha \in [0, 1]$   
 Discounting factor:  $\gamma \in [0, 1]$   
 Epsilon value:  $\epsilon \in [0, 1]$

**Ensure:**

optimal Q-function  $Q^*(s, a)$

- 1: **procedure** QLEARNING( $S, A, R, T, \alpha, \gamma, \epsilon$ )
- 2:   Initialize  $Q : S \times A \rightarrow \mathbb{R}$  randomly
- 3: *loop:*
- 4:   **while** Q is not converged **do**
- 5:     Start in arbitrary state  $s \in S$
- 6:     **while** s is not terminal **do**
- 7:       Manipulate the  $\pi$  based on Q with the  $\epsilon - greedy$  strategy
- 8:       **if**  $\epsilon \geq 0.05$  **then**
- 9:          $\pi(s) \leftarrow$  random action
- 10:       **else**
- 11:          $\pi(s) \leftarrow \arg \max_a Q(s, a)$
- 12:        $a \leftarrow \pi(s)$
- 13:        $r \leftarrow R(s, a)$
- 14:        $s' \leftarrow T(s, a)$
- 15:        $Q(s, a) \leftarrow Q(s, a) + \alpha(r + \gamma \max_{a'} Q(s', a') - Q(s, a))$
- 16:        $s \leftarrow s'$
- 17:     **return** Q

---

Fig. 2. 2 Q-Learning algorithm with epsilon greedy exploration strategy



## 2.1.2 Deep Learning

Deep learning is nothing new but still a neural network, and it can be viewed as a black box algorithm modeling the data distribution of the training data. There are three main categories which are unsupervised learning, semi-supervised learning, and supervised learning. In supervised learning, you have input variables and output variables, and utilize the learning algorithm for the mapping function from the input to the output. The goal is to have an approximated mapping function so that while having new input data, we can use it to predict the output which matches training data distribution. For example, in the problem of object recognition, the input is the image and the output will be the ground truth of the categories of object, such as human, cup, etc. On the other hand, unsupervised learning only has the input data, and doesn't have the ground truth output. The ground truth output of the testing data is usually determined by some clustering algorithm such as k-means, k-nearest neighbor. In result, the goal here is to learn how to infer the structure within the input data. Nevertheless, semi-supervised learning will utilize the existing training data of input and labeled output to predict the unlabeled input data, and trained with the predicted data which exceed the threshold of confident score.

## 2.1.3 Deep Reinforcement Learning

There are also three categories for deep reinforcement learning like in the reinforcement learning, for instance, value-based, policy-based, and actor-critic method, which is shown in the Fig. 2. 3. Value-based method mainly focuses on the learning the critic namely the value function, and the policy-based method learns the actor which is the policy function. However, both of them have their own pros and cons. Value-based

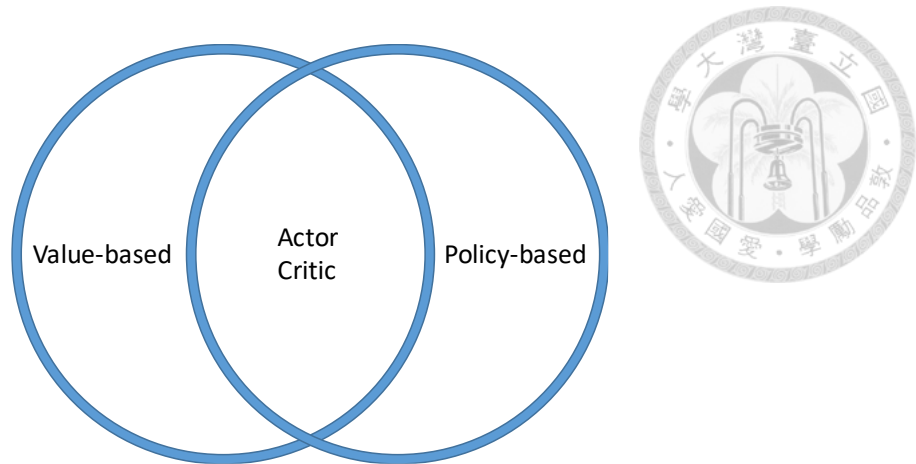


Fig. 2. 3 Three main method of reinforcement learning

method take the advantage of converging to the global optimal, but it is difficult for the algorithm to converge with a good convergence speed. In contrast, policy-based method can converge faster than the value-based method, but it will easily converge to the local optimal. Then, actor-critic method combines the policy function with the value function for estimating whether the action is suitable for the state or not.

We will focus the value-based method in this section because of the method used in the decision making system. In deep Q-Learning, the origin Q-table of the Q-Learning will be substituted by a neural network. If we properly define the state  $s$  of the MDP, we can directly use the fully connected layer to approximate the Q-function well. There are no constraints for the network architecture, but the output dimensions will be fixed as the dimensions of the number of the actions and each value of the output represents the Q-value  $Q(s, a_i)$ .

In [84], Mnih *et al.* introduced the well-known architecture Deep Q-Networks (DQN), and it is a deep convolutional network trained with a variant of Q-Learning. Instead of using the user-defined discrete state space, the deep convolutional network now enables the network to train on the high dimensional raw image data. Roughly speaking, DQN extracts the internal state from the input image data by convolutional layers and

uses the function approximation to find the  $Q(s, a)$ . The benefit of function approximation method is that the algorithm can be learned with continuous state. In addition, because of the neural network, they utilize the stochastic gradient descent to update the weight of the network. In order to alleviate the problem of learning with correlated data and non-stationary distribution, the experience replay mechanism [85] is used. This mechanism  $D_{exp}$  will store most recent  $N$  transitions of state  $s_i$ , action  $a_i$ , immediate reward  $r_i$ , and next state  $s_{i+1}$ , and randomly sample a mini-batch  $D_{train}$  from  $D_{exp}$  to smooth the distribution over many past behaviors.

$$e_i = \{s_i, a_i, r_i, s_{i+1}\}, i \in [1, N]$$

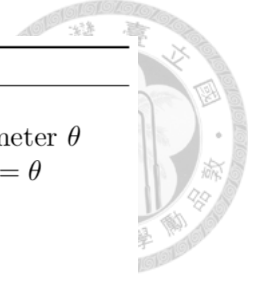
$$D_{exp} = \{e_1, e_2, \dots, e_N\} \quad (2. 10)$$

Without the experience replay, the learning process is easily unstable and the reward plotting through time is oscillating. Thus, the experience replay is needed for the Deep Q-Learning. Moreover, there are two networks for avoiding too dramatic change while training, and named as main network and target network. In the learning process, we will keep updating the main network with parameters  $\theta$  by sampling the experience from the experience replay, and utilize the target network with parameters  $\theta'$  to choose the action based on the current state  $s$ . After some training time steps, we will replace the target network parameters  $\theta'$  with the main network parameters  $\theta$ . The loss function for updating the network will be defined as following in equation (2. 11):

$$y_i = \begin{cases} r_i & , \text{if } s_{i+1} \text{ is not terminal state} \\ r_i + \gamma \max_{a'} Q(s_{i+1}, a' | \theta_i') & , \text{otherwise} \end{cases} \quad (2. 11)$$

$$L_i(\theta_i) = \mathbb{E}_{(s_i, a_i, r_i, s_{i+1}) \in D_{train}} [(y_i - Q(s_i, a_i | \theta_i))^2]$$

where  $y_i$  is the target value in the loss function  $L_i(\theta)$ , and the network will be updated with gradient descent step on the loss function as in the equation (2. 12).




---

**Algorithm 2** Deep Q-Learning
 

---

Initialize the replay memory  $D_{exp}$   
 Initialize the weight of the main Q network with arbitrary parameter  $\theta$   
 Initialize the weight of the target Q network with parameter  $\theta' := \theta$   
 Learning rate:  $\alpha \in [0, 1]$   
 Discounting factor:  $\gamma \in [0, 1]$   
 Epsilon value:  $\epsilon \in [0, 1]$

- 1: *loop*:
- 2: **for** episode = 1 to M **do**
- 3:   Receive the initial observation state  $s_1$
- 4:   **for** t = 2 to T and s is not terminal **do**
- 5:     Manipulate the  $\pi$  based on  $Q$  with the  $\epsilon - greedy$  strategy
- 6:     **if**  $\epsilon \geq 0.05$  **then**
- 7:        $\pi(s_t) \leftarrow$  random action
- 8:     **else**
- 9:        $\pi(s_t) \leftarrow \arg \max_a Q(s_t, a_t | \theta)$
- 10:      $a_t \leftarrow \pi(s_t)$
- 11:      $r_t \leftarrow R(s_t, a_t)$
- 12:      $s_{t+1} \leftarrow$  be observed after doing the new action
- 13:     Store the tuple of  $(s_t, a_t, r_t, s_{t+1})$  in the experience replay  $D_{exp}$
- 14:     Sample a random mini-batch  $(s_i, a_i, r_i, s_{i+1})$  of transitions from  $D_{exp}$
- 15:
- 16:     
$$y_i = \begin{cases} r_i & \text{if } s_{i+1} \text{ is a terminal state} \\ r_i + \gamma \max_{a'} Q(s_{i+1}, a' | \theta') & \text{otherwise} \end{cases}$$
- 17:
- 18:     Perform the gradient descent step on  $(y_i - Q(s_i, a_i | \theta))^2$  with respect to  $\theta$
- 19:     Replace the parameter  $\theta'$  in target network with the parameter  $\theta$  in the main network every C time steps.
- 20:      $s_t \leftarrow s_{t+1}$
- 21: **return** Q

---

Fig. 2. 4 Deep Q-Learning algorithm

$$\begin{aligned}
 \theta_{i+1} &= \theta_i + \alpha \nabla_{\theta_i} L_i(\theta_i) \\
 \nabla_{\theta_i} L_i(\theta_i) &= \mathbb{E}_{(s_i, a_i, r_i, s_{i+1}) \in D_{train}} [(y_i - Q(s_i, a_i | \theta_i)) \cdot \nabla_{\theta_i} Q(s_i, a_i | \theta_i)] \quad (2.12)
 \end{aligned}$$

Finally, the complete algorithm of deep Q-Learning is shown in Fig. 2. 4.

Hasselt *et al.* [86] proposed the double Q-Learning based on the Deep Q-Learning. Due to the known overestimated error of Q-Learning which is because of the error of function approximation, they introduced to use the greedy policy according to the online network which is main network and use the target network to estimate its value for reducing the overestimation by decomposing the max operation into action selection and

evaluation. Hence, the target value  $y_i$  will be revised as following in equation (2. 13):

$$y_i = \begin{cases} r_i & , \text{if } s_{i+1} \text{ is not terminal state} \\ r_i + \gamma Q(s_{i+1}, \arg \max_{a'} Q(s_{i+1}, a' | \theta_i) | \theta_i') & , \text{otherwise} \end{cases} \quad (2. 13)$$

$$L_i(\theta_i) = \mathbb{E}_{(s_i, a_i, r_i, s_{i+1}) \in D_{train}} [(y_i - Q(s_i, a_i | \theta_i))^2]$$

Wang *et al.* [87] presented the dueling network representing two estimators that are state-value function and state-dependent action advantage function. The architecture in [87] is shown in Fig. 2. 5. Also, the algorithm is based on the double Q-Learning.

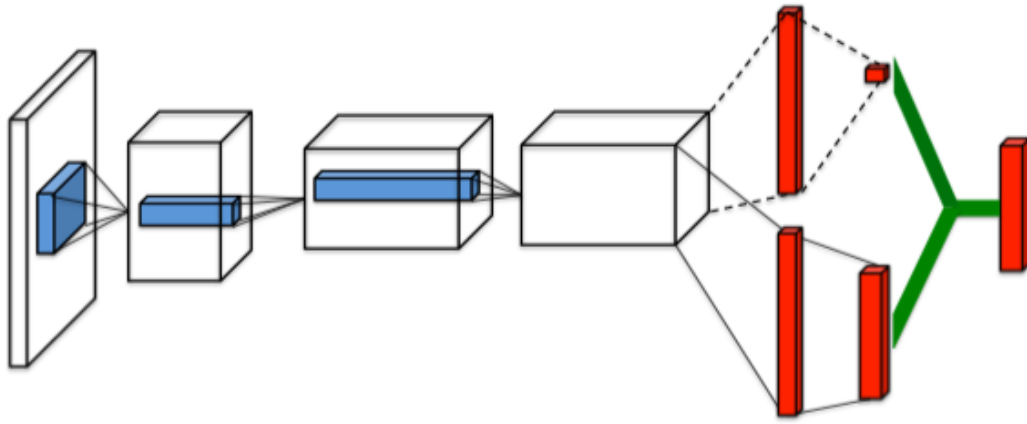


Fig. 2. 5 The architecture of dueling network. The upper one-dimension red block before the green process is the state-value function, and the lower red block is the action advantage function with the dimension being the number of actions. The green process is defined as in the equation (2. 14).

$$Q(s, a; \theta_i, \alpha, \beta) = V(s; \theta_i, \beta) + (A(s, a; \theta_i, \alpha) - \frac{1}{|\mathcal{A}|} \sum_{a'} A(s, a'; \theta_i, \alpha)) \quad (2. 14)$$

$$y_i = \begin{cases} r_i & , \text{if } s_{i+1} \text{ is not terminal state} \\ r_i + \gamma Q(s_{i+1}, \arg \max_{a'} Q(s_{i+1}, a' | \theta_i, \alpha, \beta) | \theta_i', \alpha', \beta') & , \text{otherwise} \end{cases}$$

where  $\alpha, \beta$ : parameters for action advantage and state value in the main network,  
 $\alpha', \beta'$ : parameters for action advantage and state value in the target network



## 2.2 Convolutional Neural Network

Owing to the development of graphics processing unit (GPU) and the large amount of released datasets, deep neural network has been very popular in many research fields recently. In particular, Convolutional neural network (CNN), a kind of feed-forward artificial neural network, is inspired by the connectivity patterns between neurons of the animal visual cortex and is most commonly to solve the image recognition tasks. An example of such CNN architecture is shown in Fig. 2. 6, and it is a well-known net named as ZF Net [88]. A simple CNN architecture is composed of convolutional layers, pooling layers, and fully connect layers, which are introduced in the following subsections.

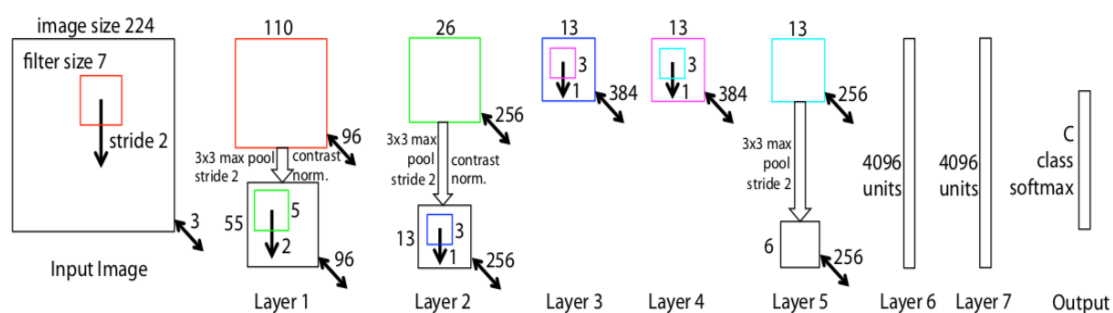
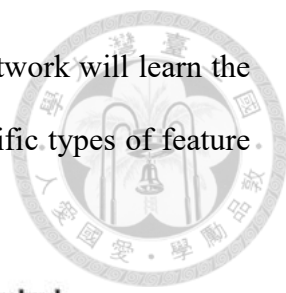


Fig. 2. 6 The architecture of ZF Net.

### 2.2.1 Convolutional Layer

Unlike the standard neural network like multilayered perceptron, the convolutional layers have neurons in 3 dimensions, namely, width, height, and depth. The parameters in the convolutional layer are composed of a set of learnable filters. For example, in Fig. 2. 6, layer 1 to 5 are all convolutional layers, and the filter size at the first layer is  $7 \times 7 \times 3$  with width and height being 7 and the depth of image is 3. During the forward pass, in each convolutional layer, each filter will slide on the input volume along width and height direction with the stride 2 and also do the dot product operation with the covered input volume, which produce a 2-dimensional activation map, namely, feature map. We





show an example of the convolution process in Fig. 2. 7, and the network will learn the parameter of the filters for having the activations toward some specific types of feature in the image, such as the edge in image.

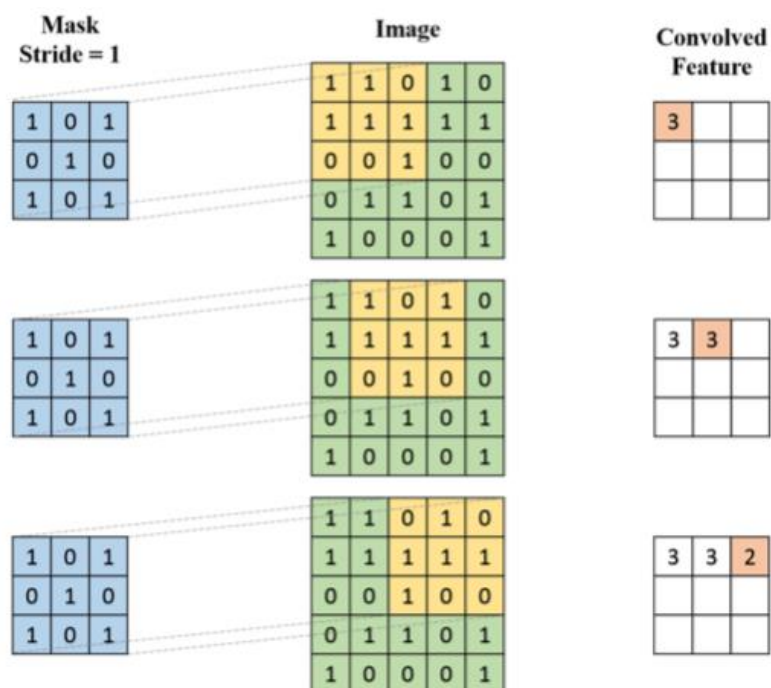


Fig. 2. 7 The example of the process of the convolution. In the first row, the filter in blue will do the dot product with the image map in the yellow color. Therefore, after the dot product operation, the output the value is 3, and the same process for the second and third row.

## 2.2.2 Pooling Layer

The most common pooling method is max-pooling which pools out the maximum value from the values covered by the mask. The function of the pooling layer is to reduce the parameters progressively and keep the important features in the feature map. In addition, it helps to avoid the overfitting of the training, but we will lose too many features if there are too many pooling layers or too large stride. In the ZF Net, there are some max-pooling layer which is inserted in layer 1, 2, and 5.



Fig. 2. 8 The example of how the max-pooling with  $2 \times 2$  mask and the stride 2.

### 2.2.3 Fully Connected Layer

It is just like the standard neural network, such as the multilayered perceptron which each layer is fully connected to each other. In the ZF Net, the layer 6, 7, and output are all fully connected layers.

### 2.2.4 VGG-16 Net

Simonyan *et al.* [89] proposed the VGG-16 net which shows that the deeper depth of the network can highly improve the performance of the object recognition. The architecture is shown in Fig. 2. 9, and it has 16 convolutional and fully connected layers. It is a homogeneous architecture, with only  $3 \times 3$  filters along with  $2 \times 2$  max-pooling layers, which reduced the needed number of parameters. VGG-16 Net works well in both image classification and detection, and it has been utilized as a pre-trained model for many works.

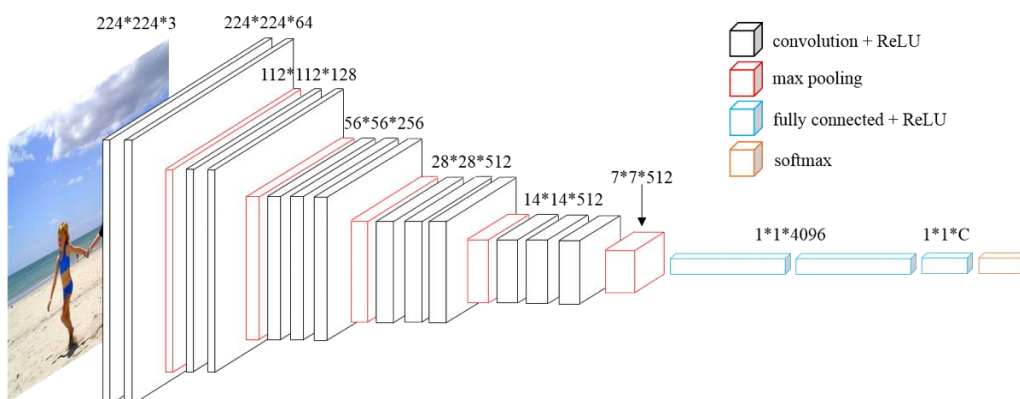
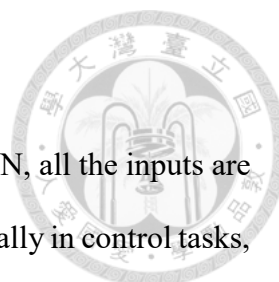


Fig. 2. 9 The architecture of VGG-16 net. C is the number of the classes.



## 2.3 Recurrent Neural Network

In a standard neural network like multilayered perceptron or CNN, all the inputs are assumed to be independent of each other. However, many tasks especially in control tasks, the sequential information is very important for getting a better performance. The idea behind recurrent neural network (RNN) is to make use of every element in the sequence, and perform the same operation on every element  $x_t$  and the state information  $s_{t-1}$  from pervious computation to produce the output. In other words, RNN has the function like memory, which computes the information what has been calculated so far.

We show the process of the RNN in Fig. 2. 10, and unfold the process. The element will be inputted in sequence, and do the linear combination with the previous state information to pass through the activation function and get the new state  $s_t$  like in the equation (2. 15). After weighting the  $s_t$  and passing through the softmax function, we can update the parameter by cross-entropy computed on predicted  $o_t$  and target  $y_t$ .

$$s_t = \sigma(Ws_{t-1} + Ux_t)$$

$$o_t = softmax(Vs_t)$$

$\sigma(\cdot)$ : activation function, ex:  $\tanh, ReLU$  (2. 15)

All model weight parameters:  $\theta = \{U, V, W\}$

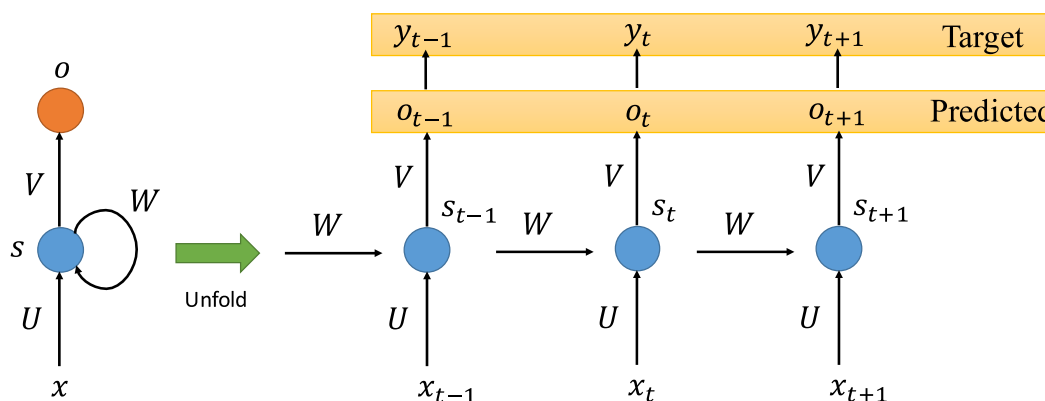
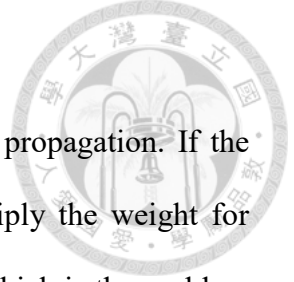


Fig. 2. 10 The operation process and unfold process of the RNN



### 2.3.1 Long Short-Term Memory

While updating the RNN, we utilize the error to do the back propagation. If the weight is smaller than 1, the error of back propagation will multiply the weight for multiple times, and get an extremely error close to 0 for updating, which is the problem of gradient vanishing. In contrast, if the weight is bigger than 1, it will get an extremely big error, which is gradient exploding. They are the reasons that it is difficult for RNN to operate on a long sequence.

Hochreiter *et al.* [90] proposed the Long Short-Term Memory (LSTM) to solve the problem of RNN. Compared the LSTM with RNN, LSTM get three more gates for controlling the information, such as input gate  $z^i$ , output gate  $z^o$ , and forget gate  $z^f$ .

$$\begin{aligned}
 z^f &= \sigma(W^f \langle x_t, s_{t-1} \rangle), \quad z^i = \sigma(W^i \langle x_t, s_{t-1} \rangle) \\
 z^o &= \sigma(W^o \langle x_t, s_{t-1} \rangle), \quad z = \tanh(W \langle x_t, s_{t-1} \rangle) \\
 c_t &= z^f \odot c_{t-1} + z^i \odot z, \quad s_t = z^o \odot \tanh(c_t) \\
 o_t &= \sigma(V s_t)
 \end{aligned} \tag{2.16}$$

$\sigma(\cdot)$ : activation function, sigmoid function

$\langle \cdot \rangle$ : concatenation,  $\odot$ : matrix multiplication

All model weight parameters:  $\theta = \{V, W, W^i, W^o, W^f\}$

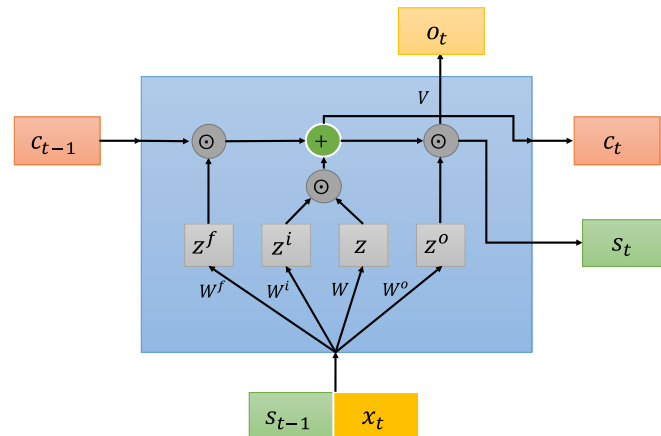


Fig. 2. 11 The mechanism of LSTM and the operation process will be shown in (2. 16).

The mechanism of LSTM is shown in Fig. 2. 11. The input gate controls the input information, and utilize the forget gate to control the usage of previous calculation in memory and the input information. The output gate decides the output of the hidden state to pass. Those gates helps to alleviate the problem of forgetting the beginning information in the long sequence.

### 2.3.2 Gated Recurrent Unit

In order to reduce the needed parameters and improve the convergence speed for training, Cho *et al.* [91] proposed Gated Recurrent Unit (GRU), which replaces the input gate and the forget gate with the reset gate.

$$\begin{aligned}
 r &= \sigma(W^r \langle x_t, s_{t-1} \rangle), \quad z = \sigma(W^u \langle x_t, s_{t-1} \rangle) \\
 s' &= \tanh(W' \langle x_t, r \odot s_{t-1} \rangle), \\
 s_t &= z \odot s_{t-1} + (1 - z) \odot s', \quad o_t = \sigma(V s_t)
 \end{aligned}
 \tag{2. 17}$$

$\sigma(\cdot)$ : activation function, sigmoid function

$\langle \cdot \rangle$ : concatenation,  $\odot$ : matrix multiplication

All model weight parameters:  $\theta = \{V, W', W^r, W^u\}$

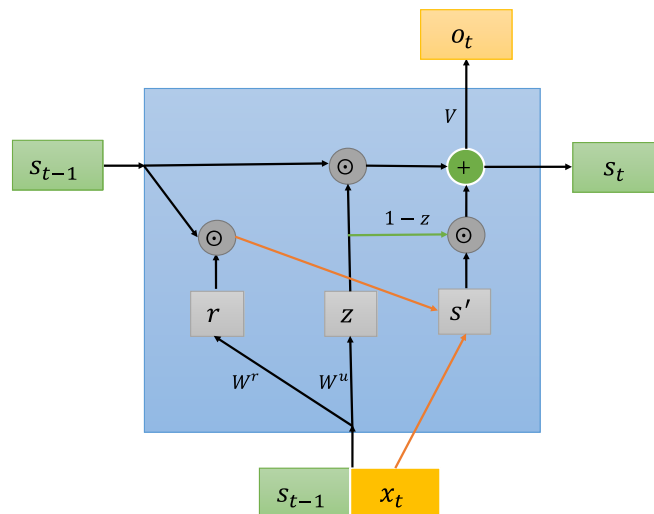
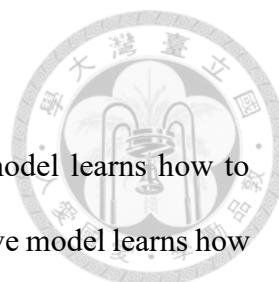


Fig. 2. 12 The mechanism of GRU and the operation process is shown in the (2. 17).



## 2.4 Generative Model

Having the input variable  $x$  and the label  $y$ , the generative model learns how to build the joint probability distribution  $p(x,y)$  while the discriminative model learns how to build the conditional probability distribution  $p(y|x)$  of the data directly.  $p(y|x)$  represents that given  $x$  and classify it to class  $y$ , and an example is shown below:

We have the data as  $(x,y) = (\text{price}, \text{pet}) : (50, \text{dog}), (50, \text{dog}), (20, \text{dog}), (20, \text{cat})$ .

Price \ Pet	dog	cat
	50	1
20	1/2	1/2

price \ Pet	dog	cat
	50	1/2
20	1/4	1/4

The generative model  $p(x,y)$  can also be transformed into  $p(y|x)$  by the Bayes rule, which is  $p(x,y) = p(y|x)p(x)$ . In addition, the  $p(x,y)$  takes advantage of generating the likely  $(x,y)$  pairs.

One of the important goals of AI is to understand how the world works, as Richard Feynman said, “What I could not create, I cannot understand.”, which is the basic idea behind the generative model. Although we need a lot of data for learning the joint distribution of data, the neural network uses few parameters to try to capture the essence of the data and generate it. In the subsections, we are going to introduce three deep-learning based generative model, such as auto-encoder, variational auto-encoder, and generative adversarial network.



### 2.4.1 Auto-Encoder

Auto-encoder is a type of feedforward neural network, and the main goal is to learn an efficient data encoding with dimension reduction in the unsupervised manner. The idea of auto-encoder is to encode the input data in domain  $\mathcal{X}$  to a lower dimensional latent representation  $z$  in latent space  $\mathcal{Z}$ , and transform the latent representation  $z$  back to the data in domain  $\mathcal{X}$ . Recently, the generative model has also been highly influenced by the idea of auto-encoder.  $E_x$  is the transition from domain  $\mathcal{X}$  to latent space  $\mathcal{Z}$ , and the  $G_z$  is the transition from latent space  $\mathcal{Z}$  to domain  $\mathcal{X}$ . The aim is to minimize the reconstruction loss as in the equation (2. 18), and the idea of it is shown in Fig. 2. 13.

$$\begin{aligned}
 E_x: \mathcal{X} &\rightarrow \mathcal{Z}, G_z: \mathcal{Z} \rightarrow \mathcal{X} \\
 \hat{\mathcal{X}}: &(G_z \circ E_x)\mathcal{X} \\
 E_x, G_z &= \underset{E_x, G_z}{\operatorname{argmin}} \|\mathcal{X} - \hat{\mathcal{X}}\|^2
 \end{aligned}
 \tag{2. 18}$$

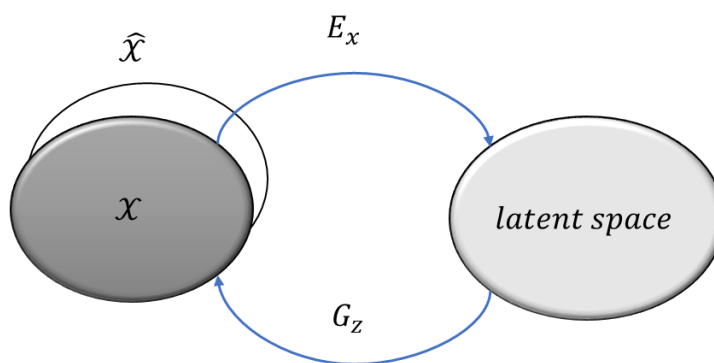


Fig. 2. 13 The idea of auto-encoder.

For example, the basic example for the generative model is the reconstruction of the MNIST dataset. The input data is the number image  $x$  and extract the low dimensional latent representation by passing through the encoder  $E_x$ , which is constructed by the neural network, such as multilayered perceptron or CNN. In order to reconstruct the image back, we will utilize the generator  $G_z$  which is also a neural network, and minimize a l2-norm loss function. We show the result as in the Fig. 2. 14.

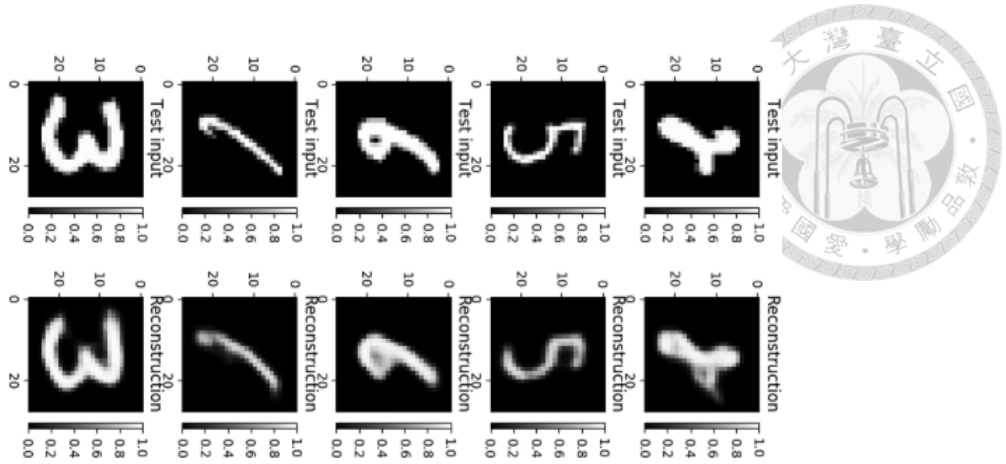


Fig. 2. 14 The example result of the auto-encoder. The upper row is the testing input image, and the lower row is the reconstruction result.

## 2.4.2 Variational Auto-Encoder

Based on the auto-encoder, Kingma *et al.* [92] proposed the variational auto-encoder (VAE), which learns the parameters of the probability distribution representing the data and sample from the distribution to generate the output. First, the encoder  $E_x$  will encode the input data  $x$  into the mean and variance of the probability distribution, and utilize the re-parameterization trick to make use of taking the stochastic gradient descent to update the encoder part. We show the derivation process in the (2. 19), and get the loss function  $L = -\log p_\theta(x^{(i)}) = -\mathbb{E}_z[\log p_\theta(x^{(i)}|z)] + \text{KL}(q_\phi(z|x^{(i)})||p_\theta(z))$ , which is the variational lower bound of the original loss function. The first term is the reconstruction loss, and the second term is to enforce the distribution of  $q_\phi(z|x)$  as close as the  $p_\theta(z)$ .

The idea behind the VAE is shown in the Fig. 2. 15

Encoder network  $E_x: q_\phi(z|x)$  with parameter  $\phi$

Generator network  $G_z: p_\theta(x|z)$  with parameter  $\theta$

re-parameterization trick:  $z = \mu_{z|x} + \Sigma_{z|x} \odot \epsilon$  (2. 19)

mean:  $\mu_{z|x}$ , variance:  $\Sigma_{z|x}$ , random noise:  $\epsilon \sim \mathcal{N}(0, I)$





The data likelihood:

$$\begin{aligned}
 \log p_{\theta}(x^{(i)}) &= \mathbb{E}_{z \sim q_{\phi}(z|x^{(i)})} [\log p_{\theta}(x^{(i)})] \\
 &= \mathbb{E}_z [\log \frac{p_{\theta}(x^{(i)}|z)p_{\theta}(z)}{p_{\theta}(z|x^{(i)})}] \quad (\text{by Bayes' Rule}) \\
 &= \mathbb{E}_z [\log \frac{p_{\theta}(x^{(i)}|z)p_{\theta}(z)}{p_{\theta}(z|x^{(i)})} \frac{q_{\phi}(z|x^{(i)})}{q_{\phi}(z|x^{(i)})}] \\
 &= \mathbb{E}_z [\log p_{\theta}(x^{(i)}|z)] - \mathbb{E}_z \left[ \log \frac{q_{\phi}(z|x^{(i)})}{p_{\theta}(z)} \right] + \mathbb{E}_z \left[ \log \frac{q_{\phi}(z|x^{(i)})}{p_{\theta}(z|x^{(i)})} \right] \\
 &= \mathbb{E}_z [\log p_{\theta}(x^{(i)}|z)] - \text{KL}(q_{\phi}(z|x^{(i)})||p_{\theta}(z)) + \\
 &\quad \text{KL}(q_{\phi}(z|x^{(i)})||p_{\theta}(z|x^{(i)})) \\
 &\geq \mathbb{E}_z [\log p_{\theta}(x^{(i)}|z)] - \text{KL}(q_{\phi}(z|x^{(i)})||p_{\theta}(z)) \\
 &\because \text{KL}(q_{\phi}(z|x^{(i)})||p_{\theta}(z|x^{(i)})) \geq 0 \text{ and } p_{\theta}(z|x^{(i)}) \text{ is intractable}
 \end{aligned}$$

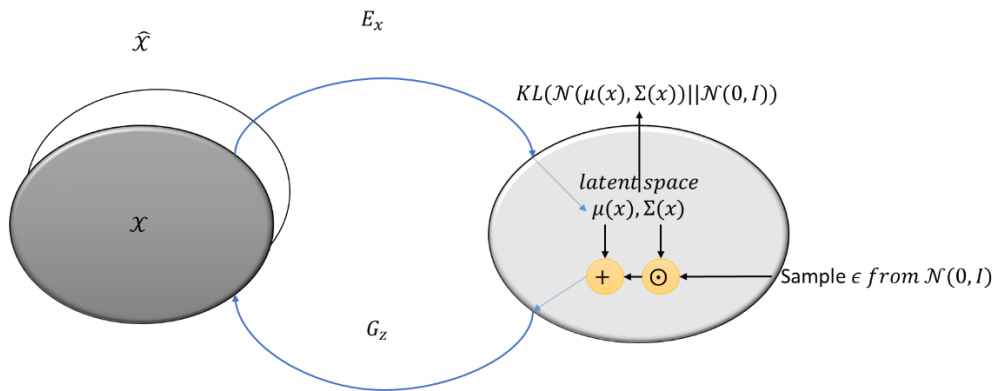


Fig. 2. 15 The idea of Variational Auto-Encoder (VAE)

### 2.4.3 Generative Adversarial Network

The generative adversarial network (GAN) is composed of two deep neural networks, which are the generator and the discriminator. Generator tries to generate the new data instance with the input random noise while the discriminator will evaluate the quality of the generated instance by classifying the generated one and the real data. The data flow

is shown in the Fig. 2. 16.

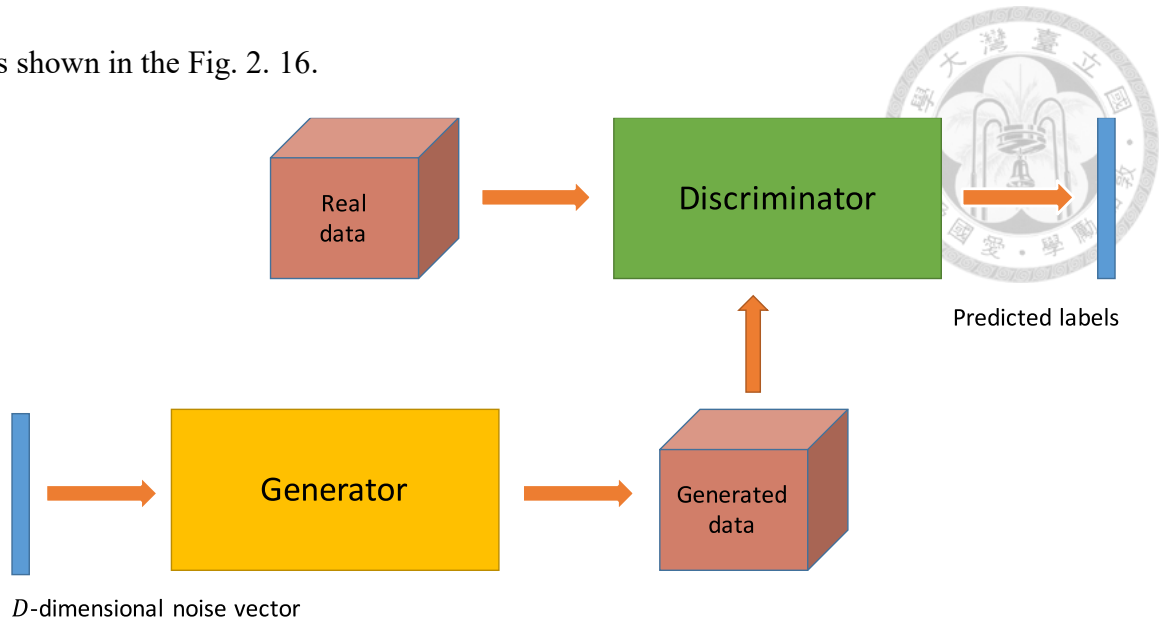


Fig. 2. 16 The data flow of the GAN.

The optimal generator  $G^*$  is defined in the equation (2. 20), and it can be seen as the discriminator  $D$  maximizing the function  $V(G, D)$  and the generator  $G$  minimizing the function  $V(G, D)$ . This is the reason that GAN has been named as min-max problem or adversarial problem.

$$G^* = \arg \min_G \max_D V(G, D)$$

$$V(G, D) = \mathbb{E}_{x \sim p_{data}} [\log D(x)] + \mathbb{E}_{\hat{x} \sim p_G} [\log D(\hat{x})]$$

*Generator G: a function with input  $z$  and output  $\hat{x}$*

*Discriminator D: a function with input  $x$  or  $\hat{x}$*  (2. 20)

*$x$ : instance from real data,  $\hat{x}$ : generated instance*

*$p_{data}$ : the probability distribution for real data*

*$p_G$ : the probability distribution for generated data*

However, in practice, we can only sample the data  $\{x^1, x^2, \dots, x^m\}$  from  $p_{data}$  and sample the generated data  $\{\hat{x}^1, \hat{x}^2, \dots, \hat{x}^m\}$  from  $p_G$ , therefore, we can re-write the equation into (2. 21). The actual loss function will become just like updating with the cross-entropy function, and the output of the  $D$  passes through the sigmoid function.



For *Generator G*:

Maximize the  $\hat{V} = \frac{1}{m} \sum_{i=1}^m (\log D(\hat{x}^i))$  with respect to parameters in  $G$ .

For *Discriminator D*:

Minimize the  $\hat{L} = -\frac{1}{m} \sum_{i=1}^m (\log D(x^i)) - \frac{1}{m} \sum_{i=1}^m \log(1 - D(\hat{x}^i))$  with respect to parameters in  $D$ .

Finally, the discriminator and the generator will take turn to update the parameter by the stochastic gradient method, and the algorithm is shown in the Fig. 2. 17. Although there are many varying version of GAN recently, the basic ideas of them are all the same. To our best knowledge, the main difference is that the different way to evaluate the distance between two distributions, which is the loss function forcing the generated data distribution close to real data distribution.

---

**Algorithm 3** Generative Adversarial Network

---

- The noise prior  $p_z$
- Initialize the weight  $\theta$  of the Generator  $G(z)$
- Initialize the weight  $\phi$  of the Discriminator  $D(x)$
- 1: *loop*:
- 2: **for** episode = 1 to M **do**
- 3:     **for** k steps **do**
- 4:         Sample mini-batch of data  $\{x^1, x^2, \dots, x^m\}$  from real data.
- 5:         Sample mini-batch of data  $\{\hat{x}^1, \hat{x}^2, \dots, \hat{x}^m\}$  from generated data which is generated by the  $G(z)$  with input noise sample from  $p_z$ .
- 6:         Update  $D(x)$  with respect to  $\phi$  by the stochastic gradient descent:

$$\hat{L} = -\frac{1}{m} \sum_{i=1}^m (\log D(x^i)) - \frac{1}{m} \sum_{i=1}^m (\log(1 - D(\hat{x}^i)))$$

- 7:         Sample mini-batch of data  $\{\hat{x}^1, \hat{x}^2, \dots, \hat{x}^m\}$  from generated data which is generated by the  $G(z)$  with input noise sample from  $p_z$ .
- 8:         Update  $G(z)$  with respect to  $\theta$  by the stochastic gradient ascent:

$$\hat{V} = \frac{1}{m} \sum_{i=1}^m (\log D(\hat{x}^i))$$


---

Fig. 2. 17 The algorithm of the Generative Adversarial Network.

# Chapter 3 Methodology



## 3.1 System Overview

The proposed autonomous system of the robot is shown in Fig. 3. 1. The autonomous system interacts with the environment, and the robot will get feedbacks and the sensing information back. This work is inspired by [43][44][45], and we model the internal needs of robot as the drives which generates the intensity of motivations with the stimulus sensed by the robot. Since the homeostasis requires to balance the drives, a decision making system is utilized to bound the motives. Motive is the state of the decision making system, which decides what actions the robot should take. After the chosen action is executed, we can get the feedback from the user to tune the preference of the user. What's more, the mechanism of moods is build up by the value of drives and the stimulus, and it will influence the style of chatting.

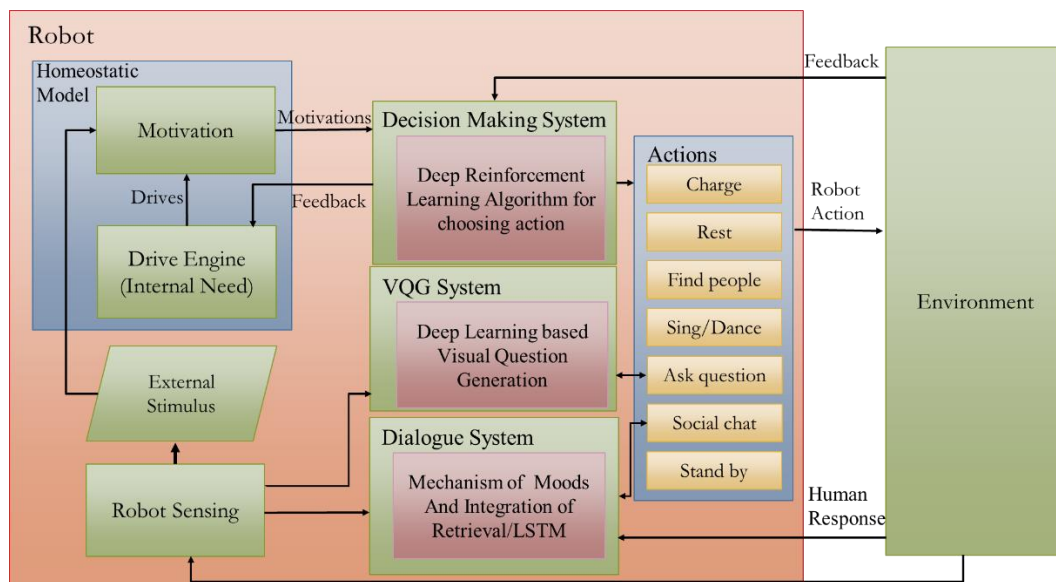


Fig. 3. 1 The proposed autonomous system.

In the decision making part, deep reinforcement learning algorithm is applied to choose the robot actions, such as charging, resting, finding people, singing or dancing, asking question, chatting, and standing by. We show the flow of the decision making in

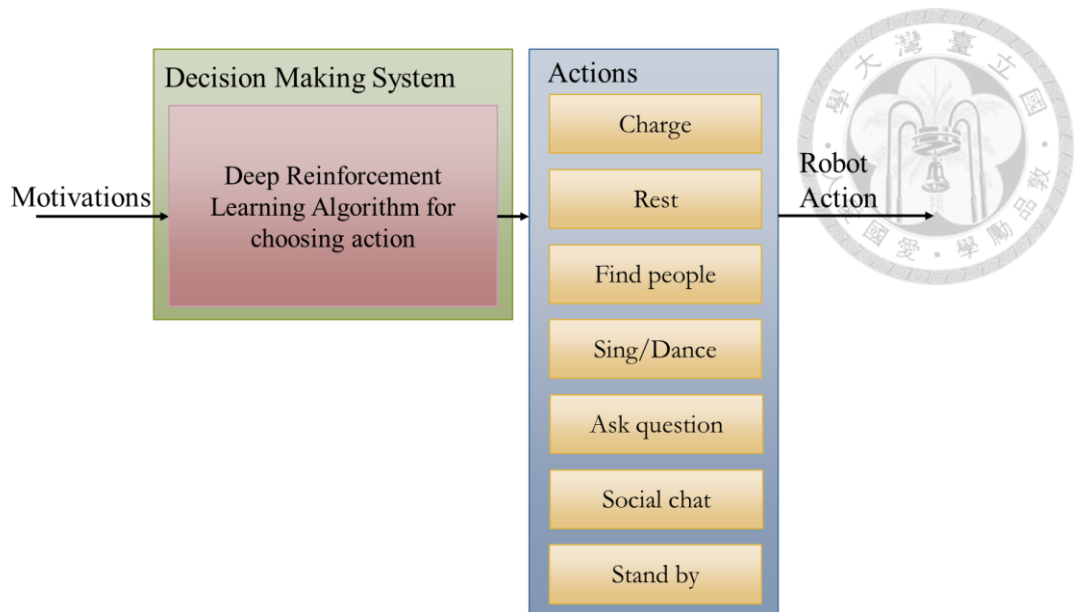


Fig. 3. 2 The decision making system in the autonomous system.

Fig. 3. 2. Another main focused point in this work is the module of asking question. For satisfying the need of curiosity, we utilize the deep learning based Visual Question Generation (VQG) for asking question, and combine the variational auto-encoder with gumbel softmax and adversarial training to make the VQG generate multiple different questions. In addition, the dialogue system of social chat is also made for a more interesting interaction between robot and human, and the system flow is shown in Fig. 3.

3. First, what the human says will be transferred into text by speech-to-text module, and an interactive dialogue management module is going to manage the system how it should respond to the user by QA Bot or by Seq2Seq chit-chat module. As we mentioned previously, the mechanism of moods is a gate for controlling the response being positive or negative style, and the text style translation module will transfer the style of the sentence from the chit-chat module. Eventually, the generated sentence from QA Bot or from Text style translation will be turned into speech to be spoken out by the robot through the text-to-speech module.

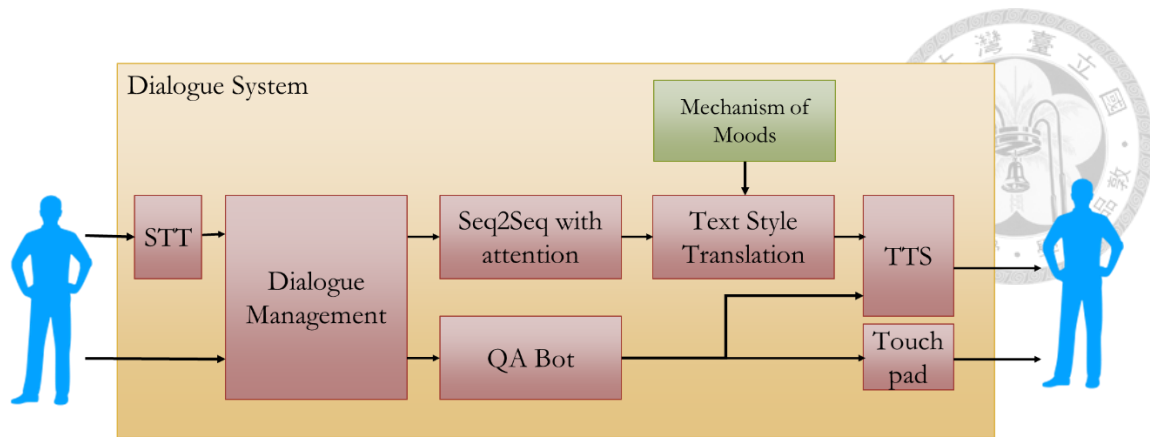


Fig. 3. 3 The dialogue system of social chat in the robot actions.

In the rest of this chapter, the Autonomous system will be described in Section 3.2 firstly. Then, we will describe how the dialogue system and the deep learning based text style translation module work in Section 3.3.1. Finally, the deep learning based visual question generation is mentioned in the last Section 3.4.

## 3.2 Autonomous System

In order to make the robot be autonomous at home or nursing home, the autonomous system of the robot is needed. The homeostasis is adopted to model the robot, and keep the robot stay in a steady state, which is satisfying the unsatisfied internal needs of the robot by taking the actions chosen by the decision making system. Similar to the homeostatic drive theory, the drive will decrease after the action is accomplished. The operational flow within the homeostasis is shown in Fig. 3. 4. Moreover, inspired by the Maslow's hierarchy of needs [46], we extend the internal needs of the robot and make it behave in a more human manner, and build the mechanism of the moods.

Therefore, we will introduce first how the drive is defined in this work in Section 3.2.1, and the definition of the stimulus is mentioned in Section 3.2.2. Then, the operation of how to generate the motivation with the drive and stimulus is presented in Section 3.2.3. After having motivation of robot, the robot will take the action which is decided by

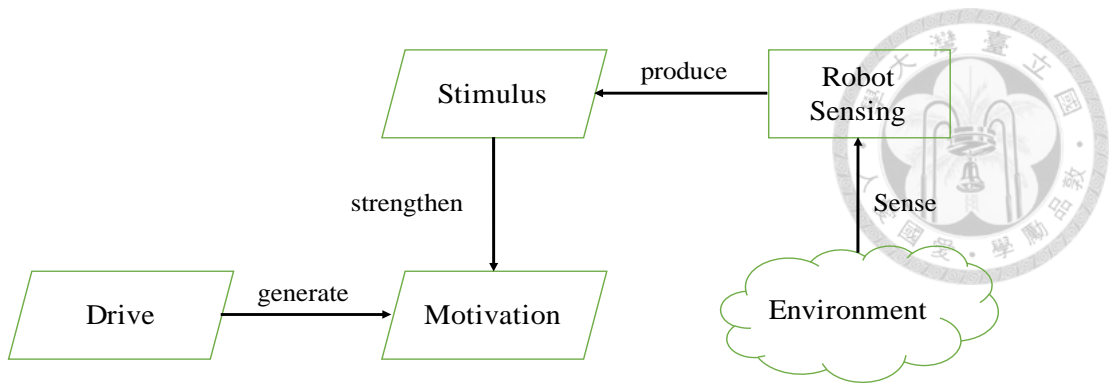


Fig. 3. 4 The operational flow within Homeostasis.

decision making system, which will be described in Section 3.2.4. Next, how the mechanism of moods is designed for our robot in Section 3.2.5. At last, the deep reinforcement learning based decision making system is explained in Section 3.2.6.

### 3.2.1 Drive

In this works, the internal needs of the robot are modeled as drives. Inspired by Maslow’s hierarchy of needs, we re-define drives in our work for making the robot act like a human as in Table. 3. 1. In Maslow’s hierarchy of needs [46], Maslow extended the homeostatic principle of physiology need to needs, such as safety, love and belonging, esteem, and self-actualization, as shown in Fig. 3. 5. On our research work, we define 7 kinds of need, namely, Need of Energy (NEner), Need of Safety and Rest (NSaR), Need of Belonging (NBel), Need of Esteem (NEst), Need of Self-Actualization (NSA), and Need of Curiosity (NCur). NEner represents that robot relies on energy to maintain working condition. Because the robot will have some safety issue and overheat issue due to robot motions, NSaR is needed as well by the robot. Just as human needs the company of friends or family, therefore, NBel means the need of the robot for its belonging to a society with human friend. Notice that NEst is the need for the sense of the achievement, which means to offer the services like singing or dancing, because providing service to

Table. 3. 1 The difference of drives compared with Maslow's hierarchy of needs.

Maslow's hierarchy of needs	Ours
Need of physiology	Need of Energy (NEner)
Need of safety	Need of Safety and Rest (NSaR)
Need of love and belonging	Need of Belonging (NBel)
Need of esteem	Need of Esteem (NEst)
Need of self-actualization	Need of Self-Actualization (NSA)
	Need of Curiosity (NCur)

humans is considered as a main functions of the social companion robot. Also, as a

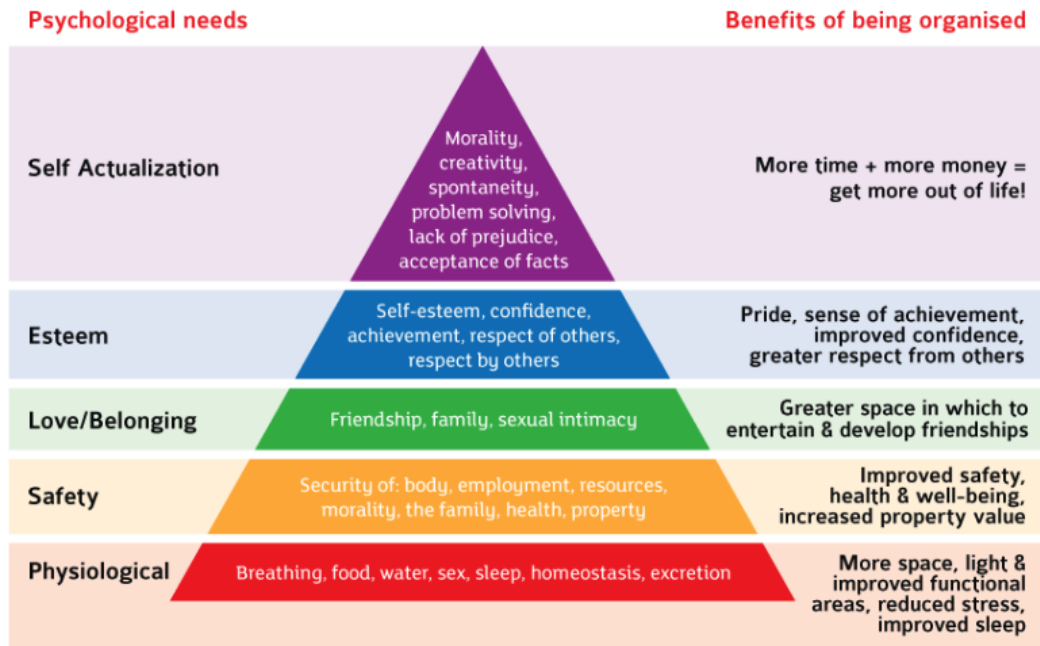


Fig. 3. 5 Maslow's hierarchy of needs.

companion robot, one crucially important skill is to chat, which refers to the need, NSA. Lastly, behaving like a human, curiosity is a very important characteristic, so NCur is defined as such need of the robot.

Drives are internal parameter of the robot in the form of real numbers, and the value of it indicates the degree of dissatisfaction of the certain need. The set of the drives used in the system is denoted as  $D$ . Each drive  $d^i \in D$  is normalized to the range  $[0,1]$ , and



smaller value represents that the need is more satisfied. In contrast, the larger value means the need is more severe. The value of the drive will get larger incrementally through the time as shown in equation (3. 1), and the specific action  $a^i$  being done will decrease the corresponding drive  $d^i$  as defined in equation (3. 2). Such action will continue for  $\gamma$  time steps as shown below.

$$d_{t+1}^i = d_t^i + \delta^i \quad (3. 1)$$

$d_{t+1}^i$ : the value of drive at  $t + 1$  time step  
 $d_t^i$ : the value of drive at  $t$  time step  
 $\delta^i$ : the incremental value of the certain drive

$$d_{t+\gamma}^i = d_t^i - f_a(d_t^i, a^i) \quad (3. 2)$$

$d_{t+\gamma}^i$ : the value of drive at  $t + \gamma$  time step  
 $d_t^i$ : the value of drive at  $t$  time step  
 $a^i$ : the action has been done  
 $f_a(d_t^i, a^i)$ : the decreased value of drive  $d_t^i$  based on the specific action  $a^i$  which has executed for  $\gamma$  time steps

### 3.2.2 Stimulus and Environment

The factors which can affect the intensity of motivations including the drive and the stimulus are twofold. The stimuli are defined as the external conditions that are sensed by the sensors of the robot. The set of stimuli is denoted as  $ST$ , and each stimulus  $st^i \in ST$  is defined as 0 or 1 as in the equation (3. 3).

$$st^i = \begin{cases} 1, & \text{if the sensor is activated} \\ 0, & \text{otherwise} \end{cases} \quad (3. 3)$$

There are three kinds of stimuli, such as Stimulus of Overheat (SOVer), Stimulus of Human (SHum), and Stimulus of Encouragement (SEnc), which are described in Table. 3. 2. SOVer is designed to handle the situation of the overheat of the motors in each joint, in result of having the movement frequently increasing the raising of temperature. It can

motivate the robot to know when to take a rest for cooling down the motors. SHum means that whether there is a human in front of the robot or not, whereby the robot will

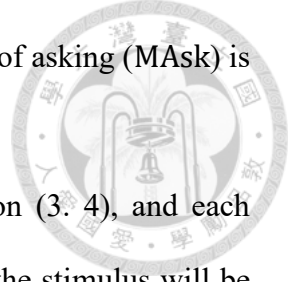
Table. 3. 2 The stimuli can be sensed by the sensors of the robot and used in this work.

Stimulus of Overheat (SOver)	Stimulus of Human (SHum)	Stimulus of Encouragement (SEnc)
---------------------------------	-----------------------------	-------------------------------------

know if it needs to interact with the human currently and also affect the belonging to human for robot. SEnc represents that whether the human give some encouragements to the robot when the robot interacts with the human. While the robot gets the encouragement from the human, the satisfaction of the certain need will continue for a while, which is similar like the feeling of human.

### 3.2.3 Motivation

Motivations can be considered as the goals for compensating the internal needs of the robot. The intensity of Motivations is generated by the value of drives and the external stimuli. The set of the Motivations is denoted as  $M$ , and each motivation is denoted as  $m^i$ . The relation between drives and motivations is one-to-one, which means that for each drive in the system, the corresponding motivations exists as shown in Table. 3. 3. Based on the NEner, the robot has the motivation of survival (MSur) if the energy of the battery is too low. For the NSaR, the robot should have the motivation to take a rest (MRes) for preventing overheat of the motor and being still or stuck. In order to satisfy NBel, robot should be motivated to get close to the human for having better relationship (MRel). Motivation of achievement (MAch) leads the robot to satisfy NEst, i.e., to drive the robot to offer and accomplish certain services to humans while meeting them. Being a social companion robot, the robot has the motivation to have social interaction with the human



(MSI) in order to meet NSA. For behaving like a human, motivation of asking (MA<sub>SK</sub>) is a basic human's personality trait driving the robot to meet NCur.

The intensity of motivation is calculated as shown in equation (3.4), and each intensity can be activated by several stimuli. The affected value of the stimulus will be derived by the function  $f_{st}(\cdot)$ , which will be described in detail in the experiment part.

$$m_t^i = d_t^i + \sum_{st^j \in ST} f_{st}(st^j, m_t^i) \quad (3.4)$$

- $m_t^i$ : the intensity of motivation at  $t$  time step.
- $d_t^i$ : the value of the drive at  $t$  time step.
- $st^j$ : each stimulus which in the set of  $ST$
- $f_{st}(\cdot)$ : the affected value determined by the certain  $st^j$  and  $m_t^i$ .

Table. 3.3 The motivations which is correlated to each drive in the system.

Drives	Motivations
Need of Energy (NEner)	Motivation of Survival (MSur)
Need of Safety and Rest (NSaR)	Motivation of Rest (MRes)
Need of Belonging (NBel)	Motivation of Relationship (MRel)
Need of Esteem (NEst)	Motivation of Achievement (MAch)
Need of self-actualization (NSA)	Motivation of Social Interaction (MSI)
Need of Curiosity (NCur)	Motivation of Asking (MA <sub>SK</sub> )

### 3.2.4 Action

The goal of the proposed autonomous system is to choose an action out of the pre-defined action set to cope with the current situation. The purpose of an action is to satisfy the internal needs that is severe through the execution of it. The target of the action could be the human or the robot itself. For example, chatting can be applied to the human, while the resting is only for the robot to cool its motors. As defined in Fig. 3.2, there are

charging, resting, finding people, singing or dancing, asking question, chatting, and standing by. The set of the actions is denoted as  $A$ , and  $a^i$  represents each action. In addition, the execution of the action will continue for a while, and it will also decrease the value of the drive like what we have described previously. In the following, we will introduce how each action function works:

- **Charging**

- Utilize the text to speech module to speak out: “Can someone help me to charge?”. After someone helps to charge the robot, the charging will last for a while until the battery reaches the desired level. Then, the robot will speak out: “Can someone helps me to unplug the charger?”

- **Resting**

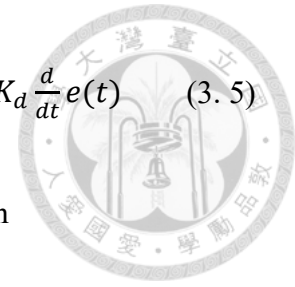
- Utilize the text to speech module to speak out: “I need to take a rest”, and turn the robot into resting posture. After resting for a while, the robot will go back to standing posture.

- **Finding people**

- Utilize the YOLO-v3 to find the human in the field of view of the camera mounted on robot, and the PID controller is activated to move the robot so that the bounding box of the found human image is centered in the entire image. Then, the robot moves toward to the human until the bounding box size reaches a predefined threshold value. Finally, use the face detection API which is built for the robot to make sure that the robot is standing in front of the human. For controlling the angular velocity and linear velocity of the robot, there are two PID controllers to handle about it. The error term in the equation (3. 5) is the error between center position of bounding box and the center of the entire image for the first controller or that between the bounding box size and the desired

$$\text{PID controller: } u(t) = K_p e(t) + K_i \int_0^t e(\tau) d\tau + K_d \frac{d}{dt} e(t) \quad (3.5)$$

$u(t)$ : the controlled output,  $e(t)$ : the current error  
 $K_p, K_i, K_d$ : the proportional, integral, derivative gain



bounding box size for the second controller, and the controlled output will be the angular velocity and the linear velocity of the robot correspondingly.

- **Singing or dancing**
  - The handcrafted movement and arm gestures at every moment have been designed, and the script and the music behind will be played at the same time while singing and dancing are being performed.
- **Asking question**
  - The interaction through talking about the surrounding scene or the objects can more easily touch the real feeling of human, and therefore we propose a visual question generation module to handle it, which will be introduced in Section 3.4. For example, when the robot sees a group of friends celebrating the birthday with a cake, it will ask: “Is the cake delicious?” or “What are you celebrating?”
- **Chatting**
  - A dialogue system is designed for chit-chat and question-answering, and it will be described in Section 3.3. The basic conversational ability is set up for the social companion purpose.
- **Standing by**
  - When the robot has no intention for interaction, charging, or resting, it can choose “standing by” to wait for human’s wake-up or interaction. In other words, the robot does not have to satisfy any need during “standing by” period.



### 3.2.5 Mechanism for Moods

In order to make the human-robot interaction more natural and interesting, the robot needs to behave more like a human, which motivates us to design the mechanism for robot to generate moods, such as positive and negative moods. This mechanism mainly affects the style of the generated sentence for chit-chat. The reason for having only two moods is that there are few datasets with emotional labels. For those existing sentence datasets, most of them are noisy or are collected from Twitter or Facebook, of which the words are seldom spoken in regular chatting way.

In this work, there are three main factors for influencing the moods of the robot, and they are the value of the drives, the emotion of the human in interaction, and the encouragement from the human. Note that an unsatisfied robot's need will make the robot become more negative, which is just similar to the human case where one feels anxious and negative when he/she suffers a severe need, like, hunger. In contrast, while the needs are satisfied, the robot act in a more positive way. On the contrary, if the human has negative emotion, the robot can speak in a more positive way to cheer him/her up. If the human has positive emotion, the robot however can speak in negative way to make fun of the human. On the other hand, the mood of the robot can also be affected if the human encourages the robot; i.e., if there is encouragement, the robot intends to act in a positive way. The detail of the mechanism is shown in the following via equation (3. 6). Although some related works[14][41] will design the mood so that it changes negatively while the same action command is issued by the human repeatedly, one of the biggest advantages of robot is believed to repetitively work as compared with human. As a result, the mechanism that we design in this work doesn't take into account the action repetition.

$$v_{mood} = \sum_i w_i \cdot d^i - \mu st_{encouragement} + \varepsilon \beta_{emotion} \quad (3.6)$$

$$Mood = \begin{cases} Positive, & \text{if } v_{mood} \leq \eta \\ Negative, & \text{if } v_{mood} \geq \eta \end{cases}$$

$\sum_i w_i = 1$ ,  $w_i$ : the weight parameter for each value of drive  
 $\mu, \varepsilon$ : the weighted factor of encouragement and emotion of human  
 $d^i$ : the value of drive,  $st_{encouragement}$ : the stimulus of encouragement,  $\beta_{emotion}$ : the observation of human's emotion  
 $\eta$ : the threshold of determining the mood of the robot



### 3.2.6 Decision Making System

In order to choose the action at every moment, a deep reinforcement learning based decision making system is applied. As we mention previously in the preliminary work in Chapter 2, we need to define the state, action, and reward. In addition, the reason to use the deep reinforcement learning is that we can model the continuous state instead of discrete state. It allows our decision making system to handle more complex information and a more sophisticated situation, and improve the performance of doing the right action.

- **State**

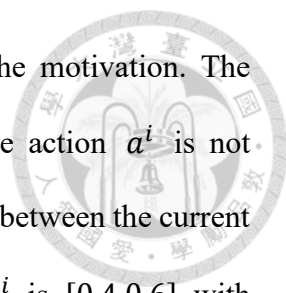
- The state in the decision making system is the intensity of the motivation  $m^i$ , and it is denoted as  $s$ , which is in the set of  $S$ . Note that  $s$  is an array with 6 dimension.

- **Action**

- The action is denoted as  $a$ , and the set of actions is denoted as  $A$ . The actions are the same as what we mentioned in Section 3.2.4.. Note that  $a$  is an array with 7 dimension.

- **Reward**

- Reward  $r$  is taken to encourage the right action at the right moment, and the goal here is to balance the value of the drives, which will be decreased after



execution of the action, and influence the intensity of the motivation. The reward will mainly be defined in two ways. First, if the action  $a^i$  is not standing by, the reward will be calculated by the difference between the current  $m^i$  and  $m_{desired}^i$ . For example, if the desired range of  $m^i$  is  $[0.4, 0.6]$  with  $m^i = 0.5$ , then you will get a positive reward  $\kappa \times (1 + 0.5 - 0.6) = 0.9\kappa$ , which is less than 1. Secondly, if the executed action is “standing by,” the value of  $m^i$  in the desired range will be calculated as  $\sum_i r^i$ . The reward will be determined by whether  $\sum_i r^i$  becomes greater than the threshold or not. For example, if the robot chooses the action “standing by” as in the previous example, it has higher possibility to get a higher reward. The detail of the reward function is shown in equation (3. 7). Moreover, the personality like whether the robot is more interested at interacting with the human can be designed by tuning the desired range of the motivation. For example, if the desired range is smaller, the intensity of the motivation will easily exceed the range. Then, the robot will do the specific action frequently to keep the motivation in the desired range. For adapting to different users’ preference, there will be an external reward to deal with the feedback from the users.





if the executed action  $a^i$  was not standing by, then:

$$r_{internal} = \begin{cases} \kappa \times (1 + m^i - m_{upper}^i), & \text{if } m^i \text{ in DR or } m_{prev}^i > m_{upper}^i \\ \lambda \times (1 - (m^i - m_{upper}^i)), & \text{otherwise} \end{cases} \quad (3.7)$$

if the executed action  $a^i$  was standing by, then:

$$r^i = \begin{cases} 1, & \text{if } m^i \text{ in desired range} \\ 0, & \text{if } m^i \text{ not in desired range} \end{cases}, \quad r_{internal} = \begin{cases} 1, & \text{if } \sum_i r^i > \nu \\ -1, & \text{otherwise} \end{cases}$$

$$r_{external} = \begin{cases} 1, & \text{positive feedback} \\ -1, & \text{negative feedback} \\ 0, & \text{otherwise} \end{cases}$$

$r_{internal}$ : internal reward,  $r_{external}$ : external reward

$r^i$ : the reward correspond to the motivation  $m^i$

$\kappa$ : the weighted parameter for positive reward,  $0 < \kappa < 1$

$\lambda$ : the weighted parameter for negative reward,  $\lambda < 0$

$m_{upper}^i$ : the desired upper bound intensity of motivation  $m^i$

$m_{prev}^i$ : the intensity of motivation before action was done

$\nu$ : the threshold for determining the reward of standing by action, and it is an integer. DR: desired range of  $m^i$

- **Deep Reinforcement Learning Algorithm**

- The deep reinforcement learning algorithm we applied in decision making system is the Dueling Deep Q-Network (DDQN). The reason for choosing a value based reinforcement learning is that for policy based reinforcement learning and actor-critic based reinforcement learning are easily reach to optimal goal and stuck there. However, the value based method can take the advantage for reaching the global optimal more easily. We hope that the robot can adapt to and kinds of environment, so we set the initial situation randomly while training. Therefore, to our best knowledge and empirical testing, both

policy based method and actor-critic based method cannot work better than value based method like DDQN. The model architecture in this work is shown in Fig. 3. 6. Since the state is of 6 dimensions and the action is of 7 dimensions, the input dimension is 6 and the output dimension is 7. All the layers here are fully connected layer. The algorithm utilizes the advantage of Double DQN to overcome the estimation error of max-operation on Q-value and also takes advantage of the dueling architecture for separating the state-value and action-advantage estimation. To sum up, the algorithm is shown in Fig. 3. 7, and the action will be chosen by the max Q-value after training.

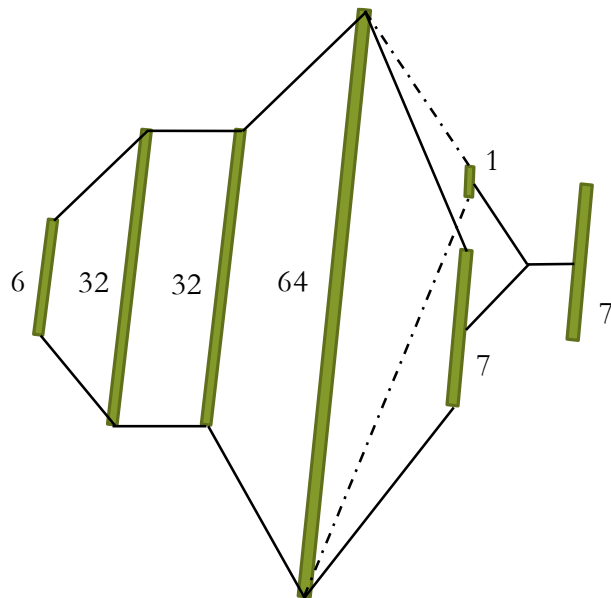


Fig. 3. 6 The model architecture of the DDQN in this work for decision making system. All the green blocks mean the fully connected layers, and the number next to it is the number of neurons in the layer.




---

**Algorithm 4** Dueling Deep Q-Network
 

---

Initialize the replay memory  $D_{exp}$   
 Initialize the weight of the main Q network with arbitrary parameter  $\theta$ , and the state-value layer with parameter  $\beta$  and the action-advantage layer with parameter  $\alpha$   
 Initialize the weight of the target Q network with parameter  $\theta' := \theta$ ,  $\beta' := \beta$ , and  $\alpha' := \alpha$   
 Learning rate:  $\eta \in [0, 1]$   
 Discounting factor:  $\gamma \in [0, 1]$   
 Epsilon value:  $\epsilon \in [0, 1]$

- 1: *loop*:
- 2: **for** episode = 1 to M **do**
- 3:   Receive the initial observation state  $s_1$
- 4:   **for** t = 2 to T and s is not terminal **do**
- 5:      $Q(s, a; \theta, \alpha, \beta) = V(s; \theta, \beta) + (A(s, a; \theta, \alpha) - \frac{1}{|\mathcal{A}|} \sum_{a'} A(s, a'; \theta, \alpha))$
- 6:     Manipulate the  $\pi$  based on Q with the  $\epsilon - greedy$  strategy
- 7:     **if**  $\epsilon \geq 0.05$  **then**
- 8:        $\pi(s_t) \leftarrow$  random action
- 9:     **else**
- 10:       $\pi(s_t) \leftarrow \arg \max_a Q(s_t, a_t | \theta)$
- 11:       $a_t \leftarrow \pi(s_t)$
- 12:       $r_t \leftarrow R(s_t, a_t)$
- 13:       $s_{t+1} \leftarrow$  be observed after doing the new action
- 14:      Store the tuple of  $(s_t, a_t, r_t, s_{t+1})$  in the experience replay  $D_{exp}$
- 15:      Sample a random mini-batch  $(s_i, a_i, r_i, s_{i+1})$  of transitions from  $D_{exp}$
- 16:
- 17:      
$$y_i = \begin{cases} r_i & \text{if } s_{i+1} \text{ is a terminal state} \\ r_i + \gamma Q(s_{i+1}, \arg \max_{a'} Q(s_{i+1}, a'; \theta, \alpha, \beta); \theta', \alpha', \beta') & \text{otherwise} \end{cases}$$
- 18:
- 19:      Perform the gradient descent step on  $(y_i - Q(s_i, a_i | \theta))^2$  with respect to  $\theta$
- 20:      Replace the parameter  $\theta'$  in target network with the parameter  $\theta$  in the main network every C time steps.
- 21:       $s_t \leftarrow s_{t+1}$
- 22: **return** Q

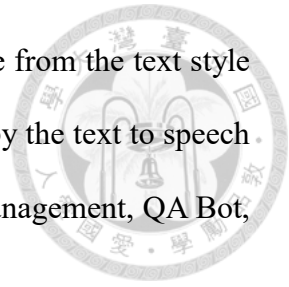
---

Fig. 3. 7 The algorithm for DDQN in the decision making system.

### 3.3 Dialogue System

Dialogue system is utilized to handle the action of social chatting, and the integrated system module is shown in Fig. 3. 3. In the human-robot interaction via chatting, we will utilize the speech to text (STT) module to transfer the speech into text, and use dialogue management to choose the kind of dialogue preferred by the human. If chit-chat is chosen by the human, a sequence chat-bot, to which the text style transfer module is connected, is applied. If the Question-Answering Bot (QA Bot) is operating, some

specific kinds of questions can be answered. After answering is done from the text style transfer module or QA Bot module, the answers will be spoken out by the text to speech (TTS) module of the robot. In the following, STT, TTS, dialogue management, QA Bot, and chit-chat bot will be first introduced:



- **STT module**

- Google speech API, which is a speech recognition library of python, is used for transferring the audio signal to the text.

- **TTS module**

- Aldebaran Naoqi API, which is a built-in library for our robot, is used to transfer the text into audio signal, which is then turned into sound.

- **Dialogue management**

- A webpage is made for human to choose different kinds of chatting, and it is shown on the touch pad mounted on the robot shown as the Fig. 3. 8. The user just needs to touch the button on the pad to choose. After the preferred button is pressed, the text will be passed directly to “the sequence to sequence with attention chit-chat module” or passed through the “entity extraction and keyword classification module” to select different QA functions. The flow of the dialogue management is shown in Fig. 3. 9.

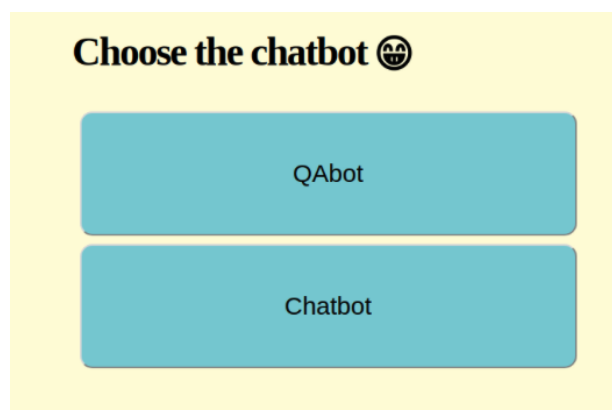


Fig. 3. 8 The webpage interface on the pad of the robot.

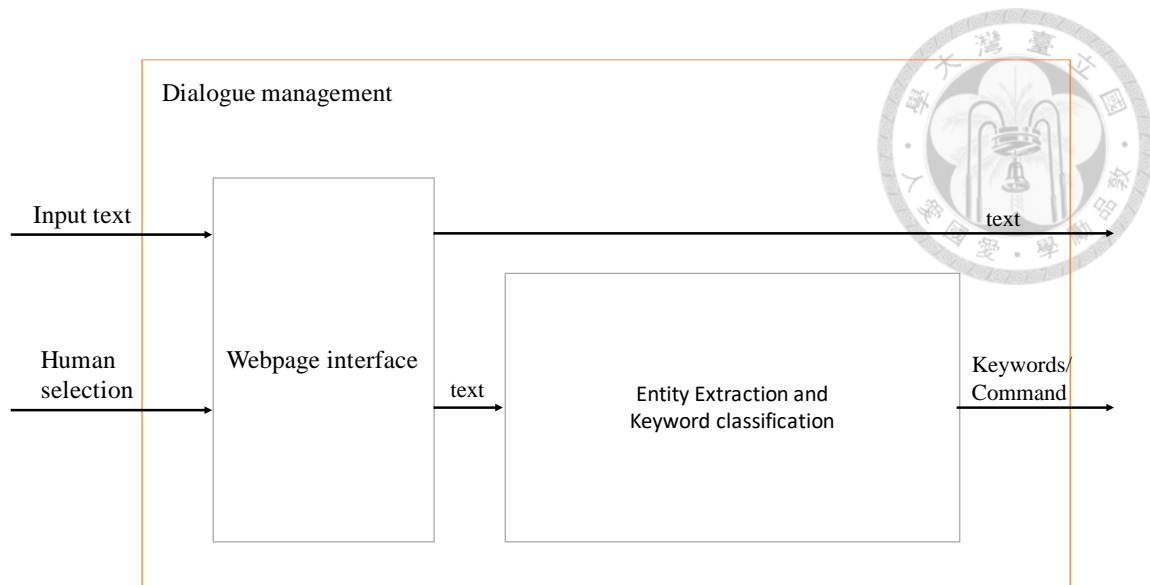


Fig. 3. 9 The flow of the dialogue management.

- The Entity Extraction and Keyword Classification for QA Bot:** There are several categories of questions that the robot can answer or react to, and it is listed in Table. 3. 4. There are 8 types in total, such as information retrieval, weather, time, news, related topics, google map, playing music, and taking photo. We will first try to tell the category by certain words  $\zeta$ , and utilize the Natural Language Toolkit (NLTK), like word tokenizer, part-of-speech (pos) tagging, and name entity recognition, to get useful message for answering. In other words, the function will extract the keyword query for the QA Bot. After knowing the category of the question, the label of name entity of each word will be checked first. If the word has the label, the word will be added to the keyword query list. Then, check each pos-tagging for the word that is without label of name entity. The words with useful pos-tagging will also be added into the keyword query list. Finally, one utilizes the keywords for the QA Bot.

For example, if the human asks what “National Taiwan University” is, the pos-tagging of word which is NN, NNS, NNP, CD or has label of name entity will be extracted out. Thus, “National Taiwan University” will be extracted.

Moreover, once  $\zeta$  is not in “other” category, the keyword, National Taiwan University, will be categorized into information retrieval type, and the description is searched from the google knowledge graph.

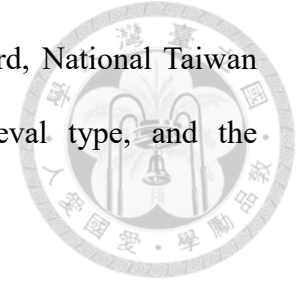


Table. 3. 4 The categories of Questions and how the functions react correspondingly.

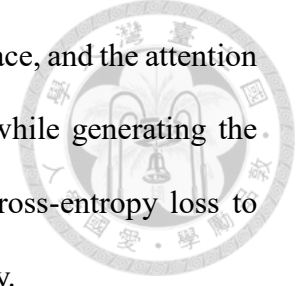
Categories of Questions	Description of the answering function in QA Bot
Information Retrieval type	<p>Search the description of the keyword which is given, and it is manipulated by search in the Google Knowledge Graph. In addition, the website of where the description is found will be shown on the touch pad of the robot.</p> <ul style="list-style-type: none"> <li>✓ <u>Checking category: If <math>\zeta</math> not in other categories</u></li> <li>✓ <u>Useful pos-tagging: NN, NNS, NNP, CD</u></li> </ul>
Weather	<p>Given the keyword of the city, a web crawler is programmed to search the weather information from <a href="http://www.weather-forecast.com">http://www.weather-forecast.com</a>. The website is also shown on the touch pad.</p> <ul style="list-style-type: none"> <li>✓ <u>Checking category: <math>\zeta</math>: weather</u></li> <li>✓ <u>Useful pos-tagging: NN, NNS, NNP, CD</u></li> </ul>
Time	<p>Given the keyword of the city, the google map API (GoogleV3) is utilized to find out the latitude and longitude. Then, the python package (tzwhere) is used to get the time information, and a website of clock will be shown on the touch pad.</p> <ul style="list-style-type: none"> <li>✓ <u>Checking category: <math>\zeta</math>: what time</u></li> <li>✓ <u>Useful pos-tagging: NN, NNS, NNP, CD</u></li> </ul>
News	<p>A web crawler is programmed to crawl the news titles from google news, and the news page will be shown on the touch pad.</p>

	<ul style="list-style-type: none"> <li>✓ <u>Checking category: ζ: news</u></li> <li>✓ <u>Useful pos-tagging: NN, NNS, NNP, JJ,CD</u></li> </ul>
Related topics	<p>Given the keyword of the related topic, we utilize the python API of Google Trends to search the related topics, and shows those topics on the touch pad.</p> <ul style="list-style-type: none"> <li>✓ <u>Checking category: ζ: related topics</u></li> <li>✓ <u>Useful pos-tagging: NN, NNS, NNP, CD, JJ</u></li> </ul>
Google map	<p>Given the keyword of the place or city, google map is shown on the touch pad.</p> <ul style="list-style-type: none"> <li>✓ <u>Checking category: ζ: where</u></li> <li>✓ <u>Useful pos-tagging: No need, because the name entity recognition can capture the useful information.</u></li> </ul>
Play music	<p>Given the keyword of the music, the youtube video will be played on the touch pad.</p> <ul style="list-style-type: none"> <li>✓ <u>Checking category: ζ: play music</u></li> <li>✓ <u>Useful pos-tagging: No need, because the youtube will search the most related one to what the sentence has mentioned.</u></li> </ul>
Photo taking	<p>Given the keyword command, we will take the photo through the camera of the robot and upload it to our webpage for showing on the touch pad.</p> <ul style="list-style-type: none"> <li>✓ <u>Checking category: ζ: take photo, take picture</u></li> <li>✓ <u>Useful pos-tagging: No need, because it is a command.</u></li> </ul>

- **The Chit-Chat Bot**

- The deep learning based chit-chat bot module is applied here, and it is a sequence to sequence model with attention mechanism. This model is an auto-

encoder model for encoding the input text into the latent space, and the attention mechanism for focusing on the input tokens is utilized while generating the output sentence. The optimization method is the basic cross-entropy loss to calculate the generated tokens in the discrete sampling way.



### 3.3.1 Text Style Translation

The output sentence of the chit-chat bot and the output mood from the mechanism of the moods will be the input of the text style translation module, which is a function to translate the sentence from one style to another. This module is largely inspired by Shen *et al.* [66], Zhao *et al.* [70], and Melnyk *et al.* [71]. In the later sections, the problem formulation of our text style translation will be briefly introduced and we will show the model architecture we have proposed. In text style translation module, we utilize the pre-trained language model to speed up the convergence, and train it with cycle consistency loss for preserving a better content from source sentence. For possible multiple moods for robot, we utilize N+1 discriminator to reduce the needed parameters. N+1 discriminator is also a better choice for the adversarial training to handle this task.

### 3.3.2 Formulation and Model Architecture of Text Style Translation

It is considered as an unsupervised learning problem, because there are few parallel corpora to use. The idea of the work is shown in Fig. 3. 10. There are sentence data,  $x_1$  and  $x_2$ , from different domains,  $\mathcal{X}_1$  and  $\mathcal{X}_2$ , respectively, and their respective labels for each style, say,  $\delta_1$  and  $\delta_2$ , will be encoded into the latent space as  $y_1$  and  $y_2$ , respectively, by  $E_y$ . Then, the encoder  $E_z$  will encode both  $x_1$  with  $y_1$  and  $x_2$  with  $y_2$  into the same latent space as  $z_1$  and  $z_2$ , respectively, where  $z$  is the latent variable.



For generating the data from  $\mathcal{X}_2$  to  $\mathcal{X}_1$ , we input the encoded  $z$  and the style  $y_1$  to the generator  $G$ , and the generated data is denoted as  $\hat{\mathcal{X}}_2$ . In contrast, for generating the data from  $\mathcal{X}_1$  to  $\mathcal{X}_2$ , we input the encoded  $z$  and the style  $y_2$  to the generator  $G$ , and the generated data is denoted as  $\hat{\mathcal{X}}_1$ .

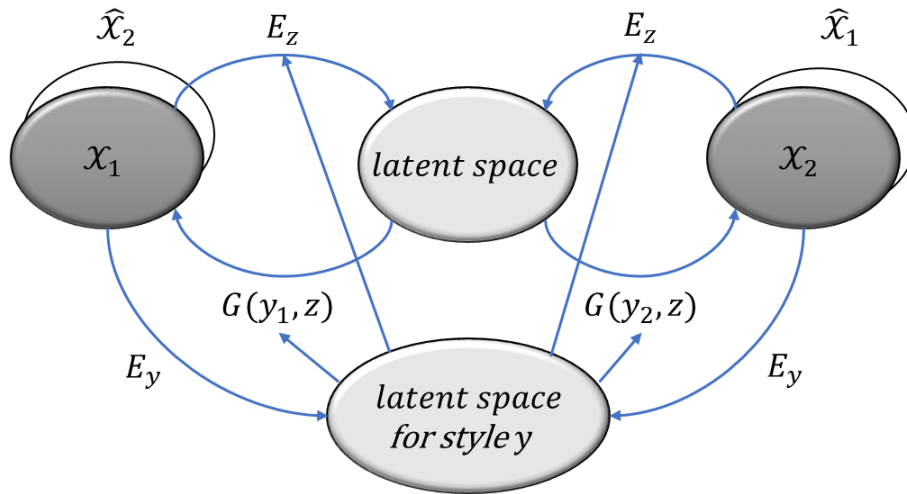


Fig. 3. 10 The idea of the text style translation in this work.

In order to force the encoded latent representations in the same latent space and the generated sentence sharing the same data distribution with desired style of data domain, we utilize  $N + 1$  discriminator for the adversarial learning. Also, while generating the style representation  $y$ , the auto-encoder is applied to make  $y$  become more meaningful information. The cycle consistency loss is here to preserve more content from the source sentence, and therefore after the style of generated sentence is transferred, the generated sentence should be possible to transfer back to the original source sentence. The learning flow for training is shown in Fig. 3. 11. While testing, we will input the style representation which is determined by the mechanism of moods to the generator, and the opposite style representation will be inputted to the encoder.

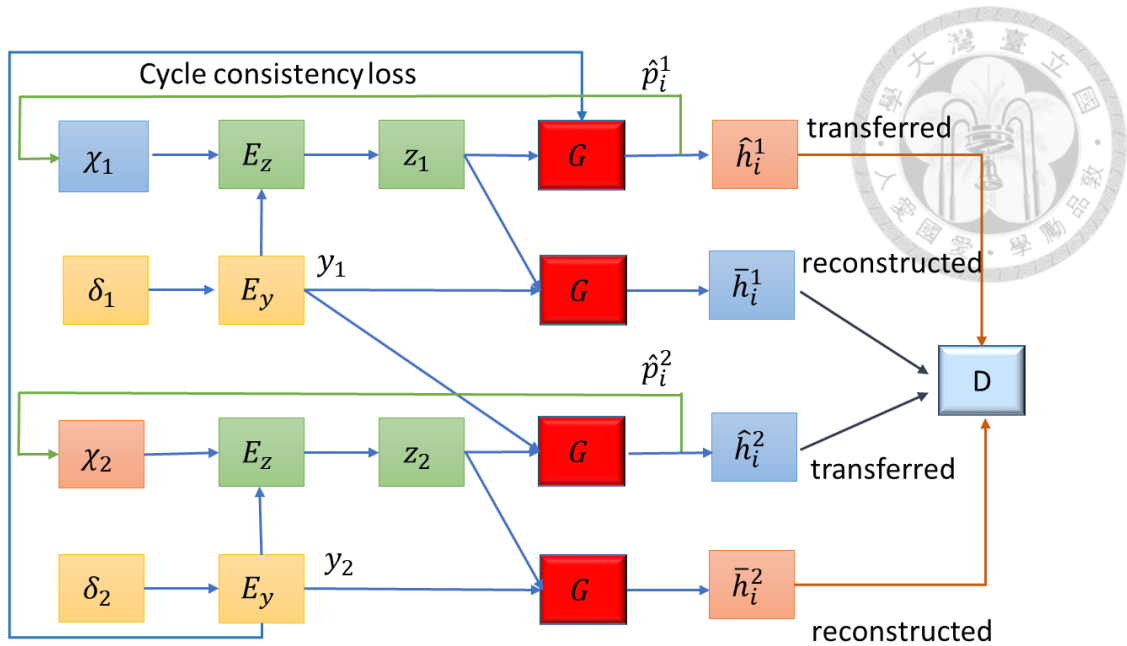


Fig. 3. 11 The learning flow of the text style translation module in the work.

The model architecture of  $E_y$  is shown in Fig. 3. 12, and it is an auto-encoder architecture. The 200 dimensional vector is the latent representation of the style  $y$ .

The model architecture of  $E_z$  is shown in the Fig. 3. 13, which is a two layer Long Short Term Memory (LSTM). The initial state in the first layer LSTM is a concatenated vector, which concatenates the vectors filled with zeros and the latent representation of style  $y$ . The output of the second layer in the LSTM is the encoded latent representation of the sentence  $z$ , which will be passed to the generator  $G$ . The generator  $G$  is one layer

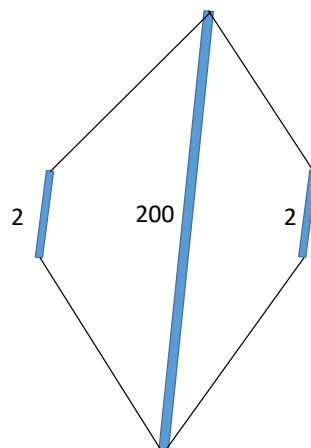
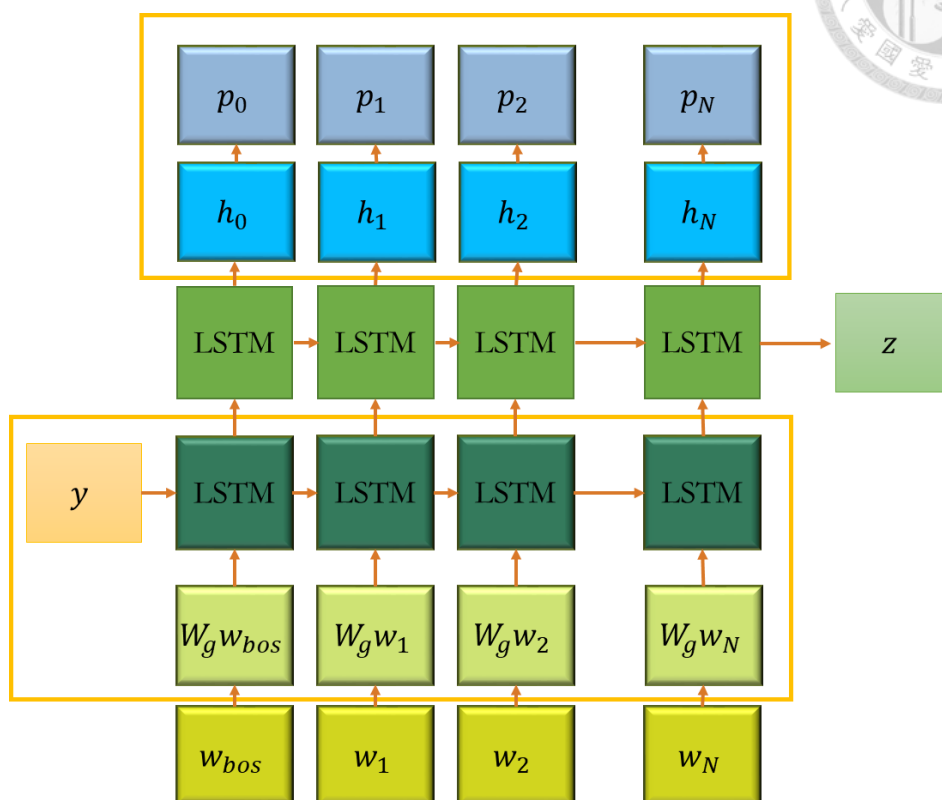


Fig. 3. 12 The model architecture of the auto-encoder  $E_y$ . The 200 dimensions vector is the latent variable  $y$ .



LSTM, and the embedding matrix shared the same weight with the word embedding matrix in the encoder  $E_z$ .



Where,

$\{w_{bos}, w_1, w_2, \dots, w_N\}$ : the tokens in the sentence.

$W_g$ : the word embedding matrix which is the same in generator.

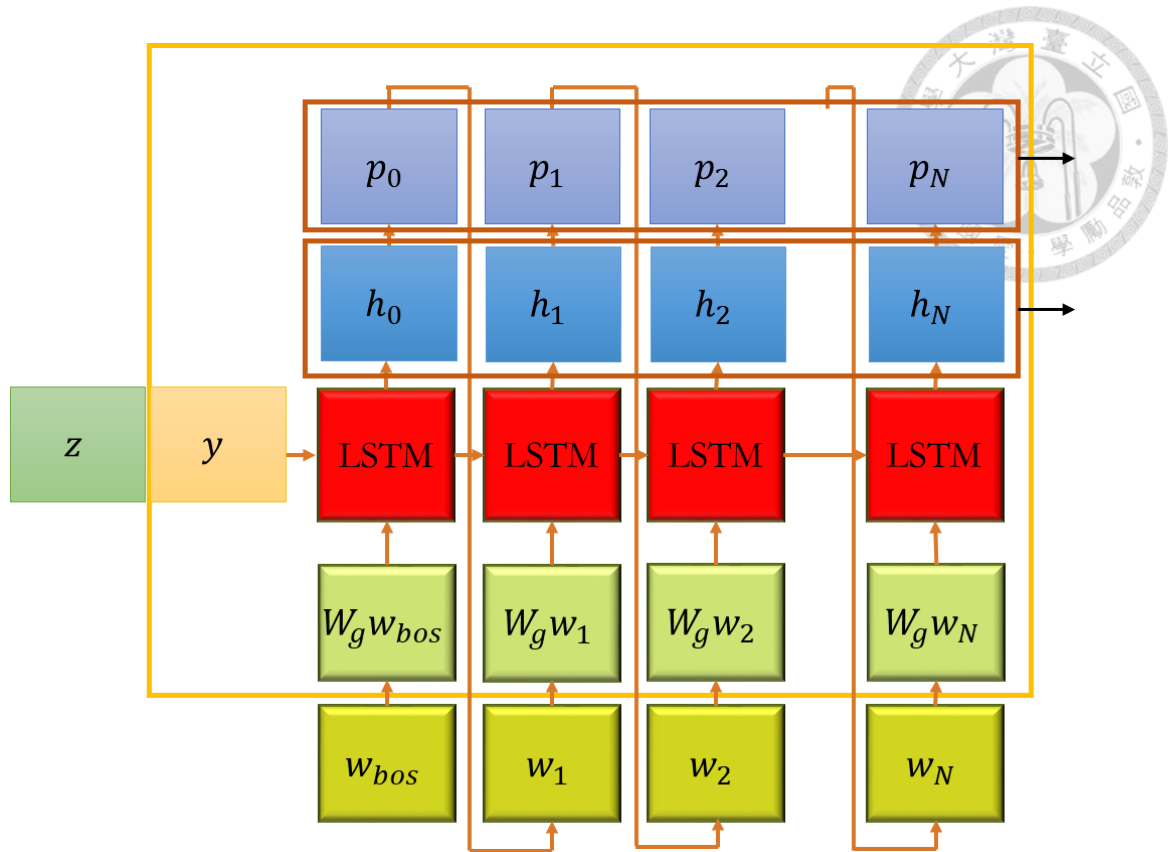
$y$ : the style latent variable,  $z$ : the encoded latent representation

$\{h_0, h_1, h_2, \dots, h_N\}$ : is the output state information of LSTM

$\{p_0, p_1, p_2, \dots, p_N\}$ : the generated logits

Fig. 3. 13 This is the model architecture of the encoder  $E_z$ .

The initial state of LSTM is the concatenated vector concatenating the style latent representation  $y$  and the sentence latent representation  $z$ . The model architecture of the generator is shown in Fig. 3. 14, whereas the one for the  $N+1$  discriminator is the same architecture of the TextCNN [93], which is a model for the sentiment classification for text using CNN.



Where,

$\{w_{bos}, w_1, w_2, \dots, w_N\}$ : the tokens of the sentence.

$W_g$ : the word embedding matrix which is the same in encoder.

$y$ : the style latent variable,  $z$ : the encoded latent representation

$\{h_0, h_1, h_2, \dots, h_N\}$ : is the output state information of LSTM.

$\{p_0, p_1, p_2, \dots, p_N\}$ : the generated logits

Fig. 3. 14 This is the model architecture of the generator  $G$ .

Before the adversarial training for generating different styles of sentences and the training for sentence reconstruction, this work pre-trains the language model first. The orange box in both encoder and generator figures and the auto-encoder of the style will be trained by the following equations (3. 8), (3. 9), and (3. 10), and they are all reconstruction optimization.

For the auto-encoder of the style, the loss function is listed below:



Mean square error:

$$L_s = -\log p_{\theta_s, \varphi_s}(\delta^{(i)}) = \frac{1}{N} \sum_i^N (\delta^{(i)} - \hat{\delta}^{(i)})^2 \quad (3.8)$$

$$\theta_s^*, \varphi_s^* = \arg \min_{\theta_s, \varphi_s} L_s$$

$\theta_s$ : The parameter of the encoder part  
 $\varphi_s$ : The parameter of the generator part  
 $\delta^{(i)}$ : the input,  $\hat{\delta}^{(i)}$ : the generated output  
 $N$ : the size of mini-batch

For the encoder of sentence, the loss function is listed below:

Cross-Entropy: Given the style latent representation  $y$  from  $E_y$

$$L_E(\theta_s, \theta_E) = - \sum_i^N \sum_j^l \log p_{E_z}(w_j^{(i)}) = - \sum_i^N \sum_j^l w_j^{(i)} \log \hat{w}_j^{(i)} \quad (3.9)$$

$$\theta_s^*, \theta_E^* = \arg \min_{\theta_s, \theta_E} L_E$$

$\theta_s$ : The parameter of the encoder of style  
 $\theta_E$ : The parameter of the encoder part  
 $w_j^{(i)}$ : the ground truth input token with one-hot encoding  
 $\hat{w}_j^{(i)}$ : the generated logit, the same as  $p_j$  in the figure  
 $N$ : the size of mini-batch,  $l$ : the sequence length

For the generator, the loss function is listed below:

Cross-Entropy: Given the style latent representation  $y$  from  $E_y$

$$L_G(\theta_s, \varphi_G) = - \sum_i^N \sum_j^l \log p_G(w_j^{(i)}) = - \sum_i^N \sum_j^l w_j^{(i)} \log \hat{w}_j^{(i)} \quad (3.10)$$

$$\theta_s^*, \varphi_G^* = \arg \min_{\theta_s, \varphi_G} L_G$$

$\theta_s$ : The parameter of the encoder of style



- $\varphi_G$ : The parameter of the encoder part
- $w_j^{(i)}$ : the ground truth input token with one-hot encoding
- $\hat{w}_j^{(i)}$ : the generated logit, the same as  $p_j$  in the figure
- $N$ : the size of mini-batch,  $l$ : the sequence length

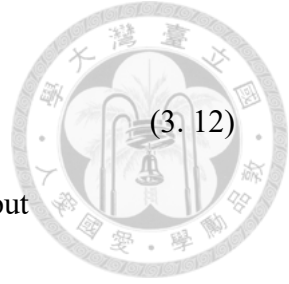
After pre-training the language model, there are mainly three loss functions for training, such as the reconstruction loss, the adversarial loss, and cycle consistency loss. The reconstruction of the sentence means that the sentence from the source domain will be reconstructed back to itself with the original style. The reconstructed loss function is shown as equation (3. 11), and the inputs for the generator are all the ground truth tokens.

$$\begin{aligned}
 L_{rec}(\theta_S, \theta_E, \varphi_G) = & \mathbb{E}_{x_1 \sim \mathcal{X}_1} \left[ -\log p_G \left( x_1 \mid E_y(\delta_1), E_z \left( x_1, E_y(\delta_1) \right) \right) \right] \\
 & + \mathbb{E}_{x_2 \sim \mathcal{X}_2} \left[ -\log p_G \left( x_2 \mid E_y(\delta_2), E_z \left( x_2, E_y(\delta_2) \right) \right) \right]
 \end{aligned} \tag{3. 11}$$

For the adversarial training, Gumbel-Softmax of continuous relaxation for discrete sampling process and the Professor Forcing for aligning the sentence population, which is also presented in Shen *et al.* [66], Lample *et al.* [69], and Zhao *et al.* [70], should be introduced first, and it is the one of common way for neural text style translation.

### 3.3.3 Gumbel-Softmax and Professor Forcing for Text Generation

- **Gumbel-Softmax of continuous relaxation for discrete sampling process**
  - The technique is to utilize the sum of the probability of previous generated logit and a random variable sampled from Gumbel distribution and divided by a temperature parameter to be the next input in the generator for the text generation. It has the same property as the re-parameterization trick while updating for the continuous relaxation.



$$w_{i+1} = f_G(p_i, G_i) = \text{softmax}\left(\frac{p_i + G_i}{\tau}\right) \quad (3.12)$$

$p_i$ : the probability of previous output logit,  $w_{i+1}$ : the next input  
 $G_i \sim G_{um}$  (Gumbel distribution):  $G_{um} = -\log(-\log(U))$   
 $U \sim \text{Unif}[0,1]$  (the uniform distribution)  
 $\tau$ : the temperature parameter in range  $(0,1]$

- **Professor Forcing for aligning the sentence population**

- The idea is shown in Fig. 3. 15, and it is an example of showing that the alignment of sentence population from style  $y_2$  to style  $y_1$ . The concept idea is to align two sequences output of hidden states in the LSTM layer by a discriminator, which is the TextCNN with five kinds of kernel filter size [1,2,3,4,5]. One sequence output of hidden states is that inputs are tokens in ground truth sentence, and other kind is that inputs are from previous generated logits. The discriminator is a  $N + 1$  discriminator in this work, and it will be mentioned in the next Section.

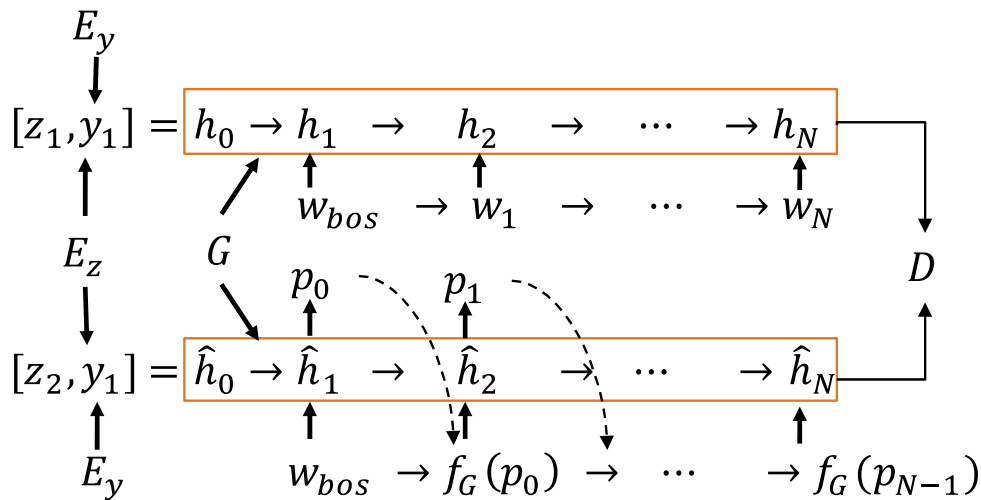
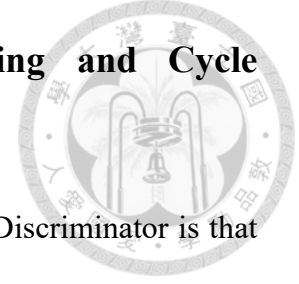


Fig. 3. 15 The idea of aligning the sentence population.

### 3.3.4 N+1 Discriminator with Adversarial Training and Cycle Consistency Loss



The difference between the general discriminator and  $N + 1$  Discriminator is that  $N + 1$  Discriminator can handle not only true and false label but also multi-classes label. Thus, there are three different labels that are style 1  $\lambda_1$ , style 2  $\lambda_2$ , and fake style  $\lambda_3$  in this work. In this text style translation part, we are translating from positive style to negative style or from negative style to positive. The discriminator is learning how to classify the real distributions of the sentence with style 1 reconstructed back to the sentence with style 1, the sentence with style 2 reconstructed back to the sentence with style 2, and the style of the sentence that are being translated. While training, sentences being translated by the generator will easily contain both words of original style and words of translated style, so the translated sentence should be in the same distribution as label fake style  $\lambda_3$  for discriminator. Later on, the generator will try to update parameters by convincing discriminator that the sentence distribution of translated sentences should be same classes as the real distribution of reconstructed one for corresponded styles. The loss function of the discriminator and generator for adversarial training is explained below in equation (3. 13).

$h_1^{(i)}$ : the sequence of hidden states which is unrolled from  $G$  with initial state  $[z_1, y_1]$

$h_2^{(i)}$ : the sequence of hidden states which is unrolled from  $G$  with initial state  $[z_2, y_2]$

$\hat{h}_1^{(i)}$ : the sequence of hidden states which is unrolled from  $G$  with initial state  $[z_1, y_2]$ , and input with previous generated logits

$\hat{h}_2^{(i)}$ : the sequence of hidden states which is unrolled from  $G$  with initial state  $[z_2, y_1]$ , and input with previous generated logits

$\theta_D$ : the parameter of discriminator,  $N$ : the size of mini-batch





For Discriminator:

$$\hat{\lambda}_1 \leftarrow D(h_1^{(i)})$$

$$\hat{\lambda}_2 \leftarrow D(h_2^{(i)})$$

$$\hat{\lambda}_3 \leftarrow D(\hat{h}_1^{(i)}), D(\hat{h}_2^{(i)})$$

$$L_D(\theta_S, \theta_E, \varphi_G, \theta_D) = - \sum_j^3 \frac{1}{N} \sum_k^N \lambda_j \log \hat{\lambda}_j \quad (3.13)$$

For Generator:

$$\hat{\lambda}_1 \leftarrow D(\hat{h}_2^{(i)})$$

$$\hat{\lambda}_2 \leftarrow D(\hat{h}_1^{(i)})$$

$$L_G(\theta_S, \theta_E, \varphi_G, \theta_D) = - \sum_j^2 \frac{1}{N} \sum_k^N \lambda_j \log \hat{\lambda}_j$$

The idea of cycle consistency loss is to make the generated sentence which has been translated the style reconstruct back to the original sentence with original style, and the loss function of it is listed in equation (3.14).

$\hat{x}_1 \sim \hat{\mathcal{X}}_1$ : the sentence with style 1 translated to style 2 (the sequence of generated logits by inputs with previous logits)

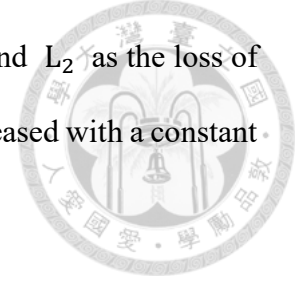
$\hat{x}_2 \sim \hat{\mathcal{X}}_2$ : the sentence with style 2 translated to style 1 (the sequence of generated logits by inputs with previous logits)

$$L_{cycle}(\theta_S, \theta_E, \varphi_G) = \mathbb{E}_{\hat{x}_1 \sim \hat{\mathcal{X}}_1} \left[ -\log p_G \left( x_1 \mid E_y(\delta_1), E_z(\hat{x}_1, E_y(\delta_2)) \right) \right] \\ + \mathbb{E}_{\hat{x}_2 \sim \hat{\mathcal{X}}_2} \left[ -\log p_G \left( x_2 \mid E_y(\delta_2), E_z(\hat{x}_2, E_y(\delta_1)) \right) \right] \quad (3.14)$$

In sum, it is going to minimize  $L_1 = L_S + L_{rec} + L_{cycle} + L_G$  with respect to parameter of  $\theta_S, \theta_E, \varphi_G$  and to minimize the  $L_2 = L_D$  with respect to parameters of  $\theta_D$ .

The training process is the same as we mentioned the algorithm in GAN of our

preliminary, and  $L_1$  can be considered as the loss of the generator and  $L_2$  as the loss of discriminator. In addition, the temperature parameter  $\tau$  will be decreased with a constant gain  $\varepsilon$  for every  $c$  epoch.



### 3.4 Visual Question Generation

In order to meet the internal need of curiosity of the robot and improve the human robot interaction, Visual Question Generation (VQG) is presented to our robot. The VQG module utilizes images captured from the camera on the robot, and generate the questions related to what the robot observes. For the advantages of diverse proposed questions for a single scene to make human feel the robot smarter, the deep learning based generative model is chosen. The problem formulation and the model architecture will be introduced first, and the methodology will be explained in the later Section.

#### 3.4.1 Problem Formulation and Model Architecture of VQG

The concept of the VQG in this work is shown in Fig. 3. 16. An image encoder  $E_I$  encode the information of the image, and the encoder  $E_z$  will encode both encoded image features and the sentence,  $x \in \mathcal{X}$ , into latent space with the latent variable  $z$  by Categorical Variational Auto-Encoder with Gumbel-Softmax techniques. It will encode the last hidden state information in encoder with random noise  $g$  sampled from the Gumbel distribution  $G_{um}$  by the reparameterization trick. The generator  $G$  will generate the sentence with the initial state as the concatenation of the latent representation  $z$  and encoded image feature. The same techniques of Gumbel softmax of continuous relaxation for discrete sampling process and the Professor Forcing for aligning the sentence population is applied here, and is used to align two sequences of hidden states

by the discriminator  $D_1$  for adversarial training. One of them is that the inputs are the ground truth tokens, and another is that the inputs are the previous generated logits with continuous relaxation. In addition, because some generated sentences do not match to the image in other previous works, the adversarial training is utilized for improving the relatedness between the generated sentence and the image. The discriminator  $D_2$  will try to classify the difference between the generated sentence with inputs of previous logits with encoded image feature and the ground truth sentence with the encoded image feature.

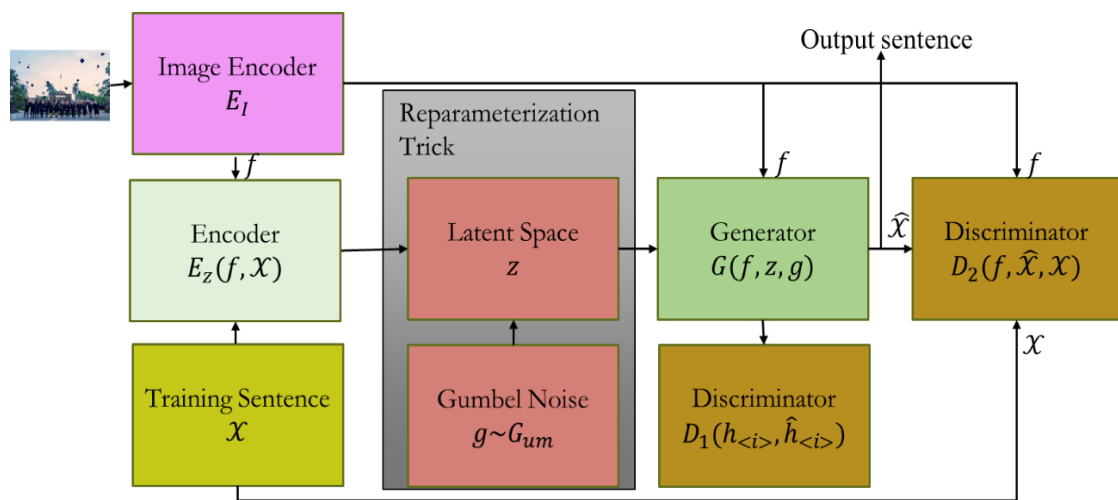


Fig. 3. 16 The concept of the VQG in this work.

- **Image Encoder**

- The VGG16 pre-trained model, which is trained on the object recognition of ImageNet, is used for extracting the features from the image, and the first layer of 4096 neurons is connected to a fully connected layer of 512 neurons for embedding to a lower dimension. While in the training process, the weight of the VGG16 pre-trained model will be fixed, but the 512 neurons will still be trained. The model architecture is shown in the Fig. 3. 17.

- **Encoder**

- The encoder  $E_z$  is a single layer LSTM that has 512 neurons with the initial state as the encoded image feature  $f$ , and the inputs for each time step are the

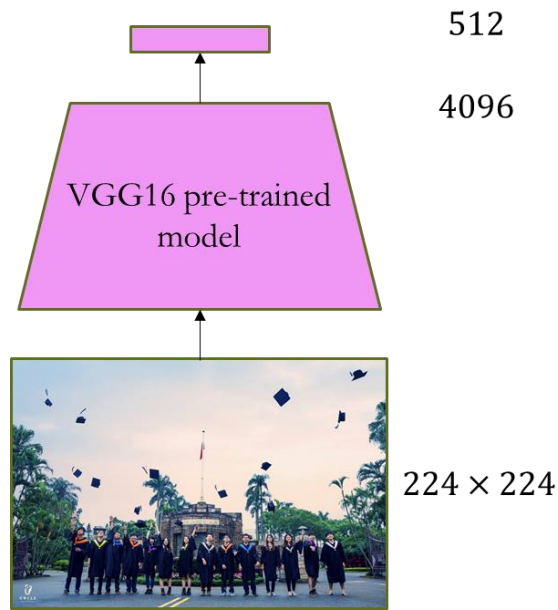
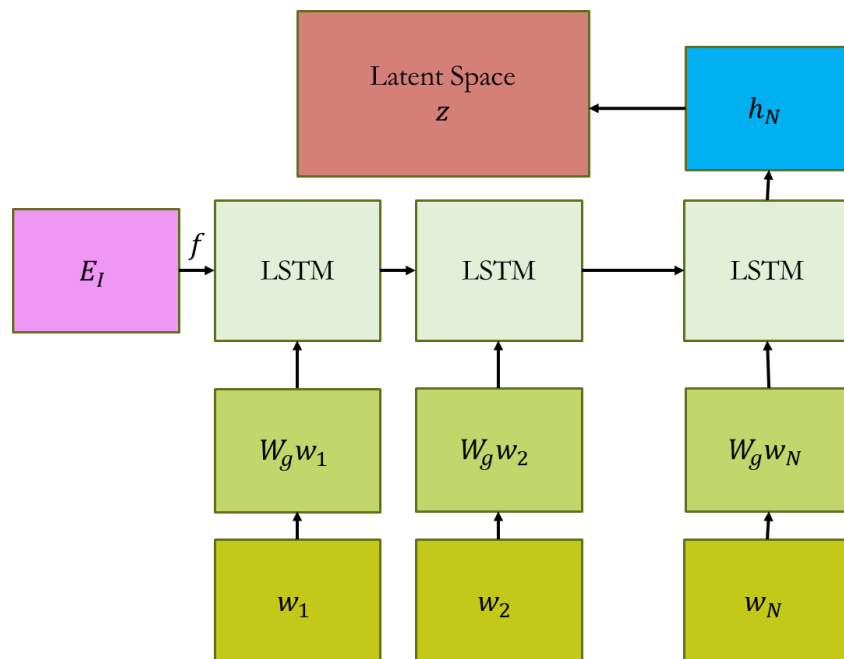


Fig. 3. 17 The model architecture of the image encoder  $E_I$ .

tokens  $\{w_1, w_2, \dots, w_N\}$  in the sentence  $x$ . It will encode last hidden state information of LSTM into 128-dimension latent space vector. The model architecture is shown in the Fig. 3. 18. We utilize the Categorical Variational Auto-Encoder with Gumbel-Softmax techniques to process the last hidden state information into the latent representation.





Where,

$E_I$ : the image encoder which outputs the encoded image feature  $f$

$\{w_1, w_2, \dots, w_N\}$ : the tokens in the input sentence

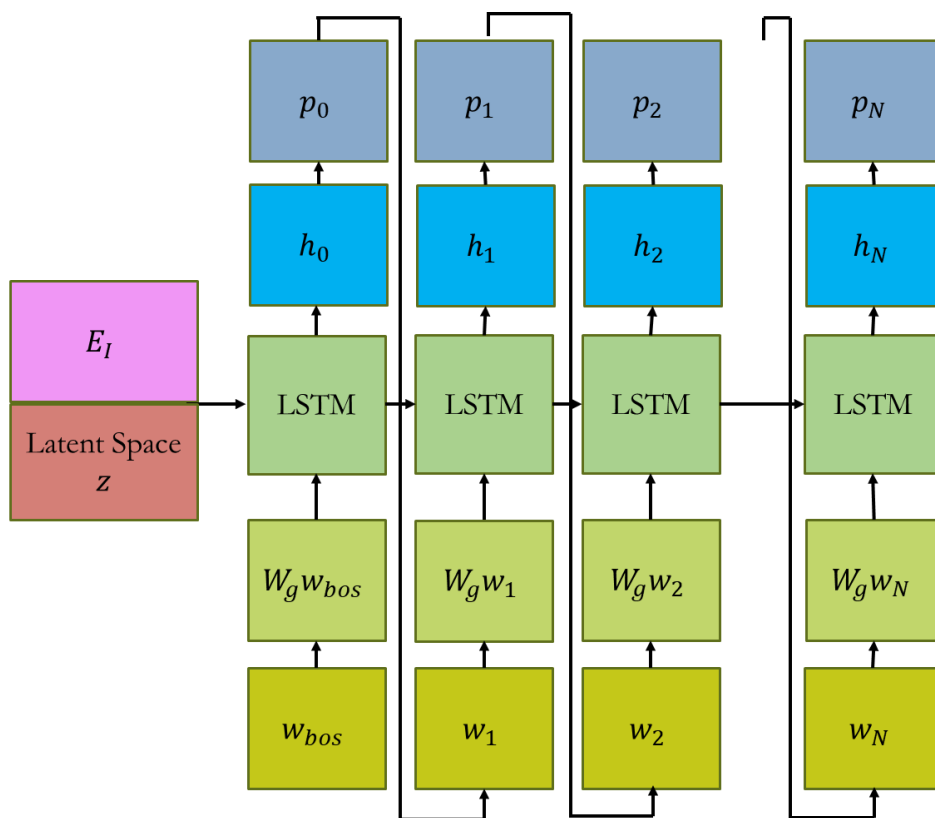
$W_g$ : the word embedding matrix,  $h_N$ : the last output of hidden state

$z$ : the latent representation

Fig. 3. 18 The model architecture of the encoder  $E_z$ .

- **Generator**

- The latent representation from the encoding phase is concatenated with the encoded image feature for being the initial state of the LSTM. Generator  $G$  is a single LSTM with 640 neurons, and the model architecture is shown in Fig. 3. 19.



$E_I$ : the image encoder which outputs the encoded image feature  $f$

$W_g$ : the word embedding matrix same as the encoder

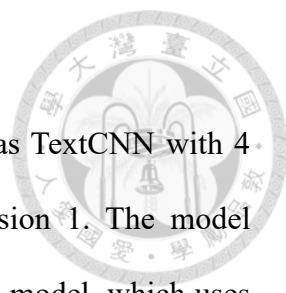
$\{w_{bos}, w_1, w_2, \dots, w_N\}$ : the tokens in the input sentence

$\{h_0, h_1, h_2, \dots, h_N\}$ : the output sequence of hidden state

$\{p_0, p_1, p_2, \dots, p_N\}$ : the output sequence of generated logits

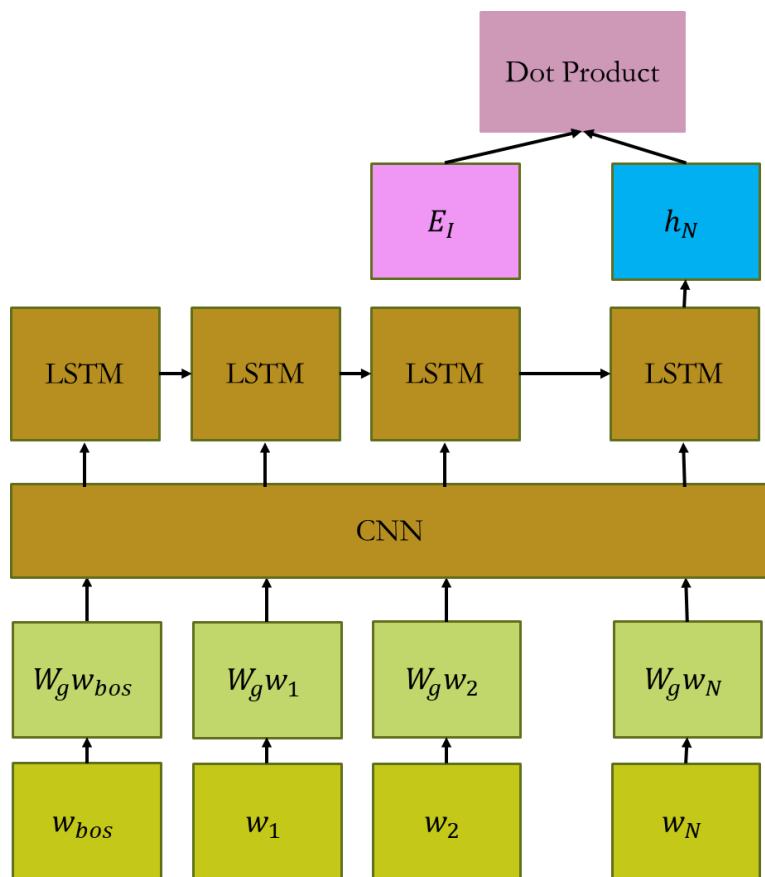
$z$ : the latent representation,  $g$ : the random noise sampled from Gumbel distribution

Fig. 3. 19 The model architecture of the generator  $G$ .



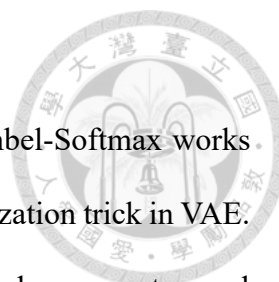
- **Discriminator**

- The model architecture of discriminator  $D_1$  is the same as TextCNN with 4 kinds of kernel filter size [1,2,3,4] with output dimension 1. The model architecture of discriminator  $D_2$  is a general CNN-LSTM model, which uses kernel filter size 2, and stride 1 for CNN layer and LSTM with 512 neurons. After getting the last output of hidden state in LSTM, the dot product operation for it and the encoded image feature is applied for the output of the discriminator. Thus, the output dimension of  $D_2$  is 1, and model architecture is shown in Fig. 3. 20.



$E_I$ : the image encoder which outputs the encoded image feature  $f$   
 $\{w_1, w_2, \dots, w_N\}$ : the tokens in the input sentence  
 $W_g$ : the word embedding matrix which is the same as encoder  
 $h_N$ : the last output of hidden state

Fig. 3. 20 The model architecture for the discriminator  $D_2$ .



Then, how the Categorical Variational Auto-Encoder with Gumbel-Softmax works will be explain first, and the idea of it is the same as the re-parameterization trick in VAE. Finally, we will introduce the loss functions for encoder, image encoder, generator, and two discriminators and adversarial training in the next Section.

The concept is the same as the one we mentioned previously in (3. 12), but we re-define the notations in (3. 15) for making it more clear and understandable. The last hidden state  $h_N$  from the encoder and the random noise sampled from the Gumbel distribution will be summed up, and divided by a temperature parameter for annealing. Then, the softmax activation is applied for it to be one of the factor of the initial state in the generator.

$$z = softmax\left(\frac{h_N + g}{\tau}\right) \tag{3. 15}$$

$z$ : the latent representation from encoder  
 $g \sim G_{um}$  (Gumbel distribution):  $G_{um} = -\log(-\log(U))$   
 $U \sim Unif[0,1]$  (the uniform distribution)  
 $\tau$ : the temperature parameter in range  $(0,1]$

### 3.4.2 The Loss Function and Adversarial Training

There are mainly three loss function, such as reconstruction loss for text generation, adversarial loss for continuous relaxation of text generation, and adversarial loss for improving the correlation of generated sentence and the image.

- **Reconstruction loss for text generation**
  - The reconstruction loss is formulated the same as the loss of VAE, and it is shown in (3. 16). The first term is to minimize the reconstruction loss for the sentence  $x$ . The second term is to minimize the KL-divergence of two distributions that are the latent representation with reparameterization trick  $z$

and the desired sampled distribution, which is sampled from Gumbel distribution. The reason for choosing technique from the Categorical Variational Auto-Encoder with Gumbel-Softmax is that it is learning the categorical features from the input information and can enable large speedups. It inspired us that there are also limited patterns of sentences to generate for the similar images, and there are also limited categories of the scene in the images. In other words, for a certain category of image, we can model the possible sentences as a distribution of it, which is the same idea as the techniques.

$$L_{rec}(\theta_I, \theta_E, \varphi_G) = \mathbb{E}_{x \sim \mathcal{X}, z \sim E_z(x, E_I(I))} [-\log p_{\varphi_G}(x | E_I(I), z)] + \text{KL}(E_z(x, E_I(I)) || p_{\varphi_G}(z)) \quad (3.16)$$

$x$ : sentence,  $z$ : the latent representation with re-parameterization trick from the encoder  $E_z$   
 $\theta_I$ : the parameter of the image encoder  $E_I$   
 $\theta_E$ : the parameter of the encoder  $E$   
 $\varphi_G$ : the parameter of the generator  $G$

- **Adversarial Loss for continuous relaxation of text generation**

- It is the same techniques as the one we mentioned in Section 3.3.4 for translating the style. There are two classes for the discriminator here, and they are  $\lambda_1$  and  $\lambda_2$ .  $\lambda_1$  is for input of tokens in ground truth sentence, and  $\lambda_2$  is for input of previous generated logits. The loss function is defined in the equation (3.17), and the main goal is to align the sentence population of them to avoid bias exposure of discrete sampling.

$h^{(i)}$ : the sequence of hidden states which is unrolled from  $G$  with the input of tokens in ground truth sentence.

$\hat{h}^{(i)}$ : the sequence of hidden states which is unrolled from  $G$  with the input of previous generated logits using the continuous relaxation technique.





$\theta_{D_1}$ : the parameter of discriminator  $D_1$ ,  $N$ : the size of mini-batch  
For Discriminator  $D_1$ :

$$\hat{\lambda}_1 \leftarrow D(h^{(i)})$$

$$\hat{\lambda}_2 \leftarrow D(\hat{h}^{(i)})$$

$$L_{D_1}(\theta_I, \theta_E, \varphi_G, \theta_{D_1}) = - \sum_j^2 \frac{1}{N} \sum_k^N \lambda_j \log \hat{\lambda}_j \quad (3.17)$$

For Generator:

$$\hat{\lambda}_1 \leftarrow D(\hat{h}^{(i)})$$

$$L_{G_1}(\theta_I, \theta_E, \varphi_G, \theta_{D_1}) = - \frac{1}{N} \sum_k^N \lambda_1 \log \hat{\lambda}_1$$

- **Adversarial Loss for improving the correlation of generated sentence and image**

- For making the generated sentence match the image better, the adversarial training is utilize as following equation (3. 18). There are two classes for discriminator  $D_2$ , and  $\lambda_1$  is for the ground truth sentence  $w^{(i)}$  with encoded image feature  $E_I(I)$  and  $\lambda_2$  is for the generated logits  $\hat{w}^{(i)}$  from G with encoded image feature  $E_I(I)$ . The discriminator learns to classify them, and the generator will try to convince the generated logits with the encoded image feature is the same distribution and the ground truth one.

$w^{(i)}$ : the tokens in the ground truth sentence

$\hat{w}^{(i)}$ : the generated logits from G with the input of previous generated logits using the continuous relaxation technique.

$\theta_{D_2}$ : the parameter of discriminator  $D_2$ ,  $N$ : the size of mini-batch

For Discriminator  $D_2$ :

$$\hat{\lambda}_1 \leftarrow D(w^{(i)}, E_I(I)) \quad (3.18)$$

$$\hat{\lambda}_2 \leftarrow D(\hat{w}^{(i)}, E_I(I))$$

$$L_{D_2}(\theta_I, \theta_E, \varphi_G, \theta_{D_2}) = - \sum_j^2 \frac{1}{N} \sum_k^N \lambda_j \log \hat{\lambda}_j$$

For Generator:

$$\hat{\lambda}_1 \leftarrow D(\hat{w}^{(i)}, E_I(I))$$

$$L_{G_2}(\theta_I, \theta_E, \varphi_G, \theta_{D_2}) = - \frac{1}{N} \sum_k^N \lambda_1 \log \hat{\lambda}_1$$

In sum, it is going to minimize  $L_1 = L_{rec} + L_{G_1} + L_{G_2}$  with respect to parameter of  $\theta_I, \theta_E, \varphi_G$  and to minimize the  $L_2 = L_{D_1} + L_{D_2}$  with respect to parameters of  $\theta_{D_1}, \theta_{D_2}$ . The training process is the same as we mentioned the algorithm in GAN of our preliminary, and  $L_1$  can be considered as the loss of the generator and  $L_2$  as the loss of discriminator. In addition, the temperature parameter  $\tau$  will be decreased with a constant gain  $\varepsilon$  for every  $c$  epoch.



## Chapter 4 Experiment

Our experiment platform is the Pepper robot [102], and it is shown in Fig. 4. 1. For a better evaluation, we evaluate each module separately. In the subsection, we will show the simulation result for the autonomous system first. Next, the performances of the chat bot and the text style translation module are discussed. At last, the human evaluation result for the VQG is shown.



Fig. 4. 1 The Pepper robot

### 4.1 Autonomous System

Due to the algorithm of decision making system, the experiment is done in the simulation, and it is evaluated with two metrics. The first one is the reward function of the algorithm for evaluating the result of the deep reinforcement learning algorithm, and the second one is the Robot Secure Rate (RSR) [44][45] for evaluating that whether the robot is secure or not. There are also some parameters we mentioned previously in the methodology part. How the drive changes through time is shown in the Table 4. 1, and reason for using exponential function for energy and safety and rest is to imitate the real condition. In Table 4. 2, it shows how the executed action affects the value of the drive,

and how much time the action costs.

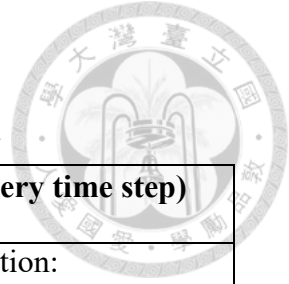


Table 4. 1 The table of drive change through time.

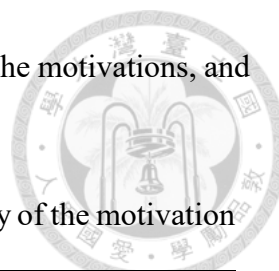
Drive	$\delta^i$ (the change for every time step)
Need of Energy (NEner)	Curve function: $y = f(x) = e^{x/8000} - 1; x > 0$ $\delta^i = f(x+1) - f(x)$
Need of Safety and Rest (NSaR)	Curve function: $y = f(x) = e^{x/6000} - 1; x > 0$ $\delta^i = f(x+1) - f(x)$
Need of Belonging (NBel)	0.0005
Need of Esteem (NEst)	0.0005
Need of self-actualization (NSA)	0.0005
Need of Curiosity (NCur)	0.0005

Table 4. 2 The table of the decreased value of drive due to the action, and the execution time of the action.

Drive	Action	$f_a(d_t^i, a^i)$	$\gamma$
NEner	Charge	-0.4	1000
NSaR	Rest	-0.15	100
NBel	Find people	-0.1	30
NEst	Sing/Dance	-0.1	30
NSA	Chat	-0.1	30
NCur	Ask question	-0.1	30
	Stand by	None	12

$f_a(d_t^i, a^i)$ : the decreased value of drive  $d_t^i$  based on the specific action  $a^i$  which is done

$\gamma$ : execution time steps for the action



In Table 4. 3, we show that the relation between the stimuli and the motivations, and the scale of how stimuli affect the intensity of the motivations.

Table 4. 3 The values in the table are  $f_{st}(\cdot)$  which affect the intensity of the motivation

<b>Stimulus</b> <b>Motivation</b>	SOver $st^1$	SHum $st^2$	SEnc $st^3$
MSur	0	0	0
MRes	$0.1st^1$	0	0
MRel	0	$0.05(1 - st^2)$	$0.05(1 - st^3)$
MAch	0	$0.05st^2$	$0.05(1 - st^3)$
MSI	0	$0.05st^2$	$0.05(1 - st^3)$
MAsk	0	$0.05st^2$	$0.05(1 - st^3)$

$f_{st}(\cdot)$ : the affected value for the motivation determined by the certain  $st^j$  and  $d_t^i$ .

There are also some parameters in the algorithm of dueling deep q network, and they are shown below. In training, only the internal rewards will be utilized. In addition, the weight parameters are initialized with random normal distribution, which the mean is 0 and the standard deviation is 0.02. The optimizer for updating is RMSProp algorithm.

Discounting factor  $\gamma$ : 0.99, Learning rate  $\eta$ : 0.0001

Initial epsilon value  $\epsilon$ : 1.0, Replay memory  $D_{exp}$  size: 50000

Replace time step C: 10000, Mini-batch size: 32

The parameters  $\kappa$ ,  $\lambda$ ,  $\nu$  are 1.0, -1.0, and 3 in the reward function. The desired range for MSur is [0.2,0.6], and others are all [0.4,0.6]. The reward can approach 650 after training 4000 epochs, and the Fig. 4. 2 shows the rewards, which is averaged for every 30 epochs, in the learning process. While testing for 100 epochs, the average reward

is 770.19. The reason that reward in testing is higher than in the training is due to the epsilon greedy strategy in training, which chooses random action for exploration.

Another evaluation metric is Robot Secure Rate (RSR). When a drive exceeds 0.9, it is considered as insecure. Thus, the metric representing the time ratio of robot being in secure robot is defined as:

$$RSR = \frac{1}{N} \sum_{t=1}^N se(t)$$

$$se(t) = \begin{cases} 1, & \text{if } \forall d_t^i \in D: d_t^i < 0.9 \\ 0, & \text{otherwise} \end{cases} \quad (4.1)$$

where N is the total time step,  $se(t)$  is an indicator for showing if the robot is secure at t time step

The higher value mean the robot is more secure. In the training process, the RSR can reach 88% in Fig. 4. 3, while RSR can reach 99.94% in testing. In the previous work [44][45], the RSR can reach 90% and 98%.

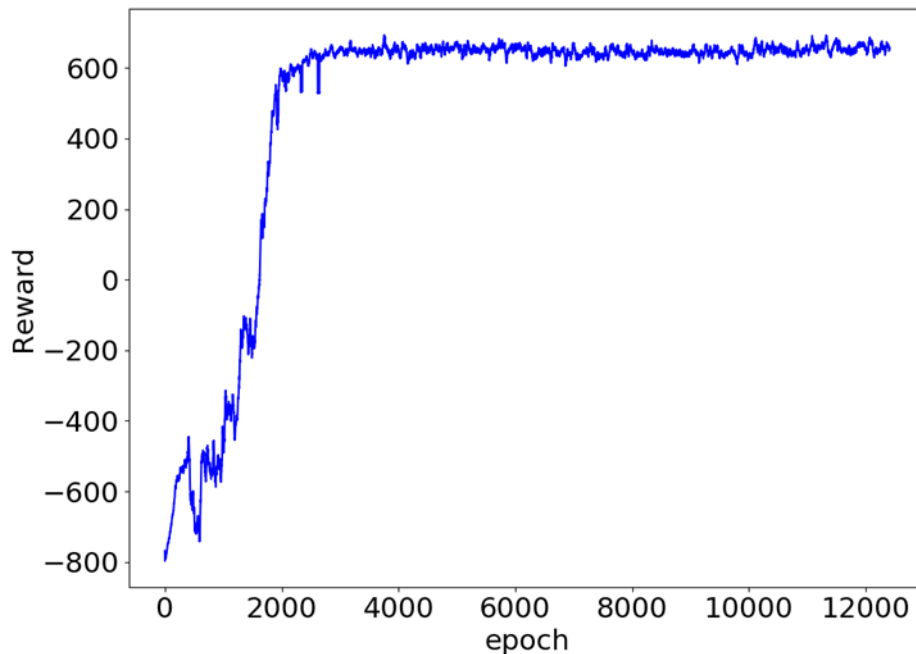


Fig. 4. 2 The reward of dueling deep q network.

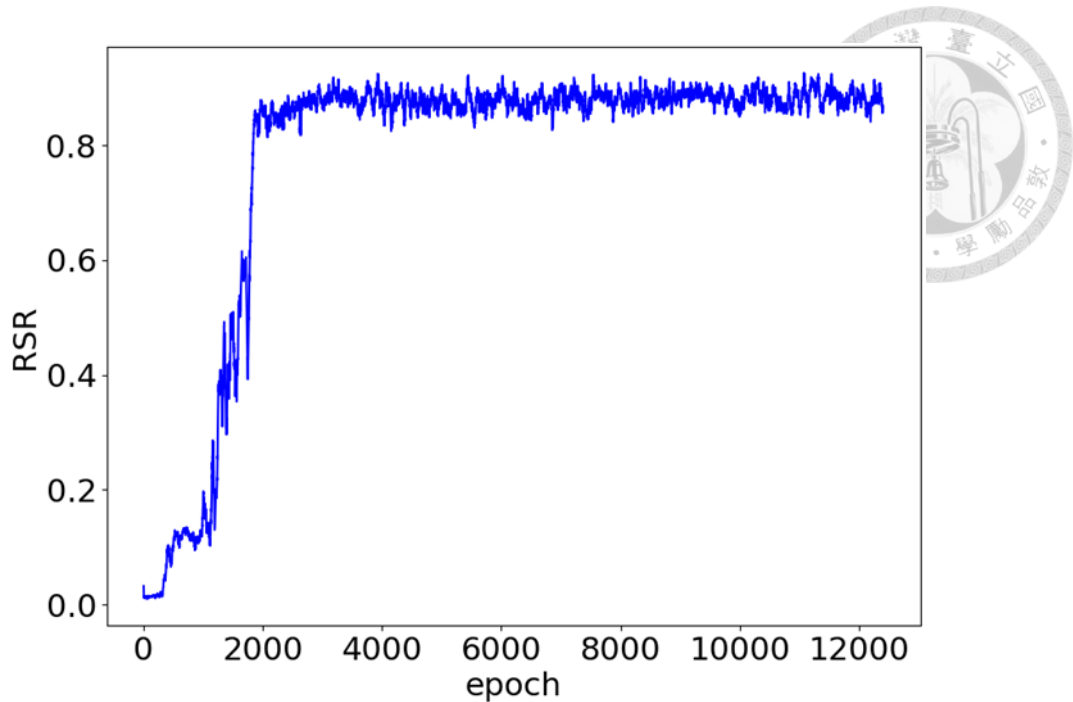


Fig. 4. 3 The secure rate in the training process.

We show the changing through time in the simulation in an epoch below. In Fig. 4. 4, we show the intensity of the motivations are changing steadily in a certain range. The reason that there are some flat lines in the changing is that the action has been done for a while, and other intensity will not change at that period of time. In Fig. 4. 5, we show the sequence of actions which were done in this epoch. It is very obvious that the charging has been done for only 3 times in this epoch, and most of the time the robot will take the action of standing by for waiting interactions. Also, sometimes the robot will go to interact with the human actively, like finding human, singing, dancing, asking questions, and social chatting in the figure. In the mechanism of the moods of robot, we will not consider the emotions of human in the simulation due to difficulty of analyzing the effect of emotions in simulation. The design in mechanism of moods in the experiment is that the weights in formula of each drive are 0.25 for NEner, 0.25 for NSaR, 0.2 for NBel, 0.1 for NEst, 0.1 for NSA, 0.1 for NCur, and 0.1 for the stimulus of encouragement. The changing for the value of moods is shown in Fig. 4. 6. If the threshold of setting the

positive mood is 0.43, we can find out that the robot will be in negative mood while it is ready for charging. Because the weights of design here are heavier on the values of  $N_{Ener}$  and  $N_{SaR}$ , the curve will be similar to the motivation of survival and motivation of rest in Fig. 4. 4.

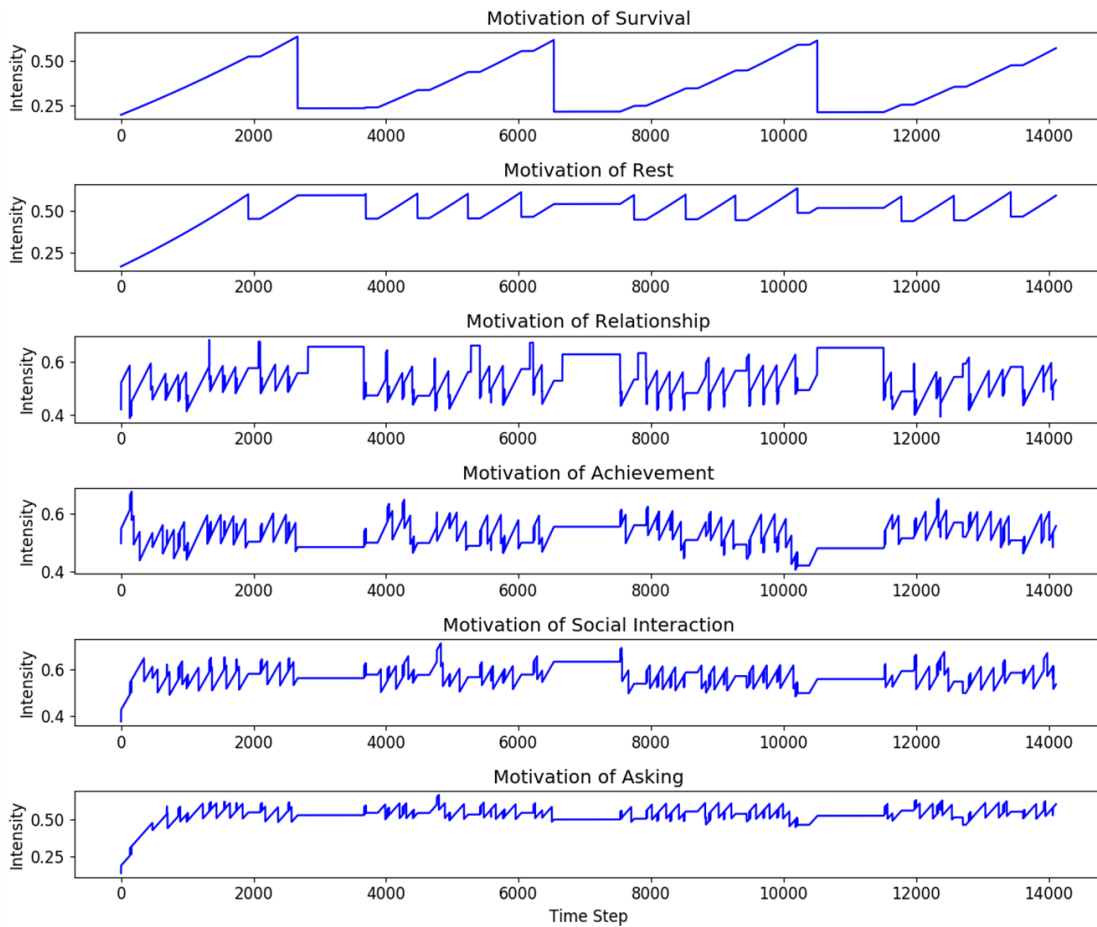
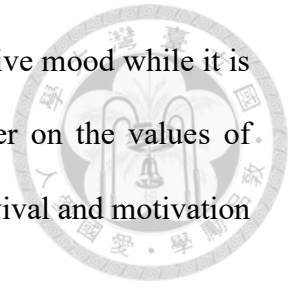


Fig. 4. 4 The intensity of motivations changes through time. The x-axis is the time step, and the y-axis is the intensity value.



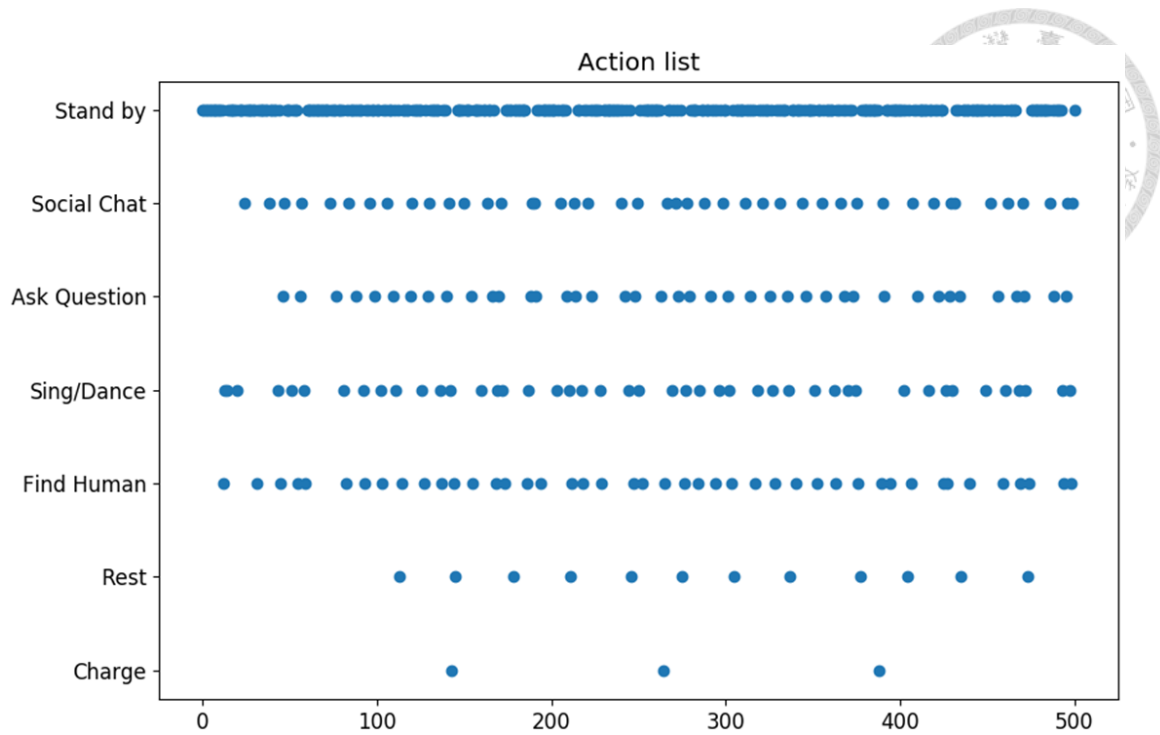


Fig. 4. 5 The sequence of actions in this epoch.

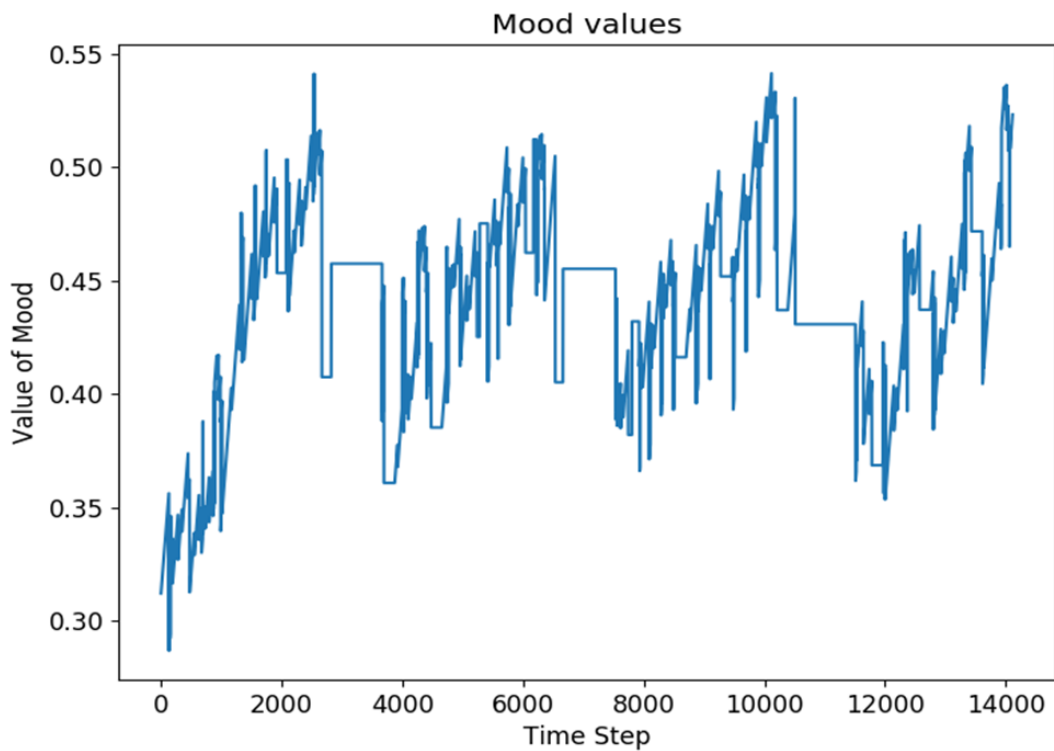


Fig. 4. 6 The value for moods.



## 4.2 Dialogue System

- **Chit-Chat bot**

For the chit-chat bot, both the encoder and generator are two layers of LSTM with 1024 neurons. It is trained on the Cornell dataset [94], Reddit dataset [95], and the handcrafted dataset from Papaya Conversational Dataset [96], and there are approximately 460K sentences. Perplexity is utilized to measure the accuracy of the model, and it can be explained as that how confused the model is for choosing the next word. The equation of perplexity is shown in (4. 2). Higher value of perplexity means the lower accuracy for predicting the next word, so the smaller value is what we expect. The model can get 1.14 perplexity here, and it is pretty good.

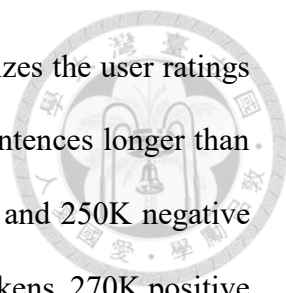
$$PP(S) = \sqrt[N]{\prod_{i=1}^N \frac{1}{p(w_i|w_1w_2 \dots w_{i-1})}} \quad (4. 2)$$

S: Sentence, N: length of sentence  
{ $w_1, w_2, \dots, w_N$ }: the tokens in the sentence.

- **Text Style Translation**

While training, the weights of each loss are different. The weights are 1 for reconstruction loss, 1 for generator loss in cross-alignment, and 0.3 for cycle-consistency loss. The loss of the generator part of adversarial training is trained three times, while the loss of the discriminator is trained one time. The temperature parameter will be decreased by a constant gain 0.99 for every 100 epochs. Moreover, the pre-trained word vector from genism google word2vec [98] for our word embedding matrix. The Adam optimizer [99] is used for the updating the parameters.

The experiment is evaluated on dataset which is provided by Shen *et al.* [65], and it



is the Yelp dataset [97]. It is a restaurant reviews dataset, which utilizes the user ratings to classify the positive reviews and the negative reviews, and the sentences longer than 15 words are eliminated. Finally, there are 350K positive sentences and 250K negative sentences, and the words appear less than 5 times will be unknown tokens. 270K positive sentences and 180K negative sentences are for training, and 76K positive sentences and 50K negative sentence for testing, while the rest of them are for validation. In order to evaluate the transferred sentences, a model-based evaluation is used. A pre-trained sentiment classifier, which is TextCNN, evaluates whether the sentence has the correct style, and it has 97.47% accuracy on the testing data. As a baseline, we compare against the model with Hu *et al.* [65] and Shen *et al.* [66], which is also provided in Shen *et al.* [66]. The result is shown in the Table 4. 4. While we are testing the code from Shen’s github, they revised the program at March and got the score of 83.94%. Our result has a better accuracy as compared with them. In addition, our method without style auto-encoder will get a lower accuracy, which represents that style auto-encoder really help to improve the accuracy by extracting and preserving better style latent representation.

Table 4. 4 The sentiment accuracy of transferred sentences evaluated by a pre-trained classifier.

Method	Accuracy
Hu <i>et al.</i> (2017)	83.5
Shen <i>et al.</i> (2017): Cross-aligned auto-encoder	78.4
Shen <i>et al.</i> (2018 github code)	83.94
Ours (W/O style auto-encoder)	82.05
Ours (W/ style auto-encoder)	84.65

We show some examples for transferring from positive sentences to negative sentences and negative sentences to positive sentences below:

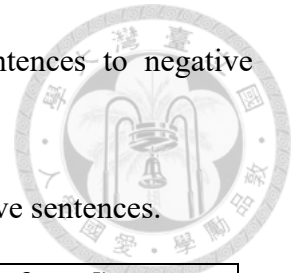
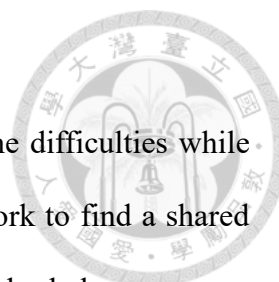


Table 4. 5 The positive sentences are transferred to the negative sentences.

<b>Positive (original)</b>	<b>Negative (transferred)</b>
Go there, eat, enjoy!	Go there, go elsewhere, do not eat!
Amazing food, very unique!	Amazing food, very poor quality!
I highly recommend this place – it is amazing.	I would not recommend this place but it is just ok.
They do an exceptional job here, the entire staff is professional and accommodating!	They don't want to mention the customer service is extremely unprofessional, rude !
She gets the work done and does an amazing job!	She did the job and she said that was not done well.

Table 4. 6 The negative sentences are transferred to the positive sentences.

<b>Negative (original)</b>	<b>Positive (transferred)</b>
This place is bad news!	This place is awesome!
I have never had pizza like in Chicago.	I have never had better pizza in Pittsburgh.
This place sucks.	This place is awesome.
The bad service of the waitresses make me dread going sometimes.	The customer service is always good job with me.
The water tasted bad, and worst of all, the food tasted horrible.	The food tasted good, and the best of course, great food.



- **Chit-Chat bot + Text Style Translation**

Due to not sharing the same dialogue dataset, it will have some difficulties while transferring the sentence for different styles. It will be our future work to find a shared dataset to train both models. However, we can still show basic examples below:

User speak/ Chit-Chat model	Transferred to negative	Transferred to positive
How was your day today? / Pretty good, how about you?	Pretty bad, but you can get?	Pretty good, you can like!
Do you feel good? / No, I feel very sad today.	No, I really felt disappointed with this place.	Totally a very nice experience with friends.
How was your lunch? / It is very delicious!	It is not very delicious!	It is very delicious!
How was your breakfast? / It is gross.	It is gross.	It is delicious.

We can find out that because they do not share the dataset, the text style translation model cannot reconstruct the sentence back well. However, the result shows that it still can transferred into the desired and correct style.

### 4.3 Visual Question Generation

The dimension of latent representation is 128, which it is 16 for categories and 8 for the categorical distribution. The temperature parameter is declined with ta constant gain 0.999 for every 100 epochs. The loss of the generator part of adversarial training will be trained for 10 times, while the loss of the discriminator is trained for once. The weights of all the loss are all 1, and the optimizer is Adam [99].

For the experiment, we combine the VQG dataset from Microsoft [100] and the VQA dataset [101] together, because the questions in VQG dataset are more natural than VQA dataset. However, the sentences in VQG dataset are not enough for the training, so VQA dataset is used. There are approximately 120K images and 3~5 sentences for each image in total. Evaluations of the generated sentences with the image are human evaluation, and there are three main evaluation metrics, such as “Diverse Meaning”, “Relatedness with the Image”, and “Relatedness with the Topic”. The reason that we use the human evaluation is due to the goal of generating diverse sentence for the image. There is no specific ground truth sentence for each image, and we should consider the meaning of the sentence, because it is weird for robot to ask similar sentences just by changing some prepositions in the sentence.

- **Diverse Meaning**

- Check if the meaning of the generated sentences is repetitive or not. For example, “how many people are there?” and “how many laptops are there?” are different meaning, while “how many people are there?” and “how many people are here?” are the same meaning.

$$DM(S) = \frac{1}{N} \sum_i^N dm(s^i), s^i \in S \quad (4.3)$$

$$dm(s^i) = \begin{cases} 1, & \text{if the meaning of } s^i \text{ different from others} \\ 0, & \text{otherwise} \end{cases}$$

where  $s^i$  is the generated sentence in the set of generated sentences  $S$ , and  $N$  is the number of sentences in the set.

- **Relatedness with the Image**

- Check if the generated sentence describes the image correctly or not. For example, the model sometimes generates the sentence: “is the dog sleeping?”

for a cat in the image, and it will be counted as not related to the image.

$$RI(S) = \frac{1}{N} \sum_i^N ri(s^i), \quad s^i \in S$$

$$ri(s^i) = \begin{cases} 1, & \text{if } s^i \text{ describes the image correctly} \\ 0, & \text{otherwise} \end{cases} \quad (4.4)$$

where  $s^i$  is the generated sentence in the set of generated sentences  $S$ , and  $N$  is the number of sentences in the set.

- **Relatedness with the Scene**

- Check if the generated sentence is related to the scene of the image and other sentences. For example, model generates the sentence: “is there pizza on the table?” for an image of hotpot restaurant. Although there is not pizza in the image, this sentence is still around the scene of food, and it will be considered that it is related to the topic.

$$RS(S) = \sum_i^N rs(s^i), \quad s^i \in S \quad (4.5)$$

$$rs(s^i) = \begin{cases} 1, & \text{if } s^i \text{ is related to the topic} \\ 0, & \text{otherwise} \end{cases}$$

where  $s^i$  is the generated sentence in the set of generated sentences  $S$ , and  $N$  is the number of sentences in the set.

We build our testing dataset by finding the images from internet, and some images are more common in daily life for robot to observe in real world. There are 50 testing images, and we use 100 sampling noises to generate the sentences. Finally, the model can generate 15.7 sentences in average, which means 785 sentences in total. The score of DM is 0.879, and it means that there are 87.9% sentences having different meaning for each image. The score of RI is 0.917 representing that 91.7 % of generated sentences can



accurately describe the image. The score of RS is 0.996, and it means that almost all the generated sentences are related to the scene of the image.

We show two examples below:



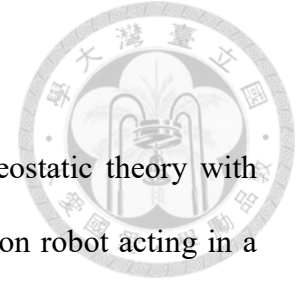
'how many people are here ?', 'how is the weather ?', 'are you wearing a shirt ?', 'are the people in the water ?', 'how many people are there ?', 'what color is yours shirt ?', 'where are the people ?', 'what are you doing ?', 'how many people are in the water ?', 'what color is the water ?', 'what are the people doing ?', 'are the people wearing wetsuits ?', 'is this a beach ?'





'what is on the table?', 'are there any children in the picture?', 'what color is the tablecloth?', 'how many people are in the picture?', 'how many candles are on the cake?', 'are the candles lit?', 'are these children?', 'is the woman wearing glasses?', 'is this a birthday cake?', 'what color is the cake?', 'what are the people eating?', 'are these people eating?', 'what is the girl eating?', 'what is the woman holding?', 'are the children happy?', 'what occasion is being celebrated?'

## Chapter 5 Conclusion



We proposed a novel autonomous system based on the homeostatic theory with inspiration from Maslow's hierarchy of needs for a social companion robot acting in a human manner, and built up the mechanism of moods for it. A robot with moods can improve the human-robot interaction, and moods are also going to be used in the dialogue system. As a social companion robot, dialogue system is indispensable, and there are two main functions. First, the chit-chat bot with the text style translation for transferring the positive or negative style, which is decided by the mechanism of the moods. Second, the QA bot for answering general questions is also built up. In addition, in order to make robot behaves in more humanlike way, we model the internal need of curiosity that is an important characteristic of human. The visual question generation module with visual perception system of robot are proposed to handle the curiosity need and to take initiative in the interaction.

Dueling Deep Q Network (Dueling DQN) is utilized for the decision making part of the autonomous system, and it can reach a more global optimal goal by modeling the continuous state, which is the intensity of motivations. For modeling different personality for the robot, the desired range of the intensity of motivation can be changed to make robot perform different dedication to interact with the human.

In the dialogue system, the chit-chat bot is a Seq2Seq with attention model, and the data driven based text style translation is proposed. In text style translation module, the pre-trained language model is used to speed up the convergence, and combine the cross-alignment training for aligning the sentence population of two styles. Moreover,  $N + 1$  discriminator is used for decreasing the parameters of the model and a style auto-encoder is used to extract more meaningful information of style. To preserve the content of the

sentence, cycle-consistency loss is also used for updating. In the QA bot, there are functions, such as information retrieval type, asking weather, asking time, asking news, asking related topics, asking for google map, asking for playing music, asking for taking photo.

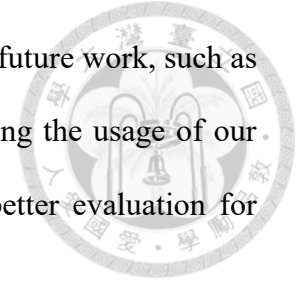


Deep learning based visual question generation deploys the categorical variational auto-encoder with Gumbel-Softmax for encoding the latent representation, and utilize the Professor-Forcing algorithm and continuous relaxation for discrete sampling process to avoid bias exposure. We also used the adversarial training to match sentences with the image better.

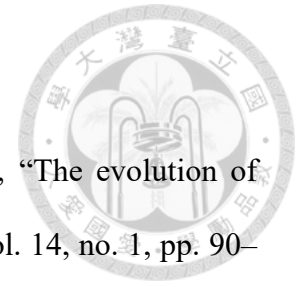
In the experiment of autonomous system, the Robot Secure Rate (RSR) can reach 99.94% in the testing, which means the robot can satisfy the internal needs well. For the text style translation module, we used a pre-trained sentiment classifier to determine the performance, and it has 84.65% accuracy to successfully transfer the style. The evaluation metrics of diverse meaning, relatedness of image, and relatedness of the topic are used for the visual question generation module, and it is done by the human evaluation. We can get 15.7 sentences for a single image in average, and get 87.9% sentences with different meaning. There are 91.7% generated sentences described the image correctly, and 99.6% of generated sentences are related to the same topic of the image.

There are future works for each part in this work. In autonomous system, changing the personality of the robot by tuning desired range of intensity of motivation should be in a more automated way. In the dialogue system, in order to combine the chit-chat bot and text style translation module, we should find a better dialogue dataset for training both of them. Otherwise, the text style translation is difficult to transfer the style of sentence outputted from the chit-chat bot. In the visual question generation, it can be developed as a visual understanding module by getting the answer back from the human.

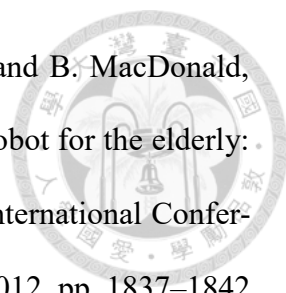
Most important of all, we will target at the companion ability in our future work, such as focusing on a specific human who needs companion and maximizing the usage of our mood mechanism. In addition, we should also try to develop a better evaluation for evaluating the users' feeling and the companion ability of the robot.

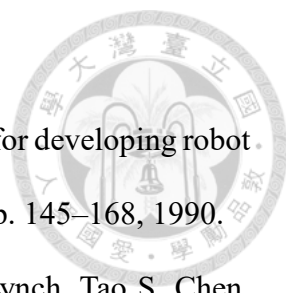



## REFERENCE



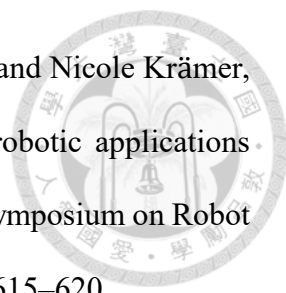
- [1] E. Garcia, M. A. Jimenez, P. G. De Santos, and M. Armada, "The evolution of robotics research," *IEEE Robotics & Automation Magazine*, vol. 14, no. 1, pp. 90–103, 2007.
- [2] "irobot: Your partner for a cleaner home," 2016, accessed: 10-July-2016. [Online]. Available: [www.irobot.com](http://www.irobot.com)
- [3] K. A. Wyrobek, E. H. Berger, H. M. Van der Loos, and J. K. Salisbury, "Towards a personal robotics development platform: Rationale and design of an intrinsically safe personal robot," in *Proceedings of IEEE International Conference on Robotics and Automation (ICRA)*, 2008, pp. 2165–2170.
- [4] S. Lauria, G. Bugmann, T. Kyriacou, J. Bos, and E. Klein, "Personal robot training via natural-language instructions," *IEEE Intelligent systems*, vol. 16, no. 3, pp. 38–45, 2001.
- [5] P. Elinas and J. J. Little, "Decision theoretic task coordination for a visually-guided interactive mobile robot," in *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2007, pp. 4108–4114.
- [6] K. Sakai, Y. Yasukawa, Y. Murase, S. Kanda, and N. Sawasaki, "Developing a service robot with communication abilities," in *Proceedings of IEEE International Workshop on Robot and Human Interactive Communication (RO-MAN)*, 2005, pp. 91–96.
- [7] C. Jayawardena, I. H. Kuo, U. Unger, A. Igic, R. Wong, C. I. Watson, R. Stafford, E. Broadbent, P. Tiwari, J. Warren, "Deployment of a service robot to help older people," in *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2010, pp. 5990–5995.

- 
- [8] C. Jayawardena, I. Kuo, C. Datta, R. Stafford, E. Broadbent, and B. MacDonald, “Design, implementation and field tests of a socially assistive robot for the elderly: Healthbot version 2,” in Proceedings of IEEE RAS & EMBS International Conference on Biomedical Robotics and Biomechatronics (BioRob), 2012, pp. 1837–1842
- [9] TIAN Guo-hui, LI Xiao-lei, ZHAO Shou-peng, LU Fei, "Research and development of intelligent space technology for home service robot." Journal of Shandong University (Engineering Science) 37.5 (2007): 53-59.
- [10] T. Breuer, G. R. G. Macedo, R. Hartanto, N. Hochgeschwender, D. Holz, F. Hegger, Z. Jin, C. Müller, J. Paulus, M. Reckhaus, “Johnny: An autonomous service robot for domestic environments,” Journal of intelligent & robotic systems, vol. 66, no. 1-2, pp. 245–272, 2012.
- [11] B. T. Clough, “Metrics, schmetrics! how the heck do you determine a uav’s autonomy anyway,” DTIC Document, Tech. Rep., 2002.
- [12] W.F. Truskowski, M.G. Hinchey, J.L. Rash, C.A. Rouff, “Autonomous and autonomic systems: A paradigm for future space exploration missions.” IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews) 36.3 (2006): 279-291.
- [13] G. A. Bekey, Autonomous robots: from biological inspiration to implementation and control. MIT press, 2005.
- [14] L. Cañamero, “Designing emotions for activity selection in autonomous agents,” Emotions in humans and artifacts, vol. 115, p. 148, 2003.
- [15] Walt Truskowski, Harold Hallock, Christopher Rouff, Jay Karlin, James Rash, Michael Hinchey, Roy Sterritt, “Autonomous and autonomic systems: with applications to NASA intelligent spacecraft operations and exploration systems.”, Springer Science & Business Media, 2009.

- 
- [16] Sony's Aibo robot dog: <https://aibo.sony.jp/en/>
- [17] T. L. Anderson and M. Donath, "Animal behavior as a paradigm for developing robot autonomy," *Robotics and Autonomous Systems*, vol. 6, no. 1, pp. 145–168, 1990.
- [18] Jonathan E. Clark, Daniel I. Goldman, Pei-Chun Lin, Goran Lynch, Tao S. Chen, Haldun Komsuoglu, Robert J. Full, and Daniel Koditschek, "Design of a Bio-inspired Dynamical Vertical Climbing Robot." *Robotics: Science and Systems*. Vol. 1. No. 2. 2007.
- [19] Guan-Horng Liu, Hou-Yi Lin, Huai-Yu Lin, Shao-Tuan Chen, Pei-Chun Lin, "A bio-inspired hopping kangaroo robot with an active tail." *Journal of Bionic Engineering* 11.4 (2014): 541-555.
- [20] Neal, Mark, and Jon Timmis. "Timidity: A useful mechanism for robot control?." *Informatica* 27.2 (2003): 197-204.
- [21] Marc Raibert, Kevin Blankespoor, Gabriel Nelson, Rob Playter, "Bigdog, the rough-terrain quadruped robot." *IFAC Proceedings Volumes* 41.2 (2008): 10822-10825.
- [22] Breazeal, Cynthia, and Rodney Brooks. "Robot emotion: A functional perspective." *Who needs emotions* (2005): 271-310.
- [23] Kwon, Dong-Soo, *et al.* "Emotion interaction system for a service robot." *Robot and Human interactive Communication, 2007. RO-MAN 2007. The 16th IEEE International Symposium on.* IEEE, 2007.
- [24] David Feil-Seifer and Maja J. Mataric. *Socially assistive robotics: Ethical issues related to technology.* *IEEE Robotics and Automation Magazine*, 18(1):24–31, 2011
- [25] Muhammad Ali, Samir Alili, Matthieu Warnier, and Rachid Alami. *An Architecture Supporting Proactive Robot Companion Behavior.* In *New Frontiers in Human-Robot Interaction at AISB*, 2009
- [26] Rida Gillani, Ali Nasir. "Incorporating artificial intelligence in shopping assistance

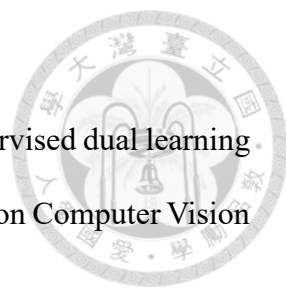
- 
- robot using Markov Decision Process." Intelligent Systems Engineering (ICISE), 2016 International Conference on. IEEE, 2016.
- [27] Cui Xuanyu, Liang Zhiwei, Yang Yongyi, Shen Ping, Wang Jiawen, Liu Haoran, Fan Kai. "Multi-robot collaboration based on Markov decision process in Robocup3D soccer simulation game." IEEE Control and Decision Conference (CCDC), 27th Chinese, 2015.
- [28] Hongtai Cheng, Heping Chen, Lina Hao, Wei Li. "Robot learning based on Partial Observable Markov Decision Process in unstructured environment." 2014 IEEE International Conference on Robotics and Automation (ICRA).
- [29] S. Omidshafiei, A.-a. Agha-mohammadi, C. Amato, and J. P. How, "Decentralized control of partially observable markov decision processes using belief space macro-actions," in Proceedings of IEEE International Conference on Robotics and Automation (ICRA), 2015, pp. 5962–5969.
- [30] S. Feyzabadi and S. Carpin, "Hcmdp: a hierarchical solution to constrained markov decision processes," in Proceedings of IEEE International Conference on Robotics and Automation (ICRA), 2015, pp. 3971–3978.
- [31] Pattie Maes. "The agent network architecture (ANA)." *Acm sigart bulletin* 2.4 (1991): 115-120.
- [32] Francisco Bellas, Richard J. Duro, Andrés Faiña, and Daniel Souto, "Multilevel darwinist brain (mdb): Artificial evolution in a cognitive architecture for real robots," *IEEE Transactions on Autonomous Mental Development*, vol. 2, no. 4, pp. 340–354, 2010.
- [33] Teruaki Ando and Masayosi Kanoh, "A self-sufficiency model using urge system," in Proceedings of IEEE International Conference on Fuzzy Systems (FUZZ), 2010, pp. 1–6.



- 
- [34] Jens Hoefinghoff, Astrid Rosenthal-von der Pütten, Josef Pauli and Nicole Krämer, “You and your robot companion—a framework for creating robotic applications usable by non- experts,” in Proceedings of IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN), 2015, pp. 615–620.
- [35] Jason R. Wilson, “Towards an affective robot capable of being a long-term companion,” in Proceedings of IEEE International Conference on Affective Computing and Intel- ligit Interaction (ACII), 2015, pp. 754–759.
- [36] W. B. Cannon, The wisdom of the body. WW Norton & Co, 1932.
- [37] R. C. Arkin, M. Fujita, T. Takagi, and R. Hasegawa, “An ethological and emotional basis for human–robot interaction,” *Robotics and Autonomous Systems*, vol. 42, no. 3, pp. 191–201, 2003.
- [38] C. L. Breazeal, Designing sociable robots. MIT press, 2004.
- [39] H.-L. Cao, P. G. Esteban, A. De Beir, R. Simut, G. Van de Perre, D. Lefeber, and B. Vanderborght, “Robee: A homeostatic-based social behavior controller for robots in human-robot interaction experiments,” in Proceedings of IEEE International Conference on Robotics and Biomimetics (ROBIO), 2014, pp. 516–521.
- [40] D. Cañamero, “Modeling motivations and emotions as a basis for intelligent behavior,” in Proceedings of the first International Conference on Autonomous Agents. ACM, 1997, pp. 148–155.
- [41] S. P. Gadanho, “Reinforcement learning in autonomous robots: an empirical investigation of the role of emotions,” 1999.
- [42] S. C. Gadanho and L. Custódio, “Asynchronous learning by emotions and cognition,” in Proceedings of the seventh International Conference on Simulation of Adap- tive Behavior on From Animals to Animats. MIT Press, 2002, pp. 224–225.
- [43] Á. Castro-González, M. Malfaz, and M. A. Salichs, “Selection of actions for an

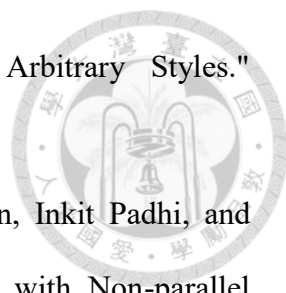
- 
- autonomous social robot,” in Proceedings of International Conference on Social Robotics, 2010, pp. 110–119.
- [44] Ching Lin, Rui-Hong Zhang, Shih-Huan Tseng, and Li-Chen Fu, “A homeostasis based decision making system on human-aware autonomous service robot.” International conference on Advanced Robotics and Intelligent Systems ( ARIS), 2017.
- [45] Chiao-Yu Yang, Ming-Jen Lu, Shih-Huan Tseng, and Li-Chen Fu, "A companion robot for daily care of elders based on homeostasis." 2017 56th Annual Conference of the IEEE Society of Instrument and Control Engineers of Japan (SICE).
- [46] Saul McLeod. "Maslow's hierarchy of needs." Simply Psychology 1 (2007).
- [47] Leon A. Gatys, Alexander S. Ecker, and Matthias Bethge. "Image style transfer using convolutional neural networks." 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR).
- [48] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, Yoshua Bengio. "Generative adversarial nets." Advances in neural information processing systems. 2014.
- [49] Ming-Yu Liu, and Oncel Tuzel. "Coupled generative adversarial networks." Advances in neural information processing systems. 2016.
- [50] Ming-Yu Liu, Thomas Breuel, and Jan Kautz. "Unsupervised image-to-image translation networks." Advances in Neural Information Processing Systems. 2017.
- [51] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A. Efros. "Unpaired image-to-image translation using cycle-consistent adversarial networks." IEEE International Conference on Computer Vision (ICCV), 2017.
- [52] Taeksoo Kim, Moonsu Cha, Hyunsoo Kim, Jung Kwon Lee and Jiwon Kim. "Learning to discover cross-domain relations with generative adversarial networks."

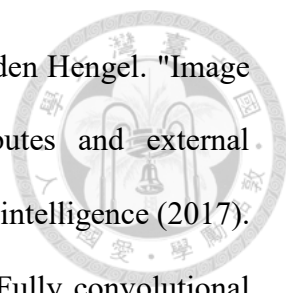
International Conference on Machine Learning (ICML) 2017.

- 
- [53] Zili Yi, Hao Zhang, Ping Tan, Minglun Gong. "Dualgan: Unsupervised dual learning for image-to-image translation." IEEE International Conference on Computer Vision (ICCV), 2017.
- [54] Jun-Yan Zhu, Philipp Krähenbühl, Eli Shechtman, and Alexei A. Efros. "Generative visual manipulation on the natural image manifold." European Conference on Computer Vision. Springer, Cham, 2016.
- [55] Peter F. Brown, John Cocke, Stephen A. Della Pietra, Vincent J. Della Pietra, Fredrick Jelinek, John D. Lafferty, Robert L. Mercer, and Paul S. Roossin. "A statistical approach to machine translation." Computational linguistics 16.2 (1990): 79-85.
- [56] Miller, George A. "WordNet: a lexical database for English." Communications of the ACM 38.11 (1995): 39-41.
- [57] Ganitkevitch, J., Van Durme, B., Callison-Burch, C.: PPDB: the paraphrase database. In: HLT-NAACL, pp. 758–764 (2013)
- [58] Peter F. Brown, Stephen A. Della Pietra, Vincent J. Della Pietra, Robert L. Mercer. "The mathematics of statistical machine translation: Parameter estimation." Computational linguistics 19.2 (1993): 263-311.
- [59] Richardson, Matthew, and Pedro Domingos. "Markov logic networks." Machine learning 62.1-2 (2006): 107-136.
- [60] Alexandrescu, Andrei, and Katrin Kirchhoff. "Graph-based learning for statistical machine translation." Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics.
- [61] Jessica Fidler, and Yoav Goldberg. "Controlling linguistic style aspects in neural


- language generation." arXiv preprint arXiv:1707.02633 (2017).
- [62] Mengqiao Han, Ou Wu, and Zhendong Niu. "Unsupervised Automatic Text Style Transfer Using LSTM." National CCF Conference on Natural Language Processing and Chinese Computing. Springer, Cham, 2017.
- [63] Zhenxin Fu, Xiaoye Tan, Nanyun Peng, Dongyan Zhao, and Rui Yan. "Style Transfer in Text: Exploration and Evaluation." In Proceeding of AACL, 2018.
- [64] Jonas Mueller, Tommi Jaakkola, and David Gifford. Sequence to better sequence: continuous revision of combinatorial structures. International Conference on Machine Learning (ICML), 2017.
- [65] Zhiting Hu, Zichao Yang, Xiaodan Liang, Ruslan Salakhutdinov, and Eric P. Xing. "Toward controlled generation of text." International Conference on Machine Learning. 2017.
- [66] Tianxiao Shen, Tao Lei, Regina Barzilay, and Tommi Jaakkola. "Style transfer from non-parallel text by cross-alignment." Advances in Neural Information Processing System, 2017.
- [67] Alex M Lamb, Anirudh Goyal ALIAS PARTH GOYAL, Ying Zhang, Saizheng Zhang, Aaron C Courville, and Yoshua Bengio. "Professor forcing: A new algorithm for training recurrent networks." In *Advances Neural Information Processing Systems*, pages 4601–4609, 2016.
- [68] Eric Jang, Shixiang Gu, and Ben Poole. "Categorical reparameterization with gumbel-softmax." arXiv preprint arXiv:1611.01144, 2016.
- [69] Guillaume Lample, Ludovic Denoyer, and Marc'Aurelio Ranzato. "Unsupervised Machine Translation Using Monolingual Corpora Only." International Conference on Learning Representations, 2018.
- [70] Yanpeng Zhao, Victoria W. Bi, Deng Cai, Xiaojiang Liu, Kewei Tu, Shuming Shi.

"Language Style Transfer from Non-Parallel Text with Arbitrary Styles."  
International Conference on Learning Representations, 2018.

- 
- [71] Igor Melnyk, Cicero Nogueira dos Santos, Kahini Wadhawan, Inkit Padhi, and Abhishek Kumar. "Improved Neural Text Attribute Transfer with Non-parallel Data." In Advances in Neural Information Processing Systems Workshop on Learning Disentangled Representations: from Perception to Control, 2017.
- [72] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. "Improved techniques for training gans." Advances in Neural Information Processing Systems. 2016.
- [73] Somak Aditya, Yezhou Ang, Chitta Baral, Cornelia Fermuller, and Yiannis Aloimonos. From images to sentences through scene description graphs using reasoning and knowledge. arXiv preprint arXiv:1511.03292, 2015.
- [74] A. Gatt and E. Reiter. Simplenlg: A realisation engine for practical applications. In Proceedings of the 12th European Workshop on Natural Language Generation, ENLG '09, pages 90–93, Stroudsburg, PA, USA, 2009. Association for Computational Linguistics.
- [75] Peng Wang, Qi Wu, Chunhua Shen, Anton Van Den Hengel, and Anthony Dick. Explicit knowledge-based reasoning for visual question answering. arXiv preprint arXiv:1511.02570, 2015.
- [76] Sören Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary Ives. "Dbpedia: A nucleus for a web of open data." The semantic web. Springer, Berlin, Heidelberg, 2007. 722-735.
- [77] Yi-Luen Wu, "Interactive Question-Posing System for Reminiscing about Personal Photos," M.S. thesis, Intelligent Robot and Automation Lab., National Taiwan Univ., Taiwan, 2017.

- 
- [78] Qi Wu, Chunhua Shen, Peng Wang, Anthony Dick, Anton van den Hengel. "Image captioning and visual question answering based on attributes and external knowledge." *IEEE transactions on pattern analysis and machine intelligence* (2017).
- [79] Justin Johnson, Andrej Karpathy, and Li Fei-Fei. "Densecap: Fully convolutional localization networks for dense captioning." *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2016.
- [80] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. "Faster r-cnn: Towards real-time object detection with region proposal networks." *Advances in neural information processing systems*. 2015.
- [81] Bo Dai, Sanja Fidler, Raquel Urtasun, and Dahua Lin. "Towards diverse and natural image descriptions via a conditional gan." *IEEE International Conference on Computer Vision (ICCV)*, 2017.
- [82] Unnat Jain, Ziyu Zhang, and Alexander Schwing. "Creativity: Generating diverse questions using variational autoencoders." *Computer Vision and Pattern Recognition*. Vol. 1. 2017.
- [83] Liwei Wang, Alexander Schwing, and Svetlana Lazebnik. "Diverse and Accurate Image Description Using a Variational Auto-Encoder with an Additive Gaussian Encoding Space." *Advances in Neural Information Processing Systems*. 2017.
- [84] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves, Martin Riedmiller, Andreas K. Fidjeland, Georg Ostrovski, Stig Petersen, Charles Beattie, Amir Sadik, Ioannis Antonoglou, Helen King, Dhharshan Kumaran, Daan Wierstra, Shane Legg, and Demis Hassabis. "Human-level control through deep reinforcement learning." *Nature*, 2014.
- [85] Long-Ji Lin. "Reinforcement learning for robots using neural networks." Technical

report, DTIC Document, 1993.

- 
- [86] Van Hasselt, Hado, Arthur Guez, and David Silver. "Deep Reinforcement Learning with Double Q-Learning." AAAI. Vol. 16. 2016.
- [87] Ziyu Wang, Tom Schaul, Matteo Hessel, Hado van Hasselt, Marc Lanctot, Nando de Freitas. "Dueling network architectures for deep reinforcement learning." arXiv preprint arXiv:1511.06581 (2015).
- [88] Matthew D. Zeiler and Rob Fergus. "Visualizing and understanding convolutional networks." In European Conference on Computer Vision, pages 818–833, 2014.
- [89] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556, 2014.
- [90] Sepp Hochreiter, and Schmidhuber Jürgen. "Long short-term memory." Neural computation 9.8 (1997): 1735-1780.
- [91] Kyunghyun Cho, Bart van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, Yoshua Bengio. "Learning phrase representations using RNN encoder-decoder for statistical machine translation." arXiv preprint arXiv:1406.1078 (2014).
- [92] Diederik P. Kingma, and Max Welling. "Auto-encoding variational bayes." arXiv preprint arXiv:1312.6114 (2013).
- [93] Yoon Kim. "Convolutional neural networks for sentence classification." arXiv preprint arXiv:1408.5882 (2014).
- [94] [https://www.cs.cornell.edu/~cristian/Cornell\\_Movie-Dialogs\\_Corpus.html](https://www.cs.cornell.edu/~cristian/Cornell_Movie-Dialogs_Corpus.html)
- [95] [https://www.reddit.com/r/datasets/comments/3bxlg7/i\\_have\\_every\\_publicly\\_available\\_reddit\\_comment/](https://www.reddit.com/r/datasets/comments/3bxlg7/i_have_every_publicly_available_reddit_comment/)
- [96] <https://github.com/bshao001/ChatLearner>
- [97] <https://www.yelp.com/dataset/challenge>

[98] <http://mccormickml.com/2016/04/12/googles-pretrained-word2vec-model-in-python/>

[99] Diederik P. Kingma, and Jimmy Ba. "Adam: A method for stochastic optimization."  
arXiv preprint arXiv:1412.6980 (2014).

[100] <https://www.microsoft.com/en-ca/download/details.aspx?id=53670>

[101] <http://www.visualqa.org>

[102] <https://www.softbankrobotics.com/emea/en/robots/pepper>

