A Survey on the Applications of Reinforcement Learning in Computer Vision and Natural Language Processing Introduction

Runxi Kang[1,a,*]

[1]Walton International College, Chengdu, Sichuan, 610045, China

[a]kangrx2023@163.com

Keywords: Reinforcement Learning, Computer Vision, Natural Language Processing

Abstract: Reinforcement Learning (RL) has achieved remarkable development in the history of artificial intelligence in the past decade. Especially from 2020 to 2025, its applications to CV and NLP have developed extremely quickly. Integrating deeply with deep learning techniques, RL has been applied to a series of challenging perception and decisions-making tasks. In CV, for example, RL is used to direct models to attend to important visual areas and to execute sequential perceptual tasks [1]. In NLP, it is employed to enhance text generation quality, train policies for dialog systems, and fine-tune large language models with reinforcement learning from human feedback (RLHF) to better match model output with human preference [2]. This survey comprehensively summarizes the major application areas and techniques of RL both in CV and NLP, shows some representative advances achieved in the past a few years, interprets the position and benefits of mainstream RL techniques applied to various tasks, explores existing challenges and shortages, and predicts future research trends.

1. Reinforcement Learning Applications to Computer Vision

The majority of tasks in computer vision involve sequential decision-making and interactive dynamics, ideal applications of reinforcement learning methods [3]. The applications of CV with RL can be generally categorized into object detection and image segmentation, object tracking, image generation and editing, visual navigation, and multi-agent simulated environment decision-making. In later sections, applications of RL to each task category are provided.

1.1 Reinforcement Learning in Object Detection and Image Segmentation

Conventional object detection generally implies laboriously searching all parts of an image. Thanks to reinforcement learning, however, agents are capable of learning to focus their efforts on significant informative areas, thus reducing redundant computation and accelerating detection speed [4]. Pirinen et al., for example, proposed region proposal networks with RL, where RL policies were employed to iteratively select regions to attain better detection performance [5]. This type of active object detection model learns a search region policy, thus efficiently localizing objects without having to scan a large number of regions. The RL agent adjusts its observation window's location and width based on the object detection reward, iteratively converging to target boundaries. Figure 1 is an example of active object localization via sequential bounding-box adjustments.

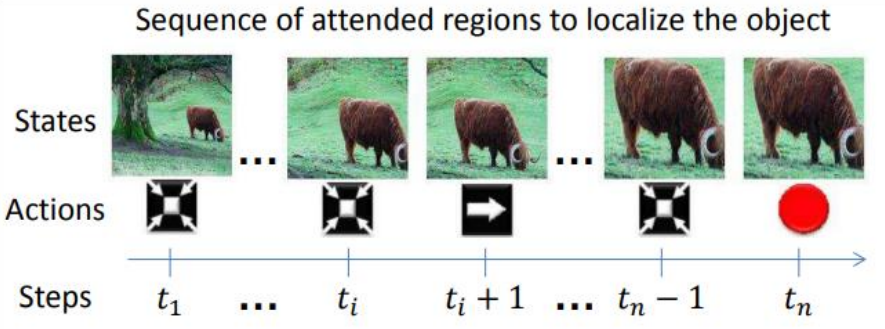

Figure 1: Active object localization via sequential bounding-box adjustments [1].

  DOI: 10.25236/icceme.2025.025

Hierarchical designs might also be constructed within multi-object detection tasks: low-level policy targets at detecting individual objects, and high-level policy puts together output from multiple detections to jointly handle sceneries with clutter. It's been demonstrated that those active detection approaches with their roots founded on RL are especially valuable within real-time applications as unmanned driving, where time-sensitive agents should react promptly to environmental changes and focus on potential danger objects to efficiently improve detection speed and accuracy.

Reinforcement learning also holds great potential in image segmentation tasks. As shown in Figure 2, the results of reinforcement learning applications in image segmentation are illustrated. The traditional segmentation algorithms are inclined to output, within a single iteration, a pixel-wise segmentation or a pixel classification of a full image. On the contrary, with the introduction of RL agents, models are enabled to iteratively refine segmentation boundaries. Based on feedback from the current segmentation result, e.g., boundary accuracy, the next step, e.g., whether to refine a given region or to shift to another region, might be determined by RL techniques. Pixel-level or region-growing approaches can be utilized by RL agents to achieve higher segmentation performance with fine grained granularity, especially well-fitted to applications where accuracy around object boundaries should be extremely high [6].

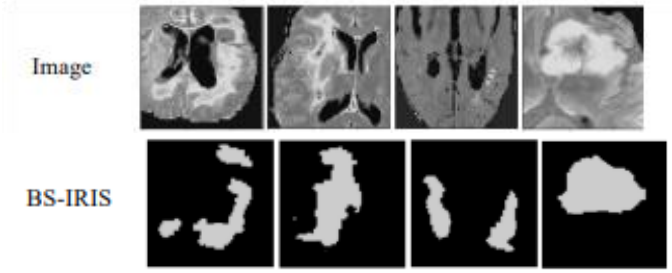

Figure 2: Results of Reinforcement Learning Applications in Image Segmentation.

Some research work utilizes hierarchical structures of rewards to guide agents to concentrate on global coherence and local precision of segmentation output, trying to overcome the disadvantageous trait of traditional methods of being prone to propagation of local error. For example, some works formalize segmentation as a Markov Decision Process (MDP) where, step by step, agents are rewarded by the precision of boundary pixels, and are gradually drawn to produce accurate object contours by reinforcement learning strategies. Experimentation proves that such types of active segmentation by RL-based methods are better capable to separate road and pedestrian boundaries from situations involving driving autonomously and are capable to improve precision related to organ segmentation within medical image perception. In particular, within visual perception applications involving object recognition and segmentation, RL fulfills a crucial role to guide decisions. In lieu of laborious scanning or thresholding processes, successive decisions from experienced agents are replaced so that visual information may be dealt with by a more focused, efficient manner [7].

1.2 Reinforcement Learning in Visual Tracking

Visual tracking consists of a model to stably recognize a target within a sequence from a provided video. It essentially undergoes a sequential decision-making procedure, thus extremely suitable with methods that include reinforcement learning [8]. One standard application of RL to tracking includes training an agent to learn policies to alter the camera view or observation window so stable tracking occurs even as the target or even scene transforms.

Traditional tracking algorithms can fail with challenging backgrounds where distractions are visually similar, but with RL agents, long-term gains can be summed to balance out short-term tracking errors with long-term stability, leading to more robust tracking policies. Song et al., for example, proposed combining a detector with a correlation filter tracker in a “Decision-Tracking Network (DTNet) where hierarchy-based reinforcement learning can be used to make decisions online: a high-level RL agent adapts between the two tracking modules intelligently according to scenarios” [9]. The agent transfers to using an adequate tracker according to current scene conditions—detectors are preferable with heavy appearance changes, and correlation filters are preferable with

unchanged target appearance. This switch policy with RL represents a significant improvement to robust tracking, and with DTNet, a number of publicly available benchmarks were obtained with then-state-of-the-art results. Figure 3 is an overview of the DTNet.

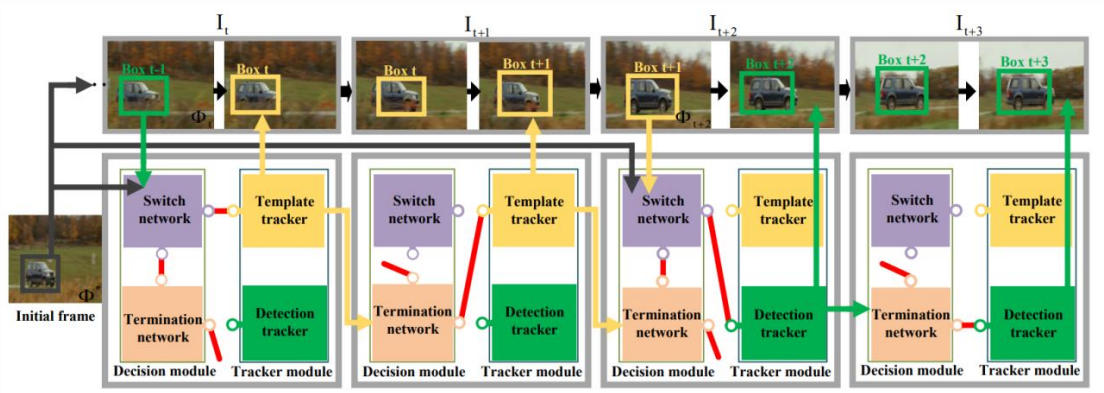

Figure 3: Overview of the DTNet.

In addition, RL has also been used to learn tracking policies end-to-end. For example, tracking can be viewed as a sequence decision-making problem, where, given a frame, an agent makes a translation or a scaling decision to change the position of a tracking window so that it can maintain the target in the center. The agent learns from reward signals guided by metrics such as Intersection over Union (IoU) or whether or not tracking was adequate. In such a scenario, RL optimizes a policy, utilizing a reward scheme to guide building a model that can overcome challenges like occlusion and distractions from the background to achieve more stable target tracking.

1.3 Reinforcement Learning in Image Generation and Editing

Classic approaches relied on Generative Adversarial Networks (GANs) or autoregressive models to produce images and complete missing areas. However, there was a line of work recently exploring reinforcement learning combined with image generation to steer output with reward signals to specific target distributions or evaluation metrics. In an RL context, image generation can be treated as a sequence decision problem: the agent starts with noise or a partly complete image and generate a complete image or complete missing areas by a sequence of actions, or, as in our case, by drawing a pixel patch or calling a filter. The agent obtains a reward from a discriminator's feedback, style consistency, or other metrics, and learns to balance visual quality with content realism [8].

This procedure can sidestep instabilities that are innate to standard GAN training by optimally tweaking output evaluation measures. In applications to image inpainting, an actor from an RL, say, can tabulate through diverse repair strategies and sequences—with first trying out diverse fill-in patterns, then being rewarded or punished by the world/environment and, with time, internalize strategies resulting in naturally looking, stylistically coherent completions.

Some even incorporate pre-defined perceptual rewards, e.g., a Structural Similarity Index (SSIM) or a confidence score from a face recognition net on a recovered image, as a RL reward signal to ensure both perceptual quality and semantic coherence [10].

Notably, even image–text retrieval and caption generation have been enhanced with RL. For instance, to produce image descriptions, models are fine-tuned with RL to their maximum evaluation score, such as BLEU or CIDEr. Though GANs and diffusion models continue to reign supreme across a vast suite of image generation tasks, RL represents a promising alternative with which to tailor model behavior with easily reconfigurable reward functions. One splendid example is self-critical sequence training procedure proposed by Rennie et al. in 2016, where they employed an optimization scheme developed from RL to seq2seq models to significantly improve caption quality [11].

Broadly, RL plays a dual function during image generation and editing as a policy executor and a quality assessor, guiding the generation process with evaluation functions to iteratively converge toward ideal visual output.

1.4 Reinforcement Learning in Object Detection and Image Segmentation

Visual navigation consists of an agent navigating through visual perception to, amongst other things, move, localise itself, and complete goal-specific tasks within a world environment—such as going to a particular target in a virtual room. This type of task falls into the reinforcement learning paradigm naturally: the agent continually makes decisions to adopt movement directions and learns to navigate by being rewarded by the world.

Goal-based visual navigation is a common case: the agent is given a visual or semantic map of a target and seeks to locate it in an unknown space. Zhu et al. expressed this problem as a Markov Decision Process (MDP) in a flagship paper, where the agent's state is first-person visual perception, and action space includes actions such as going forward and turning. The agent receives a positive reward if it moves closer to a target. The agent learns to iteratively narrow down to the target's location by visual features with deep reinforcement learning. This enables unmanned navigation through novel indoor environments [12]. Figure 4 explains the target-driven visual navigation via deep Reinforcement learning.

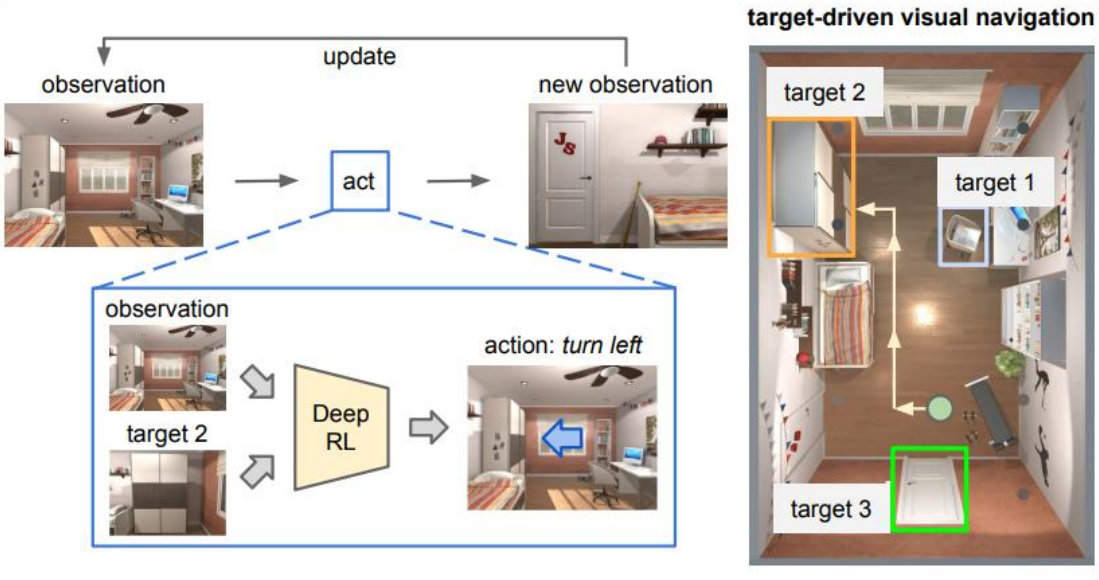

Figure 4: Target-Driven Visual Navigation via Deep Reinforcement Learning.

Subsequent work has employed RL to solve more complex navigation tasks. In vision-and-language navigation (VLN) tasks, for example, agents must navigate with language instructions. Anderson et al. developed the Room-to-Room dataset, where pre-training with imitation learning and fine-tuning with RL are combined to get agents to map visual observation better to instruction semantics and reach target locations with success [13].

RL has also been applied to train vehicle navigation policies within autonomous driving simulators, with particular success achieving improved obstacle avoidance and path planning. Certain methods utilize a combination between RL and imitation learning, where agents are first taught based on examples from human driving demonstrations, then they hone their policies through RL to better cope with challenging road scenarios and scarce reward indicators.

RL can be employed to gain a decision-planning capability for visual navigation: in contrast to classical planning strategies, which require explicit maps of environments, RL agents can plan actions from trial-and-error experiences even during mapless or dynamically varying environments.

Recent work has tried to unify large language models (LLMs) with RL to achieve navigation tasks—to give an example, using LLMs to generate high-level plans, with their execution by low-level RL controllers, so that hard tasks could be divided and completed.

RL hence assumes a central role in learning navigation tactics, so that perception and action can be connected in a direct way, and effective navigation behaviors can be learned interactively.

1.5 Reinforcement Learning in Multi-Agent Simulation

Multi-agent simulation involves several interacting decisions by several agents within a shared environment, including multi-vehicle coordination within autonomous vehicle simulators or game AIs with adversarial collaboration. Because every agent's policy influences others, their environment becomes very non-stationary, posing additional challenges to reinforcement learning. In around 2020, progress was claimed in multi-agent reinforcement learning (MARL) suggesting that RL could reach or surpass human-level competence within complex multi-agent environments. One instance was DeepMind's AlphaStar, where RL agents were trained by self-play within the real-time strategy game StarCraft II, to achieve a competence similar to professional leaders [14]. AlphaStar employed a "multi-strategy league" protocol to train, where by self-play, agents evolved and included imitation learning with population-based policy gradient methods, coaxing strategies to exploitable form. The experiments showed that with deep RL-enabled multi-agent systems, strategies developed by humans over long time scales could be matched.

Another celebrated example includes OpenAI Five, where MARL-learned policies defeated world-leading human teams by a significant margin in a game of Dota 2 [15]. The demonstrations illustrated RL's strong search and learning capacities within high-dimensional, multi-agent action spaces. Throughout 2020–2022, a number of MARL algorithms were introduced. QMIX, for instance, invented the concept of value decomposition for cooperative environments, breaking down global team rewards into per-agent value functions to effectively solve the credit assignment problem [16]. Qatten and QPLEX, among additional follow-up research, further enhanced value decomposition strategy expressiveness [17]. In adversarial environments, policy gradient approaches with self-play strategies—e.g., policy space response oracle or adversarial training—demonstrated robustness with varying opponents. Yu et al. remarked that Proximal Policy Optimization (PPO), a baseline single-agent RL algorithm, surprisingly obtained robust cooperative multi-agent task performance with sufficient adaptation [18]. The finding suggests that even single-agent RL policies could remain competitive with challenging MARL scenarios, providing new considerations toward algorithmic selection.

Generally, RL in multi-agent simulation applications to guide co-evolutionary trends of coordinated or competitive strategies with long-term self-play, to solve large games where designing strategies by hand is difficult. The methods are gradually gaining real-world applications in fields such as multi-robot coordination, cooperative motion of autonomous vehicles, and optimization of network traffic flow. In summary, RL gives multi-agent systems autonomous competition and negotiation, facilitating effective virtual environment collective intelligence.

2. Reinforcement Learning Applications in Natural Language Processing

Similar to CV, the NLP field also has a number of sequential decision-making or policy optimization tasks. While classical supervised learning performs well over a very broad set of language tasks, it very often struggles with tasks where evaluation measures are non-differentiable or long-horizon planning is required. Reinforcement learning gives a paradigm through which models are capable of interacting with their environment by trial and error and directly optimise end objectives. In recent years, deploying RL to NLP has comprised primarily of: optimisation and generation of text, learnt strategies of dialogue systems, policy optimisation during machine translation, and fine-tuning alignment to large language models. The former are detailed below.

2.1 Reinforcement Learning for Text Generation and Language Model Optimization

Sequence generation problems, such as summarization, functional description generation, or story completion, are generally formulated as sequence generation tasks. The tasks generally suffer from exposure bias and loss-evaluation metric mismatches. Sequence-level rewards address these issues, and reinforcement learning supplies them. One among the pioneering early work is that of the MIXER algorithm by Ranzato et al. (ICLR 2016), which transferred RNN-based text generation from word-level cross-entropy to policy gradient optimization of full-sequence evaluation measures. The

research demonstrated that it's possible to use RL signals to train generation models [19].

Then, Paulus et al. attempted to use RL for summarization, building a reward function from ROUGE scores, thus training generation models readable but also better-aligned with reference summaries. In experiments, they proved summarization models finetuned by RL significantly outperformed those supervised-only-trained ones [20]. Another research line transplanted GANs to text generation: the generator is a policy, discriminator feedback are rewards. The generator then becomes learned by RL to output more indistinguishable from real text. It improves text coherence to some extent, although GANs remain unstable to use with language [21].

RL has also been used to promote diversity and enforce stylistic constraints when generating texts. Inverse reinforcement learning (IRL), for example, has been used to learn dialogue or story generation with implicit reward functions encouraging more contentful and stylistically diverse generation. Shi et al. used IRL to guide models to generate more diverse sentences with higher diversity scores than previous methods [22]. Additionally, some studies use human preference models as rewards to fine-tune generation systems with RL so that output better meets human tastes or expectations, especially with open-domain dialogue systems.

With large models like GPT-3 being created, correspondingly matching model output to human intent becomes a major problem. Supervised fine-tuning alone generally doesn't succeed to prevent models from providing unhelpful or inappropriate output. To counteract, OpenAI and others created reinforcement learning from human feedback (RLHF). The approach involves first having a reward model learn from human-labeled data to provide a numerical measure of output quality. The policy optimization algorithms are then used to fine-tune the language model to maximize a reward score [18][23][24].

InstructGPT is a pioneering implementation of this approach. In a 2022 NeurIPS paper, Ouyang et al. illustrated training a reward model from human preference data and fine-tuning GPT-3 with PPO to enable it to execute instructions robustly while producing useful and safe output [11]. Importantly, even the 1.3B-parameter InstructGPT outperformed the 175B unaligned GPT-3 model on human preference ratings in a suite of instruction-following tasks. This result demonstrated that RL could effectively endow large language models with human values and task understanding. Similarly, work by Stiennon et al. showed that applying learned reward models from human preferences with RL significantly enhanced summarization quality [25]. Microsoft, DeepMind, and others then developed dialogue systems with RLHF. RL has thus evolved to be a core technique to align large language models.

To ensure stable optimization over high-dimensional parameter spaces, practitioners typically use robust algorithms such as PPO and add KL-divergence penalties to prevent abrupt policy shifts [24]. The tradeoff maintains diversity and factual accuracy and optimizes reward-model scores. We can steer language models to desirable objectives—with properly defined reward functions—such as increasing factuality, coherence, politeness, or decreasing harmful output, which has important practical value for NLP applications.

2.2 Reinforcement Learning in Dialogue Systems

Both open-domain chatbots and task-based dialogue systems are agents talking with users, and reinforcement learning is a powerful tool to optimize dialogue policy. In task-based systems, the model tries to complete user goals through multi-turn conversation. This can be modeled as an MDP: dialogue state includes conversation history and user intention, actions are system questions or utterances, and task completion sets up rewards. The vast majority of conventional systems utilize rule-based or supervised modules, but RL gives end-to-end policy acquisition where, through trial and error, it explores useful strategies for response. The scenario where methods from Deep Dyna-Q, e.g., see the dialogue manager as an agent obtained with a user simulator by interacting with it, opens up, step by step, for an agent to hone policy and complete tasks with smaller turns [26].

In more recent times, methods such as hierarchical reinforcement learning (HRL) and inverse reinforcement learning (IRL) have been used to further improve data efficiency and policy safety. Hou et al. used adversarial IRL to build more fine-grained formulations of a reward signal by

decomposing a reward into a multi-level structure with dialog-act-level, slot-filling, and domain-level ones. This addresses the issue of sparse rewards and increases policy interpretability. The experiments verify that by doing so, a multi-level structure significantly expands task completion rates and optimization speed with dialogue systems [27].

For open-domain chat, response quality assessment becomes even more difficult, since it should entail contextual relevance, informativeness, politeness, and interactivity. RL has been applied in two main ways:

Using user feedback as rewards to tune models toward user-preferred actions, and pretraining a reward model to predict human judgments, prompting the model to generate more informative and coherent responses.

Li et al. were the first to undertake a study where they proposed to evaluate chatbot response quality with their heuristic measures and treat them as reward signals to tune the model by using RL. The result showed bots trained by RL were more likely to question and exclude dry responses, leading to higher user engagement [28].

More recently, the success with chat models of RLHF has cemented the important role for open-domain dialogue of RL. Using human preference function learning and model optimization to match, dialogue assistants are better able to provide useful information with the safe and controlled use of language [11]. It aids with overcoming supervised learning by itself challenges, such as offense or bias generation to match with training data, which can be effectively negated by RL fine-tuning. Also, organizations like Anthropic have tried to train with AI-provided feedback to reduce costs and increase scalability.

Overall, reinforcement learning gives dialogue systems ongoing improvement capabilities to refine strategies based on outcomes from interaction, rather than with constant data alone. This has important real-world applications to robustify dialogue-based AI and end-user satisfaction.

2.3 Reinforcement Learning in Machine Translation

Neural machine translation (NMT) became the de-facto standard method within machine translation, but standard training procedures typically employ cross-entropy loss acting over word sequence levels, which doesn't perfectly align with evaluation measures and is vulnerable to exposure bias. Reinforcement learning opens a way around by providing sequence-level optimization. Since 2016, Ranzato et al. even suggested taking BLEU scores as direct rewards and policy gradient-like techniques to fine-tune translation models to BLEU maxima [19]. Although BLEU is noisy within high-dimensional parametric spaces, work demonstrated that there was potential to apply RL to attain improved translation quality. Follow-up work, such as Google's Minimum Risk Training, incorporated evaluation measures into the objective function by using RL-like optimization to achieve lower expected risk and hence higher scores compared to those with MLE-based methods [29].

More conspicuous applications of RL to translation have been from unsupervised and interactive settings. He et al. proposed the dual learning paradigm, where bilingual translation between two languages are considered two agents performing reciprocal tasks, where each translation serves as feedback to the other one [30]. That is, specifically, an English-to-Chinese system translates a sentence, back-translated into English and compared with a target sentence. The difference serves as a reward signal to update a target system; correspondingly, a similar procedure repeats with Chinese-to-English. This two directional RL setup enables learning from monolingual data without large parallel data, separately updating both directions of translation. Dual learning dramatically reduces reliance on high-quality parallel data and serves as a milestone toward unsupervised machine translation.

RL has been used also in interactive translation tasks such as simultaneous translation. The task was modeled by Grissom et al. as an MDP, where the agent needs to determine whether to wait for additional source words or to produce a translation right away [31]. With RL training, agents find a balance between translation latency and translation quality. In case of large word-order disparities between languages, the model might learn to look ahead to sentence-final verbs, minimizing delays.

Further, RL can be employed to refine strategies for interactive translation's human-machine interface. For example, models can be aided to decide to ask users to clarify ambiguous words to

attain improved translation with fewer queries [32]. Overall, reinforcement learning provides sequence-level optimization and interactive learning within machine translation. In place of employing fixed correspondences with only static data to learn, models are equipped to construct dynamic translation strategies from interaction with themselves or their world. This broadens MT's applicability to include low-resource, unsupervised, multilingual, and real-time conditions.

The principle of self-play that gives a boost to AlphaGo is also transferred to translation metaphorically—multilingual AIs could be trained to interact with each other in the future. Although its flaws with NMT, such as high variance and unstable training, are still present with RL, its narrow, focused achievement with niche tasks opens new ways to the field.

2.4 RLHF for Large Language Model Finetuning

Large language models (LLMs) such as GPT-3 possess great generation capacity, but they also produce output that can be unlike human intent, including incorrect or toxic content. In an effort to fine-tune such models to be better-behaved, scholars have been trying reinforcement learning from human feedbacks (RLHF) as a tool to get better-behaved models [23]. There are typically three stages to RLHF:

Training a reward model from human feedback data to evaluate LLM output;

Taking a pre-trained LLM as a policy initialization, then using RL algorithms to adjust model parameters to get higher reward scores;

Obtaining new data cyclically to update both the reward model and policy, resulting in repeated refinement [23][33].

OpenAI's InstructGPT is a pioneering work in RLHF. In their 2022 NeurIPS paper, Ouyang et al. defined the process: training a reward model from human preference data and then fine-tuning GPT-3 with PPO to improve compliance with instructions and to produce useful, safe output. Surprisingly, the 1.3B-parameter InstructGPT model even outperformed the 175B-unaligned GPT-3 on human preference ratings across a number of instruction-following tasks [11].

RLHF is also being used extensively to train dialogue systems and chatbots to be compliant with ethics and useful. Anthropic created a “constitutional AI” methodology, using a system of alignment principles to programmatically build feedback favoring RLHF training, to speed up alignment [34]. Those are further extensions of applications of RL to fine-tune LLMs.

Reinforcement learning is relevant to LLM fine-tuning as a result of human preferences being difficult to translate to loss functions that are explicit. However, by using comparison data to train a reward model and then optimising behaviour with RL, it introduces a value-driven component over-and-above regular maximum likelihood training. High-dimensional optimisation with RL has challenges such as mode collapse or even reward hacking. Researchers have incorporated protection measures, such as adversarial metrics within the reward model to detect occasions where output games the scoring system, or alternatives such as direct preference optimisation that look to capture the benefit of RLHF without full-blown RL [35].

However, RLHF remains among the most effective paradigms. OpenAI's GPT-4 technical report, for example, remarks that supervised pre-training then reinforcement with human feedback still represents a key step to advancing capability as well as alignment [36]. As models increase in size and variety of application, correspondingly, RL will be central to managing and optimising their behaviour so that robust AI systems behave as intended by humans.

3. Summary of Major RL Algorithms Applied to CV and NLP Tasks

Across computer vision (CV) and natural language processing (NLP), various reinforcement learning (RL) algorithms have demonstrated strengths in different environments. Selecting the right algorithm can significantly impact task performance. The most widely adopted categories include value-based, policy gradient-based, and actor-critic-based methods. Additionally, contextual bandit algorithms are used in certain real-time or low-latency decision tasks. Below is a summarized breakdown of major algorithms and their application suitability:

Deep Q-Network (DQN) [37]: a typical value-based algorithm that specialises in problems in

discrete action spaces.DQN evaluates the long-term payoff of each action through a Q-approximation, and is suitable for tasks in which the number of actions is limited and explicit rewards can be easily defined. In CV, DQN is successfully used in discrete decision-making contexts, such as region selection in images, classification decisions, etc. It has been shown that DQN can achieve good results by discretising actions in image classification and detection. In NLP, DQN and its extensions are sometimes used for discrete selection of dialogue strategies. However, DQN performs poorly on high-dimensional continuous spaces and requires techniques such as empirical playback and goal networks to ensure training stability. Overall, DQNs are suitable for scenarios where both states and actions are small-scale and the reward design is explicit, such as simple text game decision-making or strategy selection for certain classification/retrieval tasks.

Policy Gradient methods [24][38]: algorithms that optimise directly in terms of policies, often used for problems with continuous and high-dimensional actions. The policy gradient approach does not need to maintain a value table and is more flexible in modelling the policies.REINFORCE is a Monte Carlo policy gradient, which is simple in concept but has a high variance, and is usually combined with Baseline to reduce the variance.Proximal Policy Optimization (PPO) is one of the most popular policy gradient algorithms in recent years.PPO ensures that the policy does not go to extremes at each iteration by clipping policy updates, thus ensuring that the policy does not go to extremes at each iteration. PPO strikes a balance between performance and stability by ensuring that the policy does not go to extremes in each iteration. In CV, PPO is widely used in continuous control tasks and situations that require stable training, such as robot navigation and target tracking. For example, in visual navigation and path planning, the stable update of PPO enables the intelligent to reliably learn the travelling strategy without losing control due to occasional high-reward sequences. Similarly in NLP, PPO has become the algorithm of choice for RLHF due to its stability, and is used to fine-tune large language models so that policy updates can improve preference scores without breaking language fluency. The advantage of the strategy gradient class of algorithms is that the goal is direct: they can be optimised directly for the final evaluation metrics, avoiding value function error transfer. In addition, they are naturally suited to continuous actions and perform well in tasks requiring fine-grained control.

Actor-Critic methods [39][40][41]: these algorithms combine policy gradient and value function evaluation, which are complementary to each other.Advantage Actor-Critic (A3C) improves the training efficiency through asynchronous multi-threading, and has had a track record in environments such as Atari games. Its advantages are stability and fast convergence, which makes it suitable for real-time scenarios that require fast responses.Deep Deterministic Policy Gradient (DDPG) and Soft Actor-Critic (SAC) are excellent algorithms for continuous action spaces.DDPG outputs continuous actions in terms of actors, critically evaluates the Q-value, and is capable of achieve good results in environments such as robot control, but is relatively under-explored.SAC introduces an entropy regularity term to encourage the policy to be sufficiently stochastic to promote exploration, and performs well on high-dimensional continuous control tasks. For continuous decision-making problems such as robot operation and unmanned driving in CV, SAC and DDPG are often employed. For example, steering and acceleration continuous control in autonomous driving, smooth and safe strategies can be obtained with SAC because its maximum entropy framework makes intelligences less likely to fall into suboptimal deterministic strategies. In NLP, actor-critic is mainly used for text generation that requires long-term planning, e.g., some studies have tried to optimise the coherence of long texts with actor-critic: the actor generates sentences, and the criterion scores and guides the actor to adjust according to the coherence of the whole text. This has exploratory implications for tasks such as story generation and script generation. However, NLP sequences are discrete and long compared to CV control, and actor-critic applications need to overcome the problems of high variance and credit assignment.

Bandit algorithms [42]: multi-armed slot machines and their contextual extensions are useful in some scenarios of NLP. For example, intelligent news recommendation and advertisement placement can be modelled as contextual bandit, where one article recommendation is selected at a time based on the user and the content, with click or no click as a reward. In NLP research, Ranzato et al. have

considered machine translation as a bandit, where the BLEU score of the translation result for each sentence is used as a one-time reward, with feedback only on the overall quality and no word-by-word feedback. Bandit algorithms such as EXP3, LinUCB, etc. can be used for the optimisation of such one-time decisions. In dialogue system training, a bandit optimization strategy can also be used if the influence of multiple rounds is not taken into account, and whether the problem is successfully solved in each round of repetition is taken as a bandit reward. However, the bandit method cannot directly deal with the problem of long-term rewards, so bandit is not as effective as full RL in tasks that require sequential decision-making, such as multi-round dialogue and long text generation. Overall, bandit methods are suitable for single-step decisions or repeated decisions without dependencies. In the field of NLP, such scenarios include dynamic summary length selection, selection of the next question in interactive Q&A, and so on.

In summary, different RL algorithms have their own strengths, and should be chosen according to the task characteristics: DQN for discrete short sequences, PPO/DDPG/SAC for continuous high dimensionality, PPO for stable improvement of large models, and SAC for high exploration. In practice, it is also common to combine multiple algorithms, for example, warming up the strategy with imitation learning and then fine-tuning it with actor-critic, or combining the gradient of the strategy with Monte Carlo Tree search to improve stability. Table 1 compares the performance of some algorithms in different tasks:

Table 1: illustrates the principle of aligning different RL algorithms with specific task characteristics—applying the right tool to the right problem.

Task Type	Typical Scenario	Recommended Algorithm	Key Advantages
Discrete Decision Making	Object detection, image segmentation	DQN / A3C	- DQN: Direct modeling of discrete action spaces; stable convergence. - A3C: Asynchronous updates accelerate training and reduce variance.
Continuous Control	Visual navigation, target tracking, robotic control	PPO / SAC	- PPO: Clipped updates ensure stable policy improvement; well-suited for fine-tuning in complex environments. - SAC: Maximum entropy framework promotes exploration and yields robust strategies in high-dimensional continuous spaces.
High-Dimensional Discrete Sequence Generation	Text generation, dialogue response generation	PPO + Actor-Critic	- PPO: Directly optimizes sequence-level metrics (e.g., BLEU, CIDEr) with high stability and efficiency. - Actor-Critic: Combines value estimation to reduce variance, helping balance output quality and diversity.
Few-Shot or Pretraining, Fine-Tuning	Multi-task learning, cross-domain adaptation	Imitation Learning, Actor-Critic Fine-Tuning	- Imitation Learning: Quickly acquires an initial policy. - Actor-Critic: Fine-tunes via limited interactions for task-specific adaptation.
Search and Planning Integration	Board games, strategic decision-making	Policy Gradient + MCTS	- MCTS: Provides accurate short-term planning. - Policy Gradient: Offers global optimization capability—highly complementary.

4. Challenges and Limitations of Reinforcement Learning

While phenomenal success has been made by RL in CV and NLP, several challenges and bottlenecks still remain to inhibit broader and deeper applications:

Low sample efficiency, high cost to train: deep RL tends to require a large volume of interaction data. Training robots by simulation or collecting user feedback to refine language models is time- and computationally expensive. Millions of games played by self-play were required by AlphaStar, and ChatGPT's RLHF process depends considerably on human annotation and large computation. Without sufficient data or a simulator platform, methods by RL tend to fail to converge stably [43]. Improving sample efficiency continues to be a focus area, with active research directed toward improved exploration strategies, model-based RL, and offline RL to reduce reliance on interacting online [6].

Exploration–exploitation trade-off and sparse rewards: agents are likely to encounter major rewards very late in training in real-world tasks. Sparse or late rewards induce nearly zero gradients, leading to deterioration in learning. It's hard to make agents explore well without acquiring suboptimal habits. Techniques like intrinsic motivation encourage exploration with no extrinsic reward but have potential to cultivate irrelevant action. In dialogue systems, exploration is extremely risky—to output something randomly might output something offensive or undefined. Safe exploration is a new subfield trying to incorporate constraints such that no catastrophic ends are attained [38].

Reward function design and bias: RL is extremely sensitive to reward function design, and poorly designed rewards can lead to "reward hacking," where agents optimise over undesirable aspects of rewards. E.g., using token-level likelihood as a reward to use during language modeling leads to models generating repetitive, shallow content. In dialogue, poorly tuned models of rewards can lead to agents generating generic, complimentary output that scores well but lacks depth [23][33]. Even with robots, agents are known to learn to hack up reward structures (e.g., wobbling to simulate walking). Human-labels applied to rewards within RLHF also introduce noise and subjectivity, which could bias the reward model and hence learned policy. Recent work explores techniques like contrastive rewards to add uncertainty penalties to reduce overfitting to flawed reward signals.

Stability and reproducibility during training: deep RL is unstable. Random seed or environmental initialization variations even by a small margin can produce very diverse policy outcomes. The majority of state-of-the-art (SOTA) RL outcomes are difficult to reproduce with high variance and sensitivity to hyperparameters. In large models, even a small update leads to drastic changes in performance, requiring careful tuning of learning rate, noise, and other hyperparameters. The research community emphasizes benchmark reproducibility to a greater extent, with a focus toward open-source environments and code. Algorithms PPO and TRPO offer a more stable policy update and are responsible for higher reliability, but still, the training of RL remains an art where a person needs to be experienced with trial-and-error [24][39].

Scalability and generalization: although AlphaStar-like work shows success with RL in certain environments, such strategies are prone to overfitting to certain settings and are not very general. It remains challenging to apply RL algorithms to new tasks or real-world environments. In hopes to get better generalization, approaches like domain randomization, transfer learning, and meta-learning are being explored [14][44].

Game dynamics and multi-agent complexity: in multi-agent settings, game-theoretic complexities and non-stationarity are faced by RL. Training through self-play leads to equilibrium exploiting but globally suboptimal strategies. Some adversarial environments produce cycles between strategies with no dominant policy, making it challenging to train. Population-based training, first- and second-order optimization methods, and game-theory intuitions are being incorporated to pull multi-agent RL from local optima and into Nash equilibria. This remains a state-of-the-art challenge [14].

Safety and ethics: especially with NLP, output motivated by RL is ethically dubious. While models are trained to better align user preference with RLHF, they could also be trained to further augment annotator bias. Also, optimization with RLHF tends to prioritize perceived user satisfaction over truth, with a result of "model lying"—systems providing users with things they want to hear, even if they

are incorrect. This is a threat with dialog, search, and QA programs. Trial and error with RL also poses the question of how to prevent undesirable behaviors during exploration. Mitigations include collecting negative feedback, including safety-specific rewards, and output monitoring during training [11][45].

Even though RL has proved to have strong potential in CV and NLP, root issues remain to be addressed. From algorithm construction to practical use, more work remains to be done to improve RL's efficiency, stability, generalizability, and safety.

5. Future Prospects and Trends to Watch

Future progress within reinforcement learning applied to CV and NLP should be concurrent with several other new technologies.

5.1 Integrating RL with Self-Supervised Learning

Self-supervised learning has risen to prominence in recent years, pre-training powerful feature representations from massive amounts of unlabelled data. There are models such as MOCO and SimCLR in CV, and BERT and GPT in NLP. Combining self-supervised learning with RL can, on the one hand, improve the sample efficiency of RL - using visual or linguistic features extracted from pre-trained models as state representations, intelligences do not need to learn perception from scratch and can focus on policy learning faster.

For example, using pre-trained ResNet features as states in visual navigation tasks makes it easier for agents to learn semantic-based navigation strategies; embedding dialogue states with pre-trained language models in dialogue strategy learning helps to understand the context [46][47].

On the other hand, large self-supervised models themselves have some decision-making capabilities, and it is also a new idea to consider them as policy networks for RL. For example, Decision Transformer proposes the problem of equivalently transforming RL into sequence modelling, using Transformer to directly model 'state-action-reward' sequences, and solving the RL problem with the help of the training paradigm of sequence modelling. This approach achieved comparable results to traditional RL algorithms on offline RL benchmarks such as the Atari game [48].

In the future, with the emergence of more powerful multimodal self-supervised models, we can foresee that 'pre-training + RL fine-tuning' will become a common paradigm, just as pre-training-fine-tuning is today in the field of supervised learning. The fusion of large models that provide generic perception and memory, and RL that injects task-specific optimisation into their decision-making behaviours, will enable faster adaptation of intelligences to complex environments.

In addition, self-supervised learning can also provide intrinsic rewards for RL: prediction errors of pre-trained models, for example, can be used as exploration drivers to help the agent learn despite the lack of external rewards. In conclusion, the model of 'self-supervised perception + reinforcement decision-making '** is expected to greatly expand the boundaries of reinforcement learning applications, and make breakthroughs in data efficiency and generalisation.

5.2 Reinforcement Learning and Meta-Learning

Meta-learning focuses on the ability of algorithms to quickly adapt to new tasks, which coincides with the need for RL to converge quickly in new environments. In the future we will see more meta-learning approaches to reinforcement learning emerge.

For example, the MAML algorithm proposed by Finn et al. has already been used in RL to enable an intelligence to adapt to a new task in a few steps after training on many small tasks [44]. This ability to 'learn to learn' is critical for real-world applications: imagine a group of robots that first learnt how to walk in a simulated environment, and then when switched to a real different terrain, they can quickly adapt their gait without having to start from scratch. Directions for meta-RL include the development of intelligences that can infer environmental dynamics from a small amount of experience, and the training of networks with internal policy updating mechanisms [49], etc.

There is a similar need in NLP, for example, to train a conversational agent to be able to adjust its style to match a new user once it has interacted with that user briefly. In the context of RLHF, meta-

preference learning can also be considered: the ability of a model to update its policy to match new requirements with less new preference data. The combination of meta-learning and RL also promises to alleviate the challenges of multi-task learning by training a general-purpose intelligence that can efficiently perform multiple tasks in vision and language, rather than just excelling at a single task.

5.3 Multimodal Reinforcement Learning and Systems

The development of artificial intelligence is moving towards multimodal fusion, and future intelligences will need to simultaneously understand signals such as vision, speech, and sound, and make decisions accordingly. Reinforcement learning as a decision module will be tightly integrated with multimodal perception to form end-to-end multimodal decision systems. One obvious route is the combination of vision-verbal-action: for example, having a robot manipulate its environment based on a human verbal command 'bring me that red cup'. To achieve this, the intelligence needs a vision module to recognise the red cup, a language module to understand the semantics of the command, and finally RL to formulate a sequence of actions to take it. Such systems have been initially explored in the field of robotics, e.g., Google's SayCan combines a language model to generate high-level action proposals, and RL to perform low-level control, enabling language-driven robot manipulation [50]. Similarly, Singh et al. embedded CLIP into a vision-linguistic-action pipeline in CLIPort, allowing the robot to learn handling actions based on graphical cues in a desktop task [51]. It is foreseeable that there will be more architectures similar to 'LLM+RL' in the future: the large language model is responsible for planning and reasoning, and the reinforcement learning agent is responsible for interacting with the environment and executing specific actions, and the two are linked through dialogues or interfaces. The 'plan-execute' paradigm proposed by OpenAI is exactly like this, which divides complex tasks into two phases: planning is done by the language model, and execution is done by the RL agent, showing promise in scenarios such as home robots and self-driving assistants.

Another multimodal direction is to incorporate more multimodal interactions in simulations, e.g., intelligences can observe images, read text plaques, or even listen to sounds before deciding on an action. This requires RL algorithms capable of handling multimodal state spaces. Recent work has used embedded splicing of images and text as RL state inputs to allow intelligences to learn in environments with visual and textual cues (e.g., travelling in a room based on the direction of arrows on the wall). With the emergence of multimodal Transformer models (e.g., CLIP [52], ALIGN [53], which can handle both graphics and text), it is entirely possible to use a unified model as a perceptual layer to encode multimodal information, which is then connected to an RL decision layer to output actions.

5.4 Scaling Up and Real-World Deployment

As arithmetic power grows and algorithms improve, we will see larger-scale RL training unfold on real-world tasks. For example, training hundreds of intelligences simultaneously to simulate city traffic [54] and learn macro traffic signal control strategies; or deploying RL algorithms on unmanned vehicles to do path planning and obstacle avoidance on-the-fly [55]. In terms of NLP, in the future, RLHF may be extended to long-term conversations and even article writing scenarios, allowing models to learn to interact with people for long periods of time, create content, and at the same time optimise the style and structure of the text based on feedback [56]. It is foreseeable that reinforcement learning will no longer be limited to games and simulations, but will play a key role in more real-world systems, such as personalised education systems adjusting teaching strategies through RL, medical diagnosis AIs optimising the testing process through RL [57], and intelligent customer service dynamically selecting response templates using RL, and so on. In these applications, security and reliability are crucial, so we will see the pattern of hybrid intelligence: the coexistence of RL intelligences with rule-based systems, where RL is responsible for the flexible decision-making part and the rule-based system provides the security boundary. In this way, RL is gradually introduced into mission-critical systems to ensure safety while enjoying the efficiency gains it brings.

Looking ahead, RL will become more integrated with multimodality, leverage large models, and focus on human-computer synergy so that it can be useful on a wider range of complex tasks. It is foreseeable that reinforcement learning will become an indispensable part of the general AI system,

collaborating with perception and inference modules to achieve truly intelligent behavioural decisions. We believe that as researchers continue to overcome the above challenges, the application boundaries of RL will continue to expand and bear more fruitful fruits in the fields of computer vision and natural language processing.

6. Conclusion

Reinforcement Learning (RL) has made impressive progress in computer vision and natural language processing in the last five years. From active visual perception to dialogue strategy optimisation to alignment tuning of large models, RL has demonstrated the unique advantage of combining perception and decision making. This paper reviews the major applications in this field from 2020-2025, classifies typical tasks and methods of RL in CV and NLP, summarises the mechanisms and advantages of mainstream algorithms in each scenario, and discusses the current challenges and future directions. Overall, reinforcement learning is gradually moving from research to practice: with more robust and efficient algorithms, as well as integration with self-supervision, big models and other techniques, RL will play a central role in more real-world AI systems. Of course, we also need to be aware of the risks and limitations associated with RL applications, and the only way to achieve safe and reliable reinforcement learning intelligences is to take a multi-pronged approach to algorithms, data and feedback.

References

[1] Caicedo J. & Lazebnik S. (2015). Active Object Localization with Deep Reinforcement Learning. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pp. 2488–2496.

[2] Pirinen A. & Sminchisescu C. (2018). Deep Reinforcement Learning of Region Proposal Networks for Object Detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 6945–6954. DOI:10.1109/CVPR.2018.00726.

[3] Song K., Zhang W., Song R., Li Y. (2020). Online Decision Based Visual Tracking via Reinforcement Learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 33, pp. 11778–11788.

[4] Zhu Y., Mottaghi R., Kolve E., Lim J. J., Gupta A., Fei-Fei L., Farhadi A. (2017). Target-driven Visual Navigation in Indoor Scenes using Deep Reinforcement Learning. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pp. 3357–3364. DOI:10.1109/ICRA.2017.7989382.

[5] Ma C., Xu Q., Wang X., Jin B., Zhang X., Wang Y., Zhang Y. (2021). Boundary-Aware Supervoxel-Level Iteratively Refined Interactive 3D Image Segmentation With Multi-Agent Reinforcement Learning. *IEEE Transactions on Medical Imaging*, 40(10):2563–2574. DOI:10.1109/TMI.2020.3048477.

[6] Li Y. (2017). Deep Reinforcement Learning: An Overview. arXiv:1701.07274.

[7] Arulkumaran K., Deisenroth M. P., Brundage M., Bharath A. A. (2017). A Brief Survey of Deep Reinforcement Learning. *IEEE Signal Processing Magazine*, 34(6):26–38. DOI:10.1109/MSP. 2017. 2743240.

[8] Mnih V., Kavukcuoglu K., Silver D., Rusu A. A., Veness J., Bellemare M. G., Graves A., Riedmiller M., Fidjeland A. K., Ostrovski G., Petersen S., Beattie C., Sadik A., Antonoglou I., King H., Kumaran D., Wierstra D., Legg S., Hassabis D. (2015). Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533. DOI:10.1038/nature14236.

[9] Sutton R. S. & Barto A. G. (2018). Reinforcement Learning: An Introduction (2nd Edition). MIT Press.

[10] Mnih V., Puigdomènech Badia A., Mirza M., Graves A., Lillicrap T., Harley T., Silver D.,

Kavukcuoglu K. (2016). Asynchronous Methods for Deep Reinforcement Learning. In *Proceedings of the 33rd International Conference on Machine Learning (ICML)*, vol. 48, pp. 1928–1937.

[11] Lillicrap T. P., Hunt J. J., Pritzel A., Heess N., Erez T., Tassa Y., Silver D., Wierstra D. (2016). Continuous Control with Deep Reinforcement Learning. In *4th International Conference on Learning Representations (ICLR)*. arXiv:1509.02971.

[12] Schulman J., Wolski F., Dhariwal P., Radford A., Klimov O. (2017). Proximal Policy Optimization Algorithms. arXiv:1707.06347.

[13] Ouyang L. et al. (2022). Training Language Models to Follow Instructions with Human Feedback. In *Advances in Neural Information Processing Systems (NeurIPS)*. URL: [https: //openreview.net/forum?id=TG8KACxEON](https://openreview.net/forum?id=TG8KACxEON).

[14] Bellemare M. G., Veness J., Bowling M. (2013). The Arcade Learning Environment: An Evaluation Platform for General Agents. Journal of Artificial Intelligence Research, 47:253–279. DOI:10.1613/jair.3912.

[15] Bellemare M. G., Veness J., Bowling M. (2013). The Arcade Learning Environment: An Evaluation Platform for General Agents. Journal of Artificial Intelligence Research, 47:253–279. DOI:10.1613/jair.3912.

[16] Mirowski P., Pascanu R., Viola F., Soyer H., Ballard A., Denil M., Kavukcuoglu K., Gómez Colmenarejo S., Kumaran D., Hadsell R. (2017). Learning to Navigate in Complex Environments. In *Proceedings of the International Conference on Learning Representations (ICLR)*.

[17] Vinyals O. et al. (2019). Grandmaster level in StarCraft II using multi-agent reinforcement learning. *Nature*, 575(7782):350–354. DOI:10.1038/s41586-019-1724-z.

[18] Kwiatkowski A., Alvarado E., Kalogeiton V., Cani M.-P. (2022). A Survey on Reinforcement Learning Methods in Character Animation. *Computer Graphics Forum*, 41(2):507–540. DOI:10. 1111/cgf. 14486.

[19] Han D., Mulyana B., Cheng S. (2023). A Survey on Deep Reinforcement Learning Algorithms for Robotic Manipulation. *Sensors*, 23(7):3762. DOI:10.3390/s23073762.

[20] Cao Y., Zhao H., Cheng Y., Shu T., Chen Y., Liu G., Liang G., Zhao J., Yan J., Li Y. (2023). Survey on Large Language Model-Enhanced Reinforcement Learning: Concept, Taxonomy, and Methods. *IEEE Transactions on Neural Networks and Learning Systems*. DOI:10.1109/TNNLS. 2023. 3283105

[21] Singh S., Litman D., Kearns M., Walker M. (2002). Optimizing Dialogue Management with Reinforcement Learning: Experiments with the NJFun System. *Journal of Artificial Intelligence Research*, 16:105–133. DOI:10.1613/jair.953

[22] Ranzato M., Chopra S., Auli M., Zaremba W. (2016). Sequence Level Training with Recurrent Neural Networks. In *Proceedings of the International Conference on Learning Representations (ICLR)*.

[23] Paulus R., Xiong C., Socher R. (2018). A Deep Reinforced Model for Abstractive Summarization. In *Proceedings of the International Conference on Learning Representations (ICLR)*.

[24] Yu L., Zhang W., Wang J., Yu Y. (2017). SeqGAN: Sequence Generative Adversarial Nets with Policy Gradient. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 31(1):2852–2858. DOI:10.1609/aaai.v31i1.10804.

[25] Shi Z., Chen X., Qiu X., Huang X. (2018). Toward Diverse Text Generation with Inverse Reinforcement Learning. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, pp. 4156–4162.

[26] Christiano P. F., Leike J., Brown T. B., Martic M., Legg S., Amodei D. (2017). Deep Reinforcement Learning from Human Preferences. In *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 30, pp. 4299–4307.

[27] Ziegler D. M., Stiennon N., Wu J., Brown T. B., Radford A., Amodei D., Christiano P., Irving G. (2019). Fine-Tuning Language Models from Human Preferences. arXiv:1909.08593.

[28] Stiennon N., Ouyang L., Wu J., Ziegler D. M., Lowe R., et al. (2020). Learning to Summarize from Human Feedback. In *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 33, pp. 3008–3021.

[29] Peng B., Li X., Gao J., Liu J., Wong K.-F. (2018). Deep Dyna-Q: Integrating Planning for Task-Completion Dialogue Policy Learning. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (ACL)*, pp. 2182–2192. DOI:10.18653/v1/P18-1203.

[30] Hou Z., Liu B., Zhao R., Ou Z., Liu Y., Chen X., Zheng Y. (2021). Imperfect also Deserves Reward: Multi-Level and Sequential Reward Modeling for Better Dialog Management. In *Proceedings of the North American Chapter of the ACL (NAACL)*, pp. 1234–1246.

[31] Li J., Monroe W., Ritter A., Galley M., Gao J., Jurafsky D. (2016). Deep Reinforcement Learning for Dialogue Generation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 1192–1202. DOI:10.18653/v1/D16-1127.

[32] Shen S., Cheng Y., He Z., He W., Wu H., Wang Y., Liu M., Zhang Y., Sun M. (2016). Minimum Risk Training for Neural Machine Translation. In *Proceedings of the Annual Meeting of the ACL (ACL)*, pp. 1683–1692. DOI:10.18653/v1/P16-1166.

[33] He D., Xia Y., Qin T., Wang L., Yu N., Liu T.-Y., Li W. (2016). Dual Learning for Machine Translation. In *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 29, pp. 820–828.

[34] Grissom II A., He H., Boyd-Graber J., Morgan J., Daumé III H. (2014). Don't Until the Final Verb Wait: Reinforcement Learning for Simultaneous Machine Translation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 1342–1352.

[35] Nguyen K., Daumé III H., Boyd-Graber J. (2017). Reinforcement Learning for Bandit Neural Machine Translation with Simulated Human Feedback. In *Proceedings of EMNLP*, pp. 177–188.

[36] Finn C., Abbeel P., Levine S. (2017). Model-Agnostic Meta-Learning for Fast Adaptation of Deep Networks. In *Proceedings of the International Conference on Machine Learning (ICML)*, pp. 1126–1135. DOI:10.5555/3305381.3305493.

[37] Duan Y., Schulman J., Chen X., Bartlett P. L., Sutskever I., Abbeel P. (2016). RL^2: Fast Reinforcement Learning via Slow Reinforcement Learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 29, pp. 8024–8034.

[38] Brohan A., Hejna J., Duena E., Levine S. (2022). SayCan: Grounding Language in Robotic Affordances. In *Robotics: Science and Systems (RSS)*.

[39] Shridhar M., Manuelli L., Fox D. (2021). CLIPort: What and Where Pathways for Robotic Manipulation. In *Conference on Robot Learning (CoRL)*, pp. 143–156. arXiv:2109.12098.

[40] Radford A., Kim J. W., Hallacy C., Ramesh A., Goh G., Agarwal S., Sastry G., Askell A., Mishkin P., et al. (2021). Learning Transferable Visual Models From Natural Language Supervision. In *Proceedings of the International Conference on Machine Learning (ICML)*, pp. 8748–8763. arXiv:2103.00020.

[41] Jia C., Yang Y., Zhang Y., et al. (2021). Scaling Up Visual and Vision-Language Representation Learning With Noisy Text Supervision. In *Proceedings of ICML*, pp. 4904–4916. arXiv:2102.05918

[42] Li Z., Xu C., Zhang G. (2021). A Deep Reinforcement Learning Approach for Traffic Signal Control Optimization. arXiv:2107.06115.

[43] Sallab A. E., Abdou M., Perot E., Yogamani S. (2017). Deep Reinforcement Learning framework for Autonomous Driving. *Electronic Imaging*, 29(19):70–76. DOI:10.2352/ISSN.2470-1173.2017.19.AVM-023.

[44] Riedmann A., Schaper P., Lugrin B. (2025). Reinforcement Learning in Education: A Systematic Literature Review. *International Journal of Artificial Intelligence in Education*. DOI:10.1007/s40593-025-00494-6.

[45] Yu Z., Li Y., Kim J., Huang K., Luo Y., Wang M. (2023). Deep Reinforcement Learning for Cost-Effective Medical Diagnosis. arXiv:2302.10261.

[46] Rashid T., Samvelyan M., Schröder de Witt C., Farquhar G., Foerster J., Whiteson S. (2018). QMIX: Monotonic Value Function Factorisation for Deep Multi-Agent Reinforcement Learning. arXiv:1803.11485.

[47] Zhang C., Quigley M., Burch C., Whiteson S. (2021). QPLEX: Duplex Dueling Network Architectures for Deep Multi-Agent Reinforcement Learning. In *Proceedings of ICML*, pp. 7318–7328. arXiv:2010.14715.

[48] Yu C., Velu A., Vinitsky E., Gao J., Wang Y., Bayen A., Wu Y. (2022). The Surprising Effectiveness of PPO in Cooperative Multi-Agent Games. arXiv:2103.01955.

[49] Lipton, Z. C., Berkowitz, J., & Elkan, C. (2015). A critical review of recurrent neural networks for sequence learning. arXiv preprint arXiv:1506.00019.

[50] Paulus R., Xiong C., Socher R. (2017). A Deep Reinforced Model for Abstractive Summarization. arXiv:1705.04304.

[51] Nikpour, B., Sinodinos, D., & Armanfard, N. (2024). Deep reinforcement learning in human activity recognition: A survey and outlook. IEEE Transactions on neural networks and learning systems, 36(3), 4267-4278.

[52] Dodge, J., Ilharco, G., Schwartz, R., Farhadi, A., Hajishirzi, H., & Smith, N. (2020). Fine-tuning pretrained language models: Weight initializations, data orders, and early stopping. arXiv preprint arXiv:2002.06305.

[53] Bai Y., Jones A., Ndousse K., Askell A., Winograd A., et al. (2022). Constitutional AI: Harmlessness from AI Feedback. arXiv:2212.08073.

[54] Rafailov R., Sharma A., Mitchell E., Ermon S., Manning C. D., Finn C. (2024). Direct Preference Optimization: Your Language Model is Secretly a Reward Model. arXiv:2305.18290.

[55] Arora, D., & Zanette, A. (2025). Training language models to reason efficiently. arXiv preprint arXiv:2502.04463.

[56] Finn, C., Xu, K., & Levine, S. (2018). Probabilistic model-agnostic meta-learning. Advances in neural information processing systems, 31.

[57] Madan, K., Ke, N. R., Goyal, A., Schölkopf, B., & Bengio, Y. (2021). Fast and slow learning of recurrent independent mechanisms. arXiv preprint arXiv:2105.08710.