Publisher: State and Provincial Joint Engineering Lab. of Advanced Network

Monitoring and Control (ANMC)

Cooperate:

Xi'an Technological University (CHINA)
West Virginia University (USA)
Huddersfield University of UK (UK)
Missouri Western State University (USA)
James Cook University of Australia
National University of Singapore (Singapore)

Approval:

Library of Congress of the United States

Shaanxi provincial Bureau of press, Publication, Radio and Television

Address:

4525 Downs Drive, St. Joseph, MO64507, USA

No. 2 XueFu Road, WeiYang District, Xi’an, 710021, China

Telephone: +1-816-2715618 (USA) +86-29-86173290 (CHINA)

Website: www.ijanmc.org

E-mail: ijanmc@ijanmc.org

xxwlcn@163.com

ISSN: 2470-8038

Print No. (China): 61-94101

Publication Date: December 28, 2023

Administrator

Associate Editors

Language Editor

Would you like to be an Associate Editor? Simply send a request together with your Curriculum Vitae to xxwlcn@163.com. We will have a team of existing editors or at least three experts in your field to review your request and make a decision as soon as we can. The criteria to be an associate editor are: 1. must have advanced degree; 2. must be a leader or have outstanding achievements in the specific research field; 3. must be recommended by the review team.

Table of Contents

A Model-Based Approach to Mobile Application Testing

Weidong Xu
School of Computer Science and Engineering
Xi'an Technological University
Xi'an, 710021, China
E-mail: xuweidong@st.xatu.edu.cn

Jing Cheng
School of Computer Science and Engineering
Xi'an Technological University
Xi'an, 710021, China
E-mail: chengjing@xatu.edu.cn

***Abstract*—Modeling the automated testing of mobile applications is a crucial aspect of mobile application automation testing. Due to the varied styles and complex interactions of mobile applications, automated modeling methods are urgently required, particularly in the context of their short development cycles, large numbers, and fast version iterations and updates. In this paper, we address the challenge of exploring mobile application behavior and state based on robotic testing environment without invading the application interior, and propose a method for automated exploration of GUI components and GUI events of applications combined with application domain knowledge to generate mobile application GUI semantic test models. Our results show that the proposed semantic model achieves 70.6% and 82.4% defect detection rate in the robot vision environment and simulation environment, respectively. Compared with the comparative testing method that can only find application crash defects, our method can explore both crash defects and functional anomalies with the application semantic understanding and domain knowledge, thereby extending the automated mobile application functional testing capability of mobile applications. In response to the limitations of mobile application automated testing modeling mentioned above, this paper introduces an automated testing method based on semantic models. It uses the proposed semantic testing model to guide the purposeful exploration of the tested application's states. Subsequently, it generates positive and negative test cases based on the domain knowledge associated with the semantic model. This modeling approach leverages domain models in the mobile application field to conduct automated modeling tests imbued with functional significance, guided by domain knowledge. This optimization aims to address the shortcomings of current automated testing, particularly in terms of model reuse and test expansion.**

Keywords-Mobile Application; Semantic Testing Model; Automated Testing; Test Coverage

I. INTRODUCTION

With the gradual replacement of traditional desktop software by mobile applications as the mainstream software tools in people's daily work, study and life, the development of mobile applications introduces complex new features that make quality assurance more challenging compared to desktop software [1]-[8]. However, the growth in testing needs is in contrast to the limited availability of testing tools and testers. Currently, both industry and academia are increasingly emphasizing the adoption and exploration of automated testing techniques to address the testing needs of mobile applications [9]-[12].

Model-based automated testing is a widely studied testing method that involves constructing test models by mining the state and behavior of the application, and subsequently utilizing these models to generate test cases [13]. Model-based automated testing typically involves completing static modeling of mobile applications based on GUI and dynamic modeling based on behavior jumping, and describing mobile application behavior using inter-state relationship models such as Finite State Machine (FSM). For instance, GUI Ripper [14] facilitates automated exploration modeling of applications, while tools like UI Automator [15]-[16] are utilized to obtain the GUI tree of an application and target the selection of events to complete exploration modeling of mobile applications.

Research on improvements related to model based automated testing: on the one hand, to achieve automatic evolution of the model, e.g., Gu et al [17] can find differences and quickly achieve

DOI: 10.2478/ijanmc-2023-0071

model evolution after application version update. DeltaDroid [18] builds a defect model that can generate new cases under different conditions based on existing test cases combined with actual GUI and system actions to detect dynamic installation defects in Android applications. On the other hand, the modeling is guided by enhanced application knowledge, e.g., MEGDroid [19] uses model abstraction and model-to-model migration methods to achieve accurate generation of application events; Pan et al [20]-[23] use manual construction of richer test models to guide automated testing.

To overcome the limitations of mobile application automation test modeling mentioned above, this chapter proposes a semantic model-based automation test approach. This approach fuses the mobile application domain model to achieve a semantic matching association between mobile application GUI state and domain knowledge. This facilitates the automated generation of a mobile application GUI semantic test model, which is subsequently used to verify the testing effectiveness of the modeling method.

II. MOBILE APPLICATION SEMANTIC TESTING MODEL

When it comes to mobile application testing, a typical model-based approach involves modeling the relationship between mobile application GUI states and GUI events to establish a series of jumps triggered by events between different GUI states in the application. One example of such an approach is the finite state machine (FSM) model [24]. However, current model-based approaches for mobile application testing are limited to direct records of GUI states and GUI events. These models can only describe the jump-trigger relationship between different GUI states of the application, without understanding the functional meaning of the application. To overcome the aforementioned problems, this paper proposes a method that integrates semantic ontology models with traditional GUI testing models. This method involves building a semantic testing model for mobile applications by extracting semantic information from the GUI of mobile applications and attaching semantic concepts to GUI states and GUI events.

Definition 1: A typical semantic definition of OWL, describing ontology with a formal definition of six tuples:

$$onto\log y=\left\{C,A_C,R,A^R,H,X\right\} \tag{1}$$

A^c denotes the set of attributes of each concept, the set of concept attributes $A^c(c_i)$, each concept c_i in the set of concepts C is used to represent a set of objects of the same kind and can be described by the same set of attributes.

R denotes the set of relations between concepts, relation r_i (c_p,c_q) that is, each relation r_i in relation R represents a binary relation between concepts cp and cq, and an instance of this relation is a pair of concept objects (c_p,c_q).

A^R denotes the set of attributes of each relation, and the set of relation attributes $A^R(r_i)$ is used to represent the attributes of relation $r_{i.}$

H represents a concept hierarchy, where it is a hierarchy of sets of concepts denoted as C. H also includes a set of parent-child relations that exist between the concepts in C.

The set of axioms is represented as X, where each axiom within X serves as a constraint on the attribute values of a concept and its relation, or as a constraint on the relations between conceptual objects.

Definition 2: The mobile application domain metamodel is defined as a 4-tuple $OAPP=\left\{C,I,R,X\right\}$.

The set of mobile application ontology concepts is denoted by C and is composed of three subsets: entity, action, and task. Each subset contains a concept identifier, concept type, and semantic name.

I represents a collection of instances of mobile application ontology concepts. These instances are concrete textual representations of mobile application component values that are recognized knowledge in the domain. For instance, examples of instances for the entity "place name" could include "Xi'an", "Beijing", and "Shanghai". These instances can be used as the origin or destination in an air service application.

R denotes the set of semantic relations of the mobile application, which contains the inter-concept relation R_{cc} and the concept-instance relation R_{ci} .

X is the constraint definition of the mobile application domain model, which encompasses inter-concept relationship constraints, constraints on concept attributes, instance data constraints, and more. For instance, one constraint could be that a task must consist of at least one entity and one action, and that the instance of the entity "place name" can only be described by text.

Definition 3: The domain model action flow graph is defined as a 5-tuple, $D=\left\{A,T,F,\alpha_S,\alpha_f\right\}$.

Action state set A: finite set of mobile application interaction actions.

Action flow control set T: control set of mobile application action sequences.

set of action flow relations F: set of relations from one action state to another action state.

Initial states: a_s， $a_f \in A$.

End state: a_f， $a_f \in A$.

$A=\left\{a_1,a_2,a_3,\ldots,a_n\right\}$ is the set of action states, which is the set of mobile application test interaction actions. Each action state is connected by the control flow T in series to form an action sequence. The action state set includes the initial state a_s , the end state a_f , the intermediate states $\left\{a_m \middle| m=1,2,3,\ldots,n\right\}$.

$T=\left\{t_1,t_2,t_3,\ldots,t_n\right\}$ is the action flow control set, which is the control set of the mobile application test action sequence.

$F=\left\{f_1,f_2,f_3,\ldots,f_n\right\}$, $F\subseteq(A\times T)\cup(T\times A)$ is the set of action relations describing the combination of action states and action flow controls.

$a_S=\left\{a\in A\middle|(a,t)\in F\right\}$ For the initial state only backward control flow t, $a_f=\left\{a\in A\middle|(t,a)\in F\right\}$ for the end state only forward control flow t, $a_m=\left\{a\in A\middle|(a,t)\in F\wedge(t,a)\in F\right\}$ must have both forward and backward control flows.

Definition 4: A component in a GUI that cannot be split is an atomic component. The basic components of a GUI in general are atomic components, such as text buttons (Button), text (Text View), images (Image View), etc. AC denotes the set of atomic components in a GUI, $\forall ac\in AC$, $ac=\left\{ac^t,ac^v,ac^a,ac^s,ac^p\right\}$, where:

ac^t denotes the component type of the atomic component ac.

ac^v indicates the value of the atomic component ac, which is an optional attribute.

ac^a indicates the possible actions of the atomic component ac, e.g. a button is usually bound to a click action.

ac^s denotes the semantics of the atomic component ac. The semantic entity of ac^v is obtained by mapping ac^v to the domain model ac^s: $ac^v\rightarrow\left\{C_e\middle|C_e\in C_E\right\}$.

Definition 5: A component composed of atomic components in a GUI is a semantic composite component, e.g., ListView, ToolBar, etc. CC denotes a collection of semantic composite components in a GUI, $cc=\left\{cc^{ac},cc^t,cc^a,cc^s,cc^p\right\}$, where:

cc^{ac} denotes the composition of the semantic composite component cc, i.e., which atomic components the semantic composite component cc is composed of, cc^{ac} is the set of atomic components, $cc^{ac}\in AC$.

The component type of a semantic composite component is denoted by cct, and this attribute is optional. It's possible that there may not be a corresponding GUI component type for the semantic composite component.

The semantics of the semantic composite component cc is represented by λc^{cs}, which is determined by the relationship in the domain model where the semantics of the constituent atomic components reside. This can be denoted $\bigwedge ac^s\rightarrow cc^s$. It should be noted that despite being a composite component, the semantics of cc still corresponds to the entities present in the domain model.

Definition 6: The extended semantic FSM model, FSM-ES, is an extension of the typical FSM model that uses the 5-tuple setting. However, it introduces a semantic extension to the expression of the state-hopping relationship: $FSM-ES=\{S,\Sigma,\delta,S_0,F\}$.

The infinite non-empty state set of the GUI is denoted as S, which encompasses all possible states of the application being tested. For ∀s∈S, $s=\{AC,CC,S^S\}$, where AC represents the GUI atomic component, CC represents the GUI semantic composite component, and S^S denotes the semantics of the GUI state that is being represented by the GUI component.

δ is the state transfer function that maps S × Σ to the transition function of S δ:S × ∑→S. ∀s ∈ S, ∀e ∈ ∑.

The notation δ (s, e) refers to the set of states that can be accessed by transitioning from the GUI state s through event e.

III. A MODEL-BASED APPROACH TO MOBILE APPLICATION TESTING

We propose a semantic model-based mobile application testing method, which consists of two parts: visual semantic model-driven GUI modeling and task subgraph-based test case generation.

In the realm of semantic model-driven automated test modeling, the following critical modules are present:

A. FSM-ES model building

The process of semantic model-driven GUI modeling is illustrated in Algorithm 1, which takes the mobile application domain model (ADM), the generic model (GDM), and the application under test (AUT) as inputs and produces the FSM-ES model of the application under test as outputs. The following pseudo code outlines this process:

Algorithm 1

Input. Application under test AUT, application domain model ADM, generic model GDM

Output. Application test model FSM-ES

```
while true do
    Get the current GUI gui_current of the application under test
    Perform visual recognition of GUI component elements on gui_current to get GUI component information gui_info
    Match the gui_info of the current GUI with the ADM for semantic similarity to get the vector gui_vc
    gui_action inferAction(gui_vector)
    Get the response interface vector gui_vr
    if gui_vc differs from gui_vr then
        mark gui_a in gui_action as executed
        Generate a path to "gui_vc, gui_a, gui_vr" f
        if path f does not exist in fsm_es then
            Path f is added to fsm_es
        end if
    else
        Logging exceptions
    end if
    if there is no unexecuted action in gui_action or the exploration timeout then
        break
    end if
end while
return

function inferAction(gui_vector)
    for each gui state in ADM do
        if gui state == gui_vector then
            Reasoning generates gui actions in the domain corresponding to the gui state and saves them to gui_action
        else
            Generate executable actions for the interface based on GDM probabilities and save them to gui_action
        end if
    end for
    return gui_action
end function
```

The primary objective of the FSM-ES model is to explore the attainable states of the application and associate each state with the task ontology outlined by the domain model. The structure of the FSM-ES model is depicted in TABLE I. FSM-ES semantic structure of music playback application Cathay Pacific

Area	Mssion	State Collection	Action Semantic Set
Music playback	Register	$\{S_0,S_1,S_2,S_3,S_4,S_{12}\}$	{select login, agree to the agreement, enter the username, enter the password, and click 1ogin}
	Basin information	$\{S_5,S_6,S_7,S_8,S_9\}$	{Recently played, locally downloaded personal cloud drive, friend 1ist, favorite p1ay1ist}
	Discovering Music	$\{S_{10},S_{11},S_{12}\}$	{Dai1y recommendation, click 1ike, c1ick play}
	Search Services	$\{S_{13},S_{14},S_{15},S_{16}\}$	{Click to type, c1ick to search, clear search,1isten to music and recognize music}
	Persoral Settings	$\{S_{17},S_{18},S_{19},S_{20},S_{21},S_{22},S_{23}\}$	{Message center, personal privacy, personalized services, advanced settings, about, 1og out, switch accounts}

B. Task Subgraph Generation

The FSM-ES model is utilized to map to the action flow diagram of the application domain model, which generates task subgraphs for the functional tasks of the application to produce test cases.

If every action concept in the task concept, as defined in the FSM-ES model, corresponds to an action state found in an AFG in the domain, then the task concept is deemed to comply with a specific action flow graph (AFG). The following pseudo code outlines this process:

Algorithm 2

```
Input.    Application Semantic Model FSM-ES,
Domain Model Action Flow Graph AFG
Output.    Task subgraph TFG
for each task in the semantic model    do
    for each task task in each path f do
        task_path     generatePath(task, AFG)
        if task_path has a serial relationship with TFG then
            Generate TFG by storing task_path in sequence
        else
            return exception and interrupt location
            break
        end if
    end for
end for
return TFG
function generatePath(task, AFG)
    if task contains the action state and AFG meets rule 2 then
        Generate a feasible action flow based on AFG inference action
        task_path sets the continuity flag
    else
        task_path sets the end flag
    end if
    return task_path
end function
```

As observed, the domain knowledge incorporated in the action flow diagram can be leveraged to supplement the unexplored inter-state action behavior, thereby generating test judgment criteria for mobile applications.

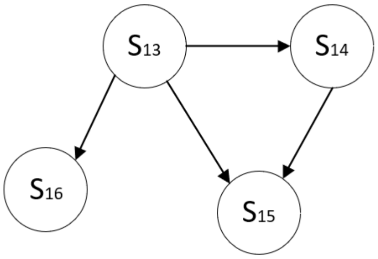


Figure 1. Task states explored by FSM-ES

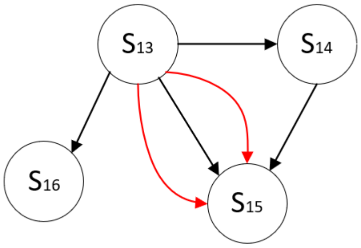


Figure 2. Task Subgraph of Domain Action Flowchart

Upon examining the actual GUI of NetEase cloud music's search song task in (f), it becomes apparent that while the FSM-ES has explored the GUI states, it has failed to account for the iterative behaviors of keying and deleting in S_{13} state and S_{15} state. This is because the FSM-ES prioritizes state exploration over other factors. However, by extending the corresponding action flow diagram definition in the domain ontology library based on domain knowledge, a pathway can be generated, and this domain knowledge can be effectively applied to GUI modeling.

The process of generating task subgraphs from FSM-ES is essentially the instantiation of the mobile application domain model on the application under test. This transforms the abstract concept relationships of the mobile application domain model into a concrete sequence of application action behaviors, making the mobile application testable for execution.

C. Test case generation based on task subgraph

Test case generation comprises two main components: test sequence generation and test data generation. Test sequences are generated by defining semantics-oriented test coverage criteria to guide the traversal of task subgraphs.

Coverage decision rule 1: Existence decision c(A, B), i.e., if the element in A exists in B, the coverage decision is satisfied.

Coverage decision rule 2: Sequential decision s(A, B), i.e., if the elements in A are sequential and conform to the sequential arrangement in B, the coverage decision is satisfied.

Coverage rule 3: Key point decision d (A, B), i.e., if there is an element in A that matches the key element in B, then the coverage decision is satisfied.

Definition a: Semantic concept entity coverage criteria

The set of test cases ensures coverage of both the conceptual entities present in the application domain model being tested and all GUI states formed by these entities.

$$EntityCoverage = c\left(F_A, O_A\right) \cdot s\left(T_G, F_G\right) \qquad (2)$$

Where:

c(F_A , O_A) — the coverage of the conceptual entity FA involved in the FSM-ES model of the application under test with the entity OA included in the domain model.

s(T_G , F_G) — the coverage of all GUI states involved in the test case set with the GUI states contained in the FSM-ES model of the application under test.

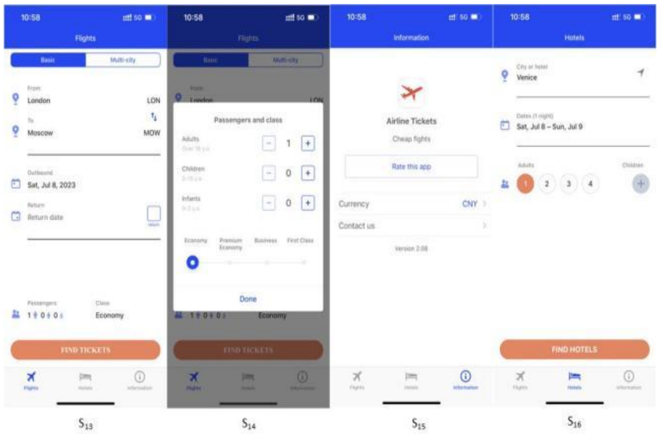


Figure 3. Login Action Test Case

Definition b: Semantic concept action coverage criteria

Test cases satisfy the coverage of actions involved in the action flow diagram of the application domain model being tested. Action coverage is evaluated independently for each action flow diagram. The coverage of semantic concept actions is calculated using the equation (3).

$$ActionCoverage = \frac{\sum_{m=1}^{M} c\left(S_m \cdot D_m\right)}{M} \qquad (3)$$

M—M is the number of task subgraphs included in the FSM-ES model of the application under test, and the action sequence coverage within a subtask is calculated for each task subgraph.

$c\left(S_m \cdot D_m\right)$ —The set of action sequences involved in the set of test cases of the subtask Sm and the coverage of feasible action sequences contained in the activity diagram of the subtaskv D_m.

Definition c: Semantic concept task coverage criteria

Test cases fulfill the coverage requirements of the subgraph of the task being tested, effectively covering all relationships between GUI states. The evaluation of coverage for semantic concept tasks is calculated using the equation presented in (4).

$$TaskCoverage = \frac{d\left(X, X_D\right)}{M} \qquad (4)$$

X — the number of application subtasks covered by the test case set

X_D — the number of tasks defined in the domain model

The coverage of semantic concept tasks, which focuses on the application's functional completeness, is calculated by assessing the coverage of tasks in the domain model using the test case set. The number of subtasks covered by the test case set is represented by X, and subtask coverage is assessed using the d-judgment rule. Specifically, if a test case can cover any pathway from the initial state to the end state of a task subgraph, it is deemed as covered and assigned a value of 1. Conversely, if there is no pathway from the initial state to the end state of the task subgraph, it is regarded as not covered and assigned a value of 0.

$$Cov = \alpha \cdot EntityCov + \beta \cdot Action + \gamma \cdot TaskCov \qquad (5)$$

Where, α、β、γ correspond to the adjustable parameter weights of the three coverage criteria.

a. To generate test cases based on task subgraphs, it is necessary to cover as many feasible paths as possible to achieve high test case coverage and optimize testing effectiveness.

Algorithm 3

Input. Task subgraph TFG of the application under test

Output. Test case sequence TCi, i=1,2,3,4,......

```
while there is an untraversed path in tfg do
    node← getStartNode(tfg)
    Create a new sequence TCi with node number i at the beginning
    while node is not the end node with degree 0 do
        if node has not been traversed then
            p← generatePath(node, tfg) //generate test behavior path
            Add the node and events from p to TCi
            Set the corresponding path in tfg to traversed state
        else
            p ← generatePath(node, tfg) //generate test behavior paths
        end if
        node ← p.nodef
    end while
    i++
end while
return TC

function generatePath(node, tfg)
    if node has a unique neighboring node and an untraversed path then
        Generate this unique path with the node p
    else if n ode does not have the same degree of adjacency then
        Generate a path with the node that has the highest degree p
        else if node has a path that has not been traversed then
        Generate p from this untraversed path
    else if node has multiple untraversed paths then
        if node's neighboring nodes have traversed nodes then
            Generate a path p between the node and its traversed neighbors
        else
            Generate a path from the node to a random neighbor node p
        end if
    else
        Generate a path p between the node and any of the next nodes
    end if
    return p← <beginning node nodes, event e, ending node nodef>
end function
```

For a given task subgraph TSG:

(i) If there exists a unique pathway of the current node with only one neighboring node, the pathway between the current node and that neighboring node is generated as the next test behavior path.

(ii) If the current node has multiple neighboring nodes with different out degrees, the pathway of the current node and the neighboring node with the highest out degree will be generated as the next test behavior path.

(iii) If there are multiple neighboring nodes at the current node and there is an untraversed path between the current node and one of the neighboring nodes, the untraversed path will be generated as the next test behavior path.

(iv) If the current node has multiple neighboring nodes and there are multiple untraversed paths between it and the neighboring nodes, the path of the current node with the traversed neighboring nodes will be generated as the next test behavior path. If there are no traversed neighboring nodes, the path of a random neighboring node will be generated as the next test behavior path.

IV. Experimentation and Analysis

When selecting applications for testing, those with similar functions are grouped together as domain applications, such as airline service applications, file management applications, news applications, and so on. To establish the domain models, two teams, each consisting of five lab personnel, were invited to create two domain models based on the definition of domain models. These models were cross-checked for accuracy and consistency.

A. Evaluation criteria

Test effectiveness is evaluated at three levels (1) the success rate of test action execution, which determines whether the actions in the test script can be executed without error; (2) the success rate of test script execution, which determines whether the test script can be executed in its entirety without encountering any errors; and (3) the success rate of defect discovery, which evaluates whether known defects in the application set can be identified.

B. Experimental Setup

Upon completion of the exploration of the application under test and subsequent building of the semantic test model, the coverage of the semantic test model with respect to the domain model is determined by analyzing the successful matching of the application state and the domain model as recorded by the exploration algorithm. The coverage results of the FSM-ES models created for each domain tested application, along with the corresponding domain model, are presented in TABLE II. The table indicates that the average coverage of entity concept is 89%, while the average coverage of action concept is also 89%. The average coverage of task concept is 81%. It is true that the non-intrusive environment may have some impact on the modeling process, particularly with regard to factors such as GUI recognition accuracy.

Application crash defects:

ActivityNotFoundException, Activity not found exception.

IllegalArgumentException, illegal parameter exception.

IllegalStateException, illegal state exception.

NullPointer Exception, null pointer exception.

TABLE I. DISTRIBUTION OF DEFECTS DISCOVERED BY VARIOUS METHODS

Error type	Defect type found	Semantic Modeling in Robot Vision Environment	Semantic modeling in a simulated environment	Humanoid	Stoat
Application crash defects	ActivityNotFoundException	1	1	3	2
	lllegalArgumentException	3	2	1	1
	lllegalState Exception	3	2	2	2
	NullPointer Exception	4	3	2	0
	OutOfMemoryError	2	2	1	2
	amount	13	10	9	7

TABLE II. DEFECT DISCOVERY RESULTS FOR EACH APPLICATION

APP	Semantic Modeling in Robot Vision Environment			Semantic modeling in a simulated environment		
	Entity Coverage	Action Coverage	Task Coverage	Entity Coverage	Action Coverage	Task Coverage
Apple Music	84%	82%	87%	84%	82%	89%
QQ Music	83%	89%	85%	86%	89%	83%
Music	88%	87%	92%	89%	90%	85%
TunePro Music	93%	92%	93%	93%	92%	93%
Shazam	92%	89%	91%	91%	91%	92%
Spotify	90%	90%	93%	92%	89%	91%
ES File Explorer	89%	87%	87%	89%	86%	89%
average	89%	89%	81%	88%	90%	89%

The button is missing, it should have interacted with a component in the GUI, but the component is not found.

Information is missing; part of the information that should exist is missing.

GUI anomaly, where the GUI changes after interaction, but the GUI interface is displayed differently than it should be.

GUI display anomalies, where GUI display anomalies such as buttons unresponsive or GUI information abnormalities and missing information appear most frequently in the experiment. In addition, only a few exceptions were found for the commercial application, which may be related to the fact that it has been adequately tested, while the open-source application has more defects. No functional anomalies were found for the commercial application in the experiments, only application crashes.

V. CONCLUSIONS

The experiments conducted in this study validate the efficacy of the proposed semantic model-driven automated testing approach. By utilizing the domain semantic model as the core, this approach enhances the reusability of the testing model and introduces a new perspective on mobile application testing. Furthermore, it facilitates a completely non-invasive testing approach, particularly in the robot vision environment. To address the shortcomings of current mobile application functional testing, a semantic model-driven automated testing approach is proposed. The paper investigates an extended semantic model-driven automated testing method, based on a domain model of mobile applications. It first explores the states of the tested application with the goal of achieving maximum reachable states, thereby establishing an extended semantic FSM-ES model. Subseque-ntly, based on the domain model's action flowchart, the FSM-ES model is extended and mapped to a task subgraph with feasible paths as the goal, aiming to cover application functionality. This modeling of the tested application is accomplished from two perspectives: the GUI state reachability relationships (FSM-ES) and feasible paths between GUI states (task subgraph).Following this, by defining semantic coverage-oriented testing criteria, the goal is to achieve the broadest path coverage within the task subgraph. This process generates test cases targeting application functionality. Through testing verification in various application domains such as aviation services, among 13 discovered defect categories totaling 34 defects, the test cases generated by the semantic testing model achieved defect detection rates of 70.6% in the robot's visual environment and 82.4% in a simulated environment. Moreover, the semantic model-generated test cases were able to simultaneously detect application crashes and functional anomalies, supporting complex automated testing of functionalities with strict requirements for behavior sequences and test inputs.

To address the shortcomings of current mobile application functional testing, a semantic model-driven automated testing approach is proposed. The paper investigates an extended semantic model-driven automated testing method, based on a domain model of mobile applications. It first explores the states of the tested application with the goal of achieving maximum reachable states, thereby establishing an extended semantic FSM-ES model. Subsequently, based on the domain model's action flowchart, the FSM-ES model is extended and mapped to a task subgraph with feasible paths as the goal, aiming to cover application functionality. This modeling of the tested application is accomplished from two perspectives: the GUI state reachability relationships (FSM-ES) and feasible paths between GUI states (task subgraph). Following this, by defining semantic coverage-oriented testing criteria, the goal is to achieve the broadest path coverage within the task subgraph. This process generates test cases targeting application functionality. Through testing verification in various application domains such as aviation services, among 13 discovered defect categories totaling 34 defects, the test cases generated by the semantic testing model achieved defect detection rates of 70.6% in the robot's visual environment and 82.4% in a simulated environment. Moreover, the semantic model-generated test cases were able to simultaneously detect application crashes and functional anomalies, supporting complex automated testing of functionalities with strict requirements for behavior sequences and test inputs.

References

[1] Tramontana P, Amalfitano D, Amatucci N, et al. Automated functional testing of mobile applications: a systematic mapping study [J]. Software Quality Journal, 2019, 27(1):149-201.

[2] Kong P, Li L, Gao J, et al. Automated testing of android apps: A systematic literature review [J]. IEEE Transactions on Reliability, 2018, 68(1): 45-66.

[3] Cruz L, Abreu R, Lo D. To the attention of mobile software developers: guess what, test your app! [J]. Empirical Software Engineering, 2019, 24(4): 2438-2468.

[4] Wimalasooriya C, Licorish S A, da Costa D A, et al. A systematic mapping study addressing the reliability of mobile applications: The need to move beyond testing reliability [J]. Journal of Systems and Software, 2022, 186: 111166.

[5] Al-Subaihin A A, Sarro F, Black S, et al. App store effects on software engineering practices [J]. IEEE Transactions on Software Engineering, 2019, 47(2): 300-319.

[6] Luo C, Goncalves J, Velloso E, et al. A survey of context simulation for testing mobile context-aware applications [J]. ACM Computing Surveys, 2020, 53(1): 1-39.

[7] Amalfitano D, Amatucci N, Memon A M, et al. A general framework for comparing automatic testing techniques of Android mobile apps [J]. Journal of Systems and Software, 2017, 125(c): 322-343.

[8] Linares-Vásquez M, Bernal-Cárdenas C, Moran K, et al. How do developers test android applications?[C]//2017 IEEE International Conference on Software Maintenance and Evolution (ICSME). IEEE, 2017: 613-622.

[9] Li N, Offutt J. Test oracle strategies for model-based testing[J]. IEEE Transactions on Software Engineering, 2016, 43(4): 372-395.

[10] Banerjee I. Advances in model-based testing of GUI-based software[M]//Advances in Computers. Elsevier, 2017, 105: 45-78.

[11] Automator[EB/OL], https://developer.android.com/training/testing/ui-automator, 2020-3. Ngo C D, Pastore F, Briand L. Automated, cost-effective, and update-driven app testing [J], ACM Transactions onSoftware Engineering and Methodology, 2022, 31(4):1-51.

[12] Gu T, Sun C, Ma X, et al. Practical GUI testing of Android applications via model abstraction and refinement[C]//2019 IEEE/ACM 41st International Conference on Software Engineering (ICSE). IEEE, 2019: 269-280.

[13] Ghorbani N, Jabbarvand R, Salehnamadi N, et al. DeltaDroid: Dynamic Delivery Testing in Android [J]. ACM Transactions on Software Engineering and Methodology, 2022.

[14] Hasan H, Ladani B T, Zamani B. MEGDroid: A model-driven event generation framework for dynamic android malware analysis [J]. Information and Software Technology, 2021, 135:106569.

[15] Pan M, Lu Y, Pei Y, et al. Effective testing of Android apps using extended IFML models [J]. Journal of Systems and Software, 2020, 159:110433.

[16] Perera A, Aleti A, Böhme M, et al. Defect prediction guided search-based software testing[C]//2020 35th IEEE/ACM International Conference on Automated Software Engineering (ASE). IEEE, 2020:448-460.

[17] Su T, Meng G, Chen Y, et al. Guided, stochastic model-based GUI testing of Android apps [C]//Proceedings of the 2017 11th Joint Meeting on Foundations of Software Engineering. 2017:245-256.

[18] Zhong B, Wu H, Li H, et al. A scientometric analysis and critical review of construction related ontology research [J]. Automation in Construction, 2019, 101:17-31.

[19] Web Ontology Language (OWL)[EB/OL]. https://www.w3.org/OWL, 2012-12/2022-9.

[20] Li Y, Yang Z, Guo Y, et al. Humanoid: a deep learning-based approach to automated black-box Android app testing [C]//2019 34th IEEE/ACM International Conference on Automated Software Engineering (ASE). IEEE, 2019:1070-1073.

[21] Wu X, Sahoo D, Hoi S C H. Recent advances in deep learning for object detection [J]. Neurocomputing, 2020, 396:39-64.

[22] Deka B, Huang Z, Franzen C, et al. Rico: A mobile app dataset for building

[23] Data-driven design applications [C]//Proceedings of the 30th Annual ACM Symposium on User Interface Software and Technology. 2017:845-854.

[24] Ioffe S, Szegedy C. Batch normalization: Accelerating deep network training by reducing internal covariate shift [C]//International conference on machine learning. PMLR, 2015:448-456.

Design and Implementation of Smart Home System Based on STM32 Microcomputer

Gao Ma
State Key Laboratory of Space Microwave Technology, China Academy of Space Technology
Xi'an, China
E-mail: gm02@126.com

Jiyao Fan
School of Electronic Engineering
Xi'an Aeronautical Institute
Xi’an, China
E-mail: fanjiyao@163.com

Lulu Chen
School of Mechanical Engineering
Xihua University,
Chengdu, China
E-mail: chenll1232022@163.com

Jiaxuan Liu
College of Communication and information engineering, Xi'an University of Posts and Telecommunications
Xi’an, China
E-mail: liujiaxuan_tuan@163.com

Yixiao Wang
School of Mecidine, Huaqiao University
Quanzhou, China
E-mail: 3095779155@qq.com

Lei Tian
School of Electronic Engineering, Xi'an University of Posts and Telecommunications
Xi’an, China
E-mail: tla02@126.com

***Abstract*—With the rapid development of the Internet of Things science and technology, people's living standards are gradually improving, and the requirements for the living environment are also getting higher and higher, which makes smart homes gradually enter thousands of households. The purpose of this project is to design a system that integrates hardware and software and can measure and transmit various data. Among them, the hardware part includes data measurement and data display. The data measurement module consists of DHT11 temperature and humidity sensor, DSM501 particle number sensor and MQ3 alcohol concentration sensor. The experimental data will be displayed on the TFTLCD screen. The system software is partly run on the Windows operating system, using the Python language development. This system takes ESP8266 module as the transfer station, realizes the communication between STM32 development board and computer. The experiment shows that the system has the advantages of high measurement data accuracy, fast data refresh speed, complete data transmission, simple design, high reliability, easy installation, economical and practical, and has certain practical value in life, production, industry and other fields.**

Keywords-Smart Home; STM32 Development Board; ESP8266; DHT11 Temperature and Humidity Sensor

I. INTRODUCTION

Smart home is an ideal lifestyle for many people for a long time, and with the support of modern information technology, it has finally become a reality [1-3]. The rapid development of the Internet has provided new ideas and directions for the smart home market, opening a new era of intelligent "Internet of things + home products". With the continuous improvement of people's requirements for home life, the application of smart home products is born, through the in-depth mining of diversified needs of users, the use of existing technologies of the Internet of things and home products to link and integrate, can make smart home products more intelligent, humanized and convenient, meet people's experience and emotional needs of smart home products, improve people's living quality [4-5]. This paper designs a system which integrates hardware and software

DOI: 10.2478/ijanmc-2023-0072

and can measure and transmit various data. The system can obtain the information about the external environment in real time, display it on the single chip microcomputer display, and send the data to the communication equipment in real time [6-7]. The smart home control system can greatly improve the comfort and reliability of people's home environment, so as to improve people's living standards [8].

II. RELATED WORKS

Smart home has become a popular trend in foreign markets [9-10]. In 1984, the first "intelligent building" in the United States opened the prelude to smart homes. In recent years, developed countries such as Europe, the United States, and Japan have constantly innovated in the research and application of smart home technology. For example, Google's smart Home system Google Home, Apple's HomeKit, etc., can realize intelligent control of home products, such as smart light bulbs and smart sockets. Germany Berlin Racecourse district completed the "AAL" system model room, AAL system used to improve the life of the elderly home. It includes a scalable intelligent technology platform on which a wide range of instruments can be interconnected to build a responsive environment, profiling and responding to customer situations and environmental objectives. Spain built one of the most advanced smart houses in Europe in 2005, which focuses on living in harmony with nature to meet people's basic needs. Intelligent lighting is used in the room, which can automatically open and close the light source according to the sunshine condition, saving energy. In addition to the smart home, the roof of the smart house is also equipped with a temperature monitoring system to monitor temperature and weather conditions.

Compared with foreign markets, the smart home industry in the domestic market started late. Smart home in China about the rise of the late 1990s, and then entered a period of rapid development [10-15]. In recent years, with the wide use of single-chip microcomputer control technology, the design and application of smart home control system based on single-chip microcomputer has been effectively developed. Through the control function of embedded single chip microcomputer, the intelligent management function of residential buildings can be realized effectively. At present, home intelligence and property management, security, a variety of information services and management combined to provide high-tech intelligent means for the service and management of residential communities, in order to achieve fast and efficient value services and management, to provide a more safe and comfortable home environment. With the continuous enhancement of chip computing power and the large-scale application of 5G communication, cloud platform construction technology, video and audio AI algorithm technology, and product intelligence technology will all empower the smart home industry.

III. OVRALL DESIGN

This system mainly realizes data measurement, data communication and data processing. The data is measured by the DHT11 temperature and humidity sensor, the MQ3 alcohol concentration sensor and the DSM501A particle number sensor. Data communication is realized by ATK-ESP8266 serial port wireless communication module. Data processing and data reception are implemented using the Python programming language. Therefore, this system is divided into hardware design and software design.

A. Overall system design

The overall framework of the system is shown in Figure 1.

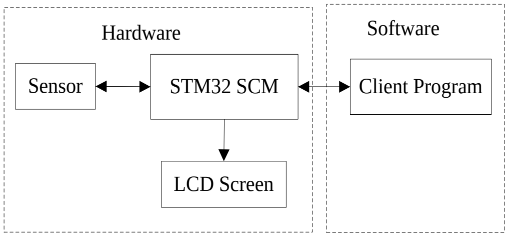


Figure 1. Overall system architecture

As can be seen from the figure above, the system is composed of two parts, namely hardware and software. The hardware part is composed of STM32 MCU, sensor and TFTLCD display. The software part is a client application developed by the Python language.

B. Data measurement module

The hardware part of the data measurement module uses a total of three sensors. The system uses DHT11 sensor to measure temperature and humidity, DHT11 sensor output TTL high and low level, STM32 through a GPIO pin to measure; The system uses MQ3 sensor to measure alcohol concentration, MQ3 sensor outputs analog data, STM32 is measured by ADC peripheral; The system uses DSM501 to measure the number of PM2.5 particles, DSM501 to output pulse signals.

C. Data Transmission Module

The data transmission module transmits data through the ATK-ESP8266 module. ESP8266 module built by TCP/IP protocol stack, can send data over the wireless network. This module can set three modes, this system uses AP mode, which is equivalent to router. The module communicates with STM32 microcontroller through serial port, and communicates with computer through network.

D. Software module

The function of the software part mainly realizes three functions, data acceptance, data display and data preservation. Data acceptance is to accept the data sent by the ESP8266 module; Data display is to display the received data in accordance with the format on the client interface; Data saving is to save the data displayed by the client in a file.

IV. SYSTEM DESIGN AND IMPLEMENTATION

A. Hardware

1) Temperature and Humidity measurement module

The DHT11 sensor transmits DATA through the TTL level of the data pin. Use this sensor to transmit data "0" and "1". When the sensor sends data "0", it starts with a low level for a duration of about 12 to 14 μ s, followed by a high level for a duration of about 28 to 28 μ s. When data "1" is sent, it also starts with a low level for a period of 12 to 14 μ s, followed by a high level for a period of 116 to 118 μ s. The DHT11 sensor distinguishes "0" from "1" by output a high level of different time lengths and verifies the sent data to ensure the accuracy of the data.

After the T/H sensor is initialized, the host sends a start signal and replies to the host before sending data. The size of the data sent is 40bit, 5 bytes. The first and second bytes are the integer and decimal parts of humidity, the third and fourth bytes are the integer and decimal parts of temperature, and the last byte is the check code, equal to the sum of the first four bytes. Through this byte, you can judge the accuracy of the sensor's measurement data. One complete data measurement using the DHT11 sensor.

A complete temperature and humidity data measurement can be divided into four steps. The first step is to set the sensor to the input mode, and then the STM32 host sends the start signal to the sensor, and the sensor is set to the output mode after the transmission is completed. Step 2: If a sensor sends a response signal to the host, after sending the response signal, the sensor pulls the low level to prepare to send data, and pulls the low level to notify the host that it can now receive data; The third step is to send 40bit data from the sensor to the host. The last step is to calculate whether the data received by the host is correct. If it is correct, the data is saved to the memory for other modules to use. If the data is incorrect, jump to step 1 and start over.

2) MQ3 Alcohol concentration measurement module

The MQ3 sensor has two output ports, digital output port and analog output port. The digital output port outputs a low level when the alcohol concentration exceeds a set threshold, and the analog output port converts the alcohol concentration into an analog signal output. The system uses the analog output port to detect the alcohol concentration value, and reads the analog signal through the AD conversion function of STM32 SCM ADC module. The MQ3 sensor operates at a voltage of 5V, so the conversion of analog signals into electrical signals has a range of 0~5V.

There are five steps to measure alcohol concentration in this system. The ADC was initialized before the measurement, but not enabled. In this system, the first step is to start ADC and prepare the measurement analog data. Each ADC has multiple measurement channels, and this ADC uses channel 16 to measure alcohol concentration data. Step 2 Set the ADC acquisition channel and enable it. In the third and fourth steps, 20 sets of alcohol concentration data are measured using the ADC. The final step converts the data from the analog signal into a concentration value and saves that value.

3) PM2.5 particle number measurement module

PM2.5 particle counts are measured using the DSM501A dust sensor, which is based on the principle of light reflection to measure the amount of dust. The DSM501A dust sensor can measure smoke particles produced by tobacco burning, particles produced by pollen, particles in dust in houses, and more. The output mode of the DSM501 sensor is PWM pulse modulation output, using the principle of particle counter, can detect particles above 1 micron, and the built-in heater can realize self-suction air. The DSM501 sensor works at a voltage of 5V, and the sensor can work normally at a temperature of -10℃~60℃. The data measurement tends to be stable after 1 minute after the heater power is turned on. The function of the heater is: heating causes the updraft, so that the outside air flows into the module, easy to measure.

The DSM501 sensor consists of 5 pins. Among them, pin 2 and pin 4 are its output pins, the difference between them is that the No. 2 output pin can detect particles with a diameter of more than 1μm, while the No. 4 output pin can only detect particles with a diameter of more than 2.5 μm, and the No. 1 pin exists as its control foot, when the external resistance of the No. 1 pin can adjust the sensitivity of the No. 4 pin, and the No. 3 pin is connected to the 5V power supply to work. Pin 5 is grounded. The operating voltage standard of the DSM501A is 5.0V±0.5, and the maximum operating current can be 90mA.

When the LED light source irradiates on the suspended particles in the air, the light will be scattered, and the reflected light will be received at a specific Angle and converted into an electrical signal by the photoelectric sensor. The output per unit time is shown in Figure 2.

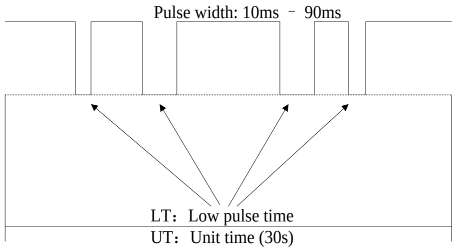


Figure 2. Output PWM waveform of DSM501 dust sensor

From the figure above, it can be seen that the output of the DSM501 sensor is a PWM waveform. When there are no dust particles, the output is about 4.5V high level, when there are dust particles, the output is about 0.7V low level. The DSM501 sensor measures the number of particles, not the concentration, and cannot be converted to each other. The DSM501 sensor calculates the number of particles by calculating the percentage of low levels per unit time, measured in units (units per liter). The measurement process of PM2.5 particle number in this system is shown in Figure 3 below.

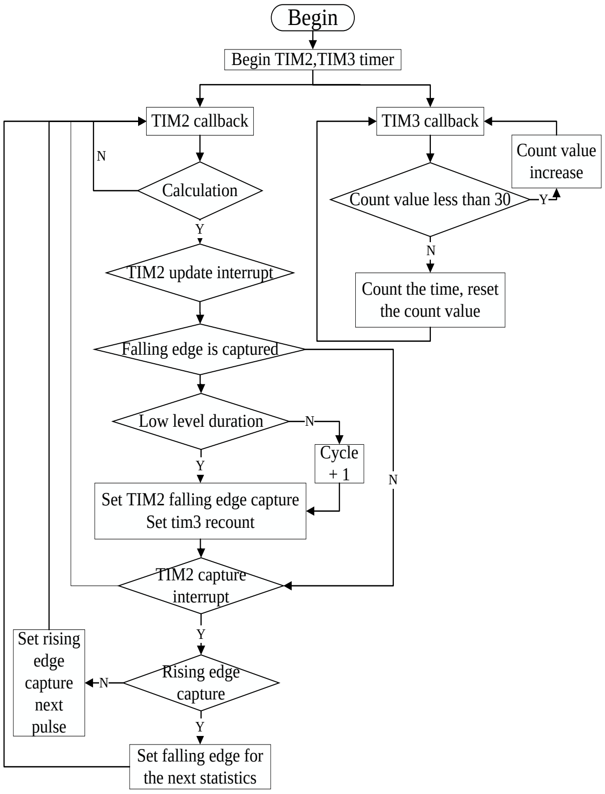


Figure 3. Flow chart of PM2.5 particle number measurement

As can be seen from the figure above, the particle count is measured by two timers. A timer is used to set the unit time, usually the unit time is set to 30S, this time is recommended by the official document of the sensor, its measurement data is relatively accurate; A timer is used to capture low-level pulses. This system uses TIM2 and TIM3 two timers. When starting, start two timers, and then enter the TIM2 or TIM3 callback function when the timer interrupt is met, calculate the length of a low-level pulse in the TIM2 callback function, and add the pulse time of each measurement. After setting the unit time length in TIM3, calculate the data result once, clear the accumulated pulse length, and start the calculation again.

4) ATK-ESP8266 Wireless transmission module

The ATK-ESP8266 sensor is a serial wireless data transmission module. ATK-ESP8266 module uses serial port and level machine for communication, the module value TCP/IP protocol stack, can realize the conversion between serial port and WIFI, only need simple configuration can transmit data through the network. ESP8266 module supports three modes, serial port to STA mode, serial port to AP mode, serial port to STA+AP mode. The AP mode provides wireless access services, allows other wireless devices to access, and provides data access. The general wireless routing/bridge works in this mode. Aps are allowed to connect to each other. STA mode is similar to the wireless terminal, STA itself does not accept wireless access, it can be connected to the AP, the general wireless card that works in this mode. STA+AP mode is both functions can be met.

The system uses AP mode. ATK-ESP8266 module is used as router + server to send the data collected on the single chip computer. Use the computer to connect to the ATK-ESP8266 router and receive data through TCP/IP protocol. Figure 4 shows the process of using ATK-ESP8266 in the system.

The ATK-ESP8266 module starts to work and exits as shown in the figure above. According to the above flow chart, check whether the ATK-ESP8266 module is successfully connected after the start. If the connection is successful after five checks, proceed to the next step. After the ATK-ESP8266 connection is successful, configure the corresponding function mode. The first step is to turn off the output and set it to AP mode. After the setting is complete, you need to restart the module before it can be used. Then set the router SSID and password, start the connection mode after the setting is complete, and set the server port. After starting the server, you can send and receive data. When you need to exit, click the KEY0 button on the MCU to exit the wireless transmission mode.

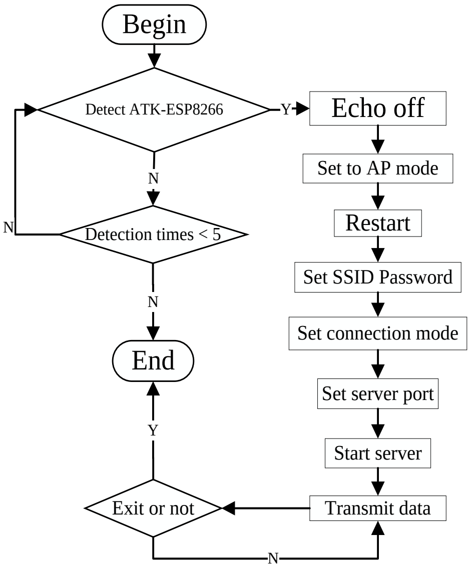


Figure 4. Flowchart of the ATK-ESP8266 wireless transmission module

B. Software part

1) System Analysis

In this system, the ATK-ESP8266 module is used as the data sender, and the computer is used as the data receiver. However, when in use, the ATK-ESP8266 module needs to be configured as AP mode, and the single chip needs to be connected to the AP module router.

The client application of this system is developed using Python language and runs on Windows operating system. Python language development software has some advantages, because the Python language is an object-oriented programming language, used to develop a small amount of program code, easy to modify, easy to understand, compatibility is better, you can run in Windows, Linux, Mac operating systems. There are also various useful Python libraries in the Python language community, such as: pandas data processing, regular expressions, network programming, graphical interface development, object-oriented programming, etc. In this chapter, we will introduce the main modules of the client application and the main Python libraries used.

2) Software Running Environment Actual system test

There are two ways to run the software, one is to run the Python script directly, the other is to generate an executable file through the PyInstall tool, and then run the executable file. The advantage of running scripts directly is that they can be run on different operating systems, but the Python interpreter and related Python libraries must be installed. The advantage of running executable files is that you do not need to install the Python interpreter, Python libraries, but can only run on Windows operating systems.

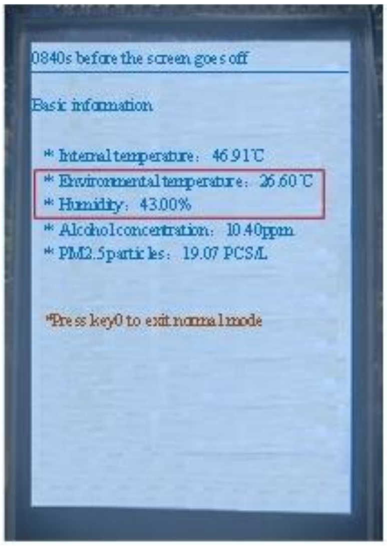
V. ACTUAL SYSTEM TEST

In addition to introducing the various functions and sensors of the system, this paper will also test the various functions of the system and analyze the measurement data of the sensor. In this chapter, DHT11 temperature and humidity measurement data, MQ3 alcohol concentration measurement data, DSM501 particle number measurement data, ATK-ESP8266 module transmission data will be analyzed, and the function of the client software will be tested.

A. Temperature and Humidity Data Measurement

After several measurements, the results measured by the DHT11 sensor are shown in Figure. 5.

The figure 5 below shows the data measurement display when the system is in common mode. The framed data are the temperature and humidity of the current region. The unit of the temperature is ℃, and the unit of the humidity is %RH.



0840s before the screen goes off

Basic information

* Internal temperature： 46.91℃
* Environmental temperature： 26.60℃
* Humidity： 43.00%
* Alcohol concentration： 10.40ppm
* PM2.5particles： 19.07 PCS/L

*Press key0 to exit normal mode

Figure 5. Measurement data of temperature and humidity

After multiple measurement records, the temperature and humidity data in a period of time are shown in Table 4.1. It can be seen from the table that the temperature measured by the DHT11 sensor is maintained between 24.50℃ and 25.30℃ for a period of time. In the local area, the temperature read by mobile phones is around 25 degrees Celsius. Through comparative analysis of the two data, it can be concluded that the temperature measured by DHT11 sensor is more accurate and the error is relatively small. As can be seen from the table, the humidity value measured by the DHT11 sensor remained between 46 and 48% over a period of time, while the humidity value read by the mobile phone in the local area was 51% over the same period of time. By comparison, the data measured by the sensor is lower than the actual data, but in general, the error is relatively small.

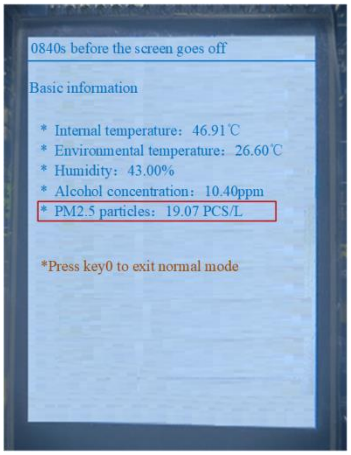
TABLE I. TEMPERATURE AND HUMIDITY MEASUREMENT DATA PROCESSING TABLE

Time	Temperature(℃)	Humidity(%)
Sat May 7 23:01:42 2022	24.90	45.00
Sat May 7 23:01:44 2022	24.90	45.00
Sat May 7 23:01:46 2022	24.90	48.00
Sat May 7 23:01:48 2022	24.90	48.00
Sat May 7 23:01:50 2022	24.90	49.00

It can be seen from the above data measurement comparison that the temperature and humidity measured by this system is relatively accurate, the data measurement results are relatively small and the data is stable, and the data measured by DHT11 sensor can meet the needs of users.

B. Alcohol concentration measurement

After measurement, the alcohol concentration measurement results are shown in Figure 6.



0840s before the screen goes off

Basic information

* Internal temperature: 46.91℃
* Environmental temperature: 26.60℃
* Humidity: 43.00%
* Alcohol concentration: 10.40ppm
* PM2.5 particles: 19.07 PCS/L

*Press key0 to exit normal mode

Figure 6. Measurement of alcohol concentration

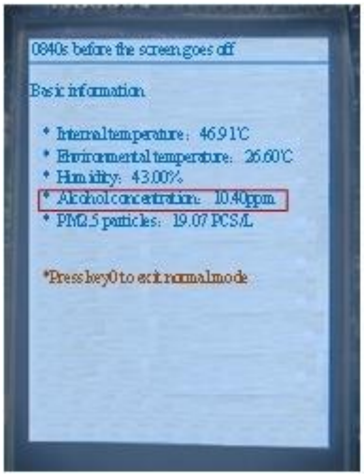
The alcohol concentration was measured indoors and the measurement result was 10.40ppm. After several measurements, the alcohol concentration data is shown in Table 2 below. The alcohol concentration was maintained between 7.28 and 12.19ppm, and the measurement data was relatively stable.

TABLE II. ALCOHOL CONCENTRATION DATA MEASUREMENT TABLE

	Time	Alcohol concentration(ppm)
Sat May	7 23:01:42 2022	9.36
Sat May	7 23:01:44 2022	12.12
Sat May	7 23:01:46 2022	8.29
Sat May	7 23:01:48 2022	7.68
Sat May	7 23:01:50 2022	8.22

C. PM2.5 particle number measurement

Through measurement, the measurement results of PM2.5 particle number are shown in Figure 7.



0840s before the screen goes off

Basic information

* Internal temperature: 46.91℃
* Environmental temperature: 26.60℃
* Humidity: 43.00%
* Alcohol concentration: 10.40ppm
* PM2.5 particles: 19.07 PCS/L

*Press key0 to exit normal mode

Figure 7. Measurement value of PM2.5 particle number

This measurement was made in an indoor environment. According to the above figure, the number of PM2.5 particles tested is 19.07 per liter. After repeated measurements, the measurement results of PM2.5 particle number are shown in Table 3, where the value of PM2.5 particle number is maintained between 5.80 and 21.47 per liter. The output data of the sensor is stable and the data change is small.

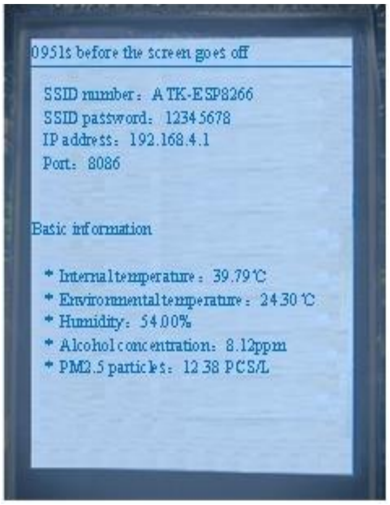
TABLE III. PM2.5 PARTICLE COUNT MEASUREMENT TABLE

Time	PM2.5 concentration (per/ liter)
Sat May 7 23:01:42 2022	7.96
Sat May 7 23:01:44 2022	7.25
Sat May 7 23:01:46 2022	7.25
Sat May 7 23:01:48 2022	7.25
Sat May 7 23:01:50 2022	9.84

D. ATK-ESP8266 Wireless transmission test

In the wireless transmission module, the data is sent through the ATK-ESP8266, and the data is received through the computer client application. The application is developed using the Python language and receives data via TCP/IP network programming.

Test the data sending function of ESP8266 wireless module and the data receiving function of client software. The wireless transmission mode on STM32 MCU is shown in Figure 8.



0951s before the screen goes off

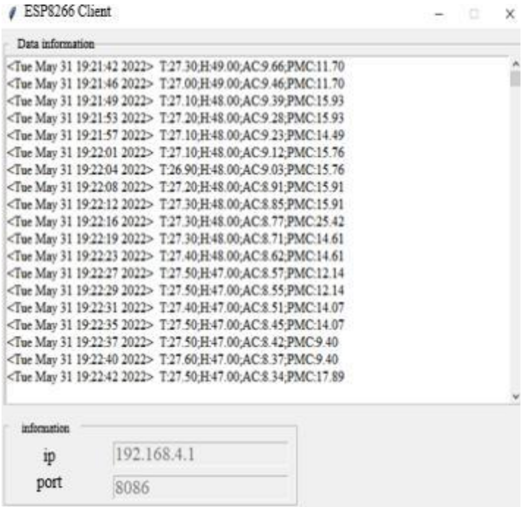
SSID number： ATK-ESP8266
SSID password： 12345678
IP address： 192.168.4.1
Port： 8086

Basic information

- Internal temperature： 39.79℃
- Environmental temperature： 24.30℃
- Humidity： 54.00%
- Alcohol concentration： 8.12ppm
- PM2.5 particles： 12.38 PCS/L

Figure 8. Wireless transmission interface

In the wireless transmission mode, the corresponding data is calculated by STM32 MCU and sent to the client software through the ESP8266 module. The format of sending data is "T: temperature; H: Humidity; AC: Alcohol concentration; PMC: Number of particles "where T stands for temperature, H for humidity, AC for alcohol concentration, and PMC for number of particles. After the client receives the data, the received data and the received time are displayed in the data information area. The data received by the client is shown in Figure 9.



Data information

```
<Tue May 31 19:21:42 2022> T:27.30;H:49.00;AC:9.66;PMC:11.70
<Tue May 31 19:21:46 2022> T:27.00;H:49.00;AC:9.46;PMC:11.70
<Tue May 31 19:21:49 2022> T:27.10;H:48.00;AC:9.39;PMC:15.93
<Tue May 31 19:21:53 2022> T:27.20;H:48.00;AC:9.28;PMC:15.93
<Tue May 31 19:21:57 2022> T:27.10;H:48.00;AC:9.23;PMC:14.49
<Tue May 31 19:22:01 2022> T:27.10;H:48.00;AC:9.12;PMC:15.76
<Tue May 31 19:22:04 2022> T:26.90;H:48.00;AC:9.03;PMC:15.76
<Tue May 31 19:22:08 2022> T:27.20;H:48.00;AC:8.91;PMC:15.91
<Tue May 31 19:22:12 2022> T:27.30;H:48.00;AC:8.85;PMC:15.91
<Tue May 31 19:22:16 2022> T:27.30;H:48.00;AC:8.77;PMC:25.42
<Tue May 31 19:22:19 2022> T:27.30;H:48.00;AC:8.71;PMC:14.61
<Tue May 31 19:22:23 2022> T:27.40;H:48.00;AC:8.62;PMC:14.61
<Tue May 31 19:22:27 2022> T:27.50;H:47.00;AC:8.57;PMC:12.14
<Tue May 31 19:22:29 2022> T:27.50;H:47.00;AC:8.55;PMC:12.14
<Tue May 31 19:22:31 2022> T:27.40;H:47.00;AC:8.51;PMC:14.07
<Tue May 31 19:22:35 2022> T:27.50;H:47.00;AC:8.45;PMC:14.07
<Tue May 31 19:22:37 2022> T:27.50;H:47.00;AC:8.42;PMC:9.40
<Tue May 31 19:22:40 2022> T:27.60;H:47.00;AC:8.37;PMC:9.40
<Tue May 31 19:22:42 2022> T:27.50;H:47.00;AC:8.34;PMC:17.89
```

information

ip 192.168.4.1
port 8086

Figure 9. Wireless transmission interface

According to the figure above, after the client connects to the router simulated by STM32 and successfully logs in to the server, it begins to receive data. The received data is displayed in the Data Info area of the client interface, and the received data time is added. The IP address and Port of the server are displayed in the Configuration Information area on the client. There are four buttons in the Control area. The Login button is used to log in to the server. Click the "Log Out" button to stop the client receiving data and clear the server information in the "Configuration Information" area. Click the "Clear" button to clear the received data in the "Data Information" area; Click the "Output" button to save the data in the "Data Information" area to Excel in the format shown in Table 4 below.

TABLE IV. STORAGE FORMAT OF MEASUREMENT INFORMATION

Time	Temperature(℃)	Humidness(%)	Alcohol (ppm)	PM2.5 (per/liter)
Sat May 7 23:01:42 2022	24.90	45.00	9.36	7.96
Sat May 7 23:01:44 2022	24.90	45.00	12.12	7.25
Sat May 7 23:01:46 2022	24.90	48.00	8.29	7.25
Sat May 7 23:01:48 2022	24.90	48.00	7.68	7.25
Sat May 7 23:01:50 2022	24.90	49.00	8.22	9.84

Through the data in the Table4, the smart home collection unit can upload the surrounding environment to the user in real time. The user achieves the real-time control function of the home environment parameters.

VI. CONCLUSIONS

The figure in the next page shows the sample data from the water temperature and turbidity sensors. Through this period of design and development, the following work has been completed:

1) This system uses STM32 as the main control chip to develop the smart home system on the STM32MINI development board. The STM32 development board has a low-power mode with wake-up function. This system uses this function to design a switch module, which has a higher priority and can interrupt any other task being

performed. On this basis, the function of automatic screen is added, and the timer control system is used to automatically disconnect the power supply of TFTLCD display after counting to the set threshold.

2) This system can measure a lot of data, these data are more in line with people's needs. The system uses DHT11 sensor to measure temperature and humidity; Measurement of alcohol concentration using MQ3 sensor; The PM2.5 particle count was measured using the DSM501 sensor.

3) This system has two main modes, ordinary mode and wireless transmission mode. In normal mode, the measured data is displayed on the TFTLCD display. In wireless transmission mode, the measured data is sent to the computer through the built-in TCP/IP protocol through the ATK-ESP8266 module.

4) The innovation of this system lies in the use of the popular Python language to develop applications. The Python language is maliciously compatible with many platforms and can be easily ported to Windows, Linux, and Mac operating systems. Python supports both procedural and object-oriented programming. Python is open source software, so the community is relatively complete, and various problems can be discussed and discussed in the community. This time it was tested on the Windows operating system, and the test results show that it can meet all task requirements.

ACKNOWLEDGMENT

This work was partly supported by the innovation and entrepreneurship training program for undergraduate plan 202311664058 and the Graduate Education and Teaching Reform Project of Xi'an University of Posts and Telecommunications (YJGJ2023035).

REFERENCES

[1] P Qian, Y Z Zhang, and Y Li, “Design of Voice Control System for Smart Home Based on STM32,” Applied Mechanics & Materials, vol. 734, Feb. 2015, pp. 369-374, doi:10.4028/www.scientific.net/AMM.734.369.

[2] H Ping, and C. Tang, “Indoor detection based on MLX90621 infrared sensor,” Electronic Measurement Technology, vol. 39, Aug. 2016, pp.118-121, doi: 10.19651/j.cnki.emt.2016.08.025.

[3] F Tian, and X Long, “Design of smart home system based on basic radio frequency wireless sensor network,” International Journal of Online Engineering,vol. 14, Apr. 2018, pp.126-136, doi:10.3991/ijoe.v14i04.8389.

[4] M Liao, GG Nie, J Zhao, and X He, “Design of a Baseband Signal for the 406 MHz Satellite Emergency Radio Transmitter Based on STM32,” ELECTRONICS, vol. 12, Jun. 2023, doi:10.3390/electronics12122717.

[5] M Z Li, M Song, and L Gao, “Design of Smart Home System Based on Zigbee,”Applied Mechanics and Materials, vol. 635-637, Sep. 2014, pp.1086-1089, doi:10.4028/www.scientific.net/AMM.734.369.

[6] M W Jian, and B W Hai, “Design of Smart Home Management System Based on GSM and Zigbee,” Applied Mechanics and Materials, vol. 842, Nov. 2013, pp.703-707, doi:10.4028/www.scientific.net/AMR. 842.703.

[7] V. Tiwari, A. Keskar, and N. C. Shivaprakash, “Design of an IoT Enabled Local Network Based Home Monitoring System with a Priority Scheme,” Eng. Technol. Appl. Sci. Res., vol. 7, Apr. 2017, pp. 1464-1472, doi:10.48084/etasr.1033.

[8] V. T. Huu and Monga Vishal, “Fast Low-Rank Shared Dictionary Learning for Image Classification,” IEEE Transactions on Image Processing, vol. 26, Jul. 2017, pp. 5160-5175, doi:10.1109/TIP.2017.2729885.

[9] C. L. Wu, and L. C. Fu, “Design and realization of a framework for human-system interaction in smart homes,” IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans, vol. 42, Jul. 2011, pp. 15-31, 2012, doi:10.1109/TSMCA. 2011. 2159584.

[10] RN Sonawane, AS Ghule, AP Bowlekar, and AH Zakane. “Design and Development of Temperature and Humidity Monitoring System,” Agricultural Science Digest-A Research Journal, vol. 39, Jul. 2019, pp.114-118, doi:10.18805/ag.D-4893.

[11] W. d. S. Costa, W. G. V. d. Santos, H. R. d. O. Rocha, M. E. V. Segatto and J. A. L. Silva, “Power Line Communication based SmartPlug Prototype for Power Consumption Monitoring in Smart Homes,” IEEE Latin America Transactions, vol. 19, Nov. 2021, pp. 1849-1857, doi: 10.1109/TLA.2021.9475618.

[12] X Ji Gang, “A Routing Algorithm Based on Zigbee Technology,” International Journal of Online Engineering, vol. 14, Nov. 2018, pp. 90-102, doi:10.3991/ijoe.v14i11.9515.

[13] L Quan, L Xiaorui,Q Zizhong, Z Duzhong, and X Wenjun, “An adaptive hybrid ARQ method for coexistence of ZigBee and WiFi,”vol. 10, Sep. 2017, pp. 310-319, doi:10.1504/IJHPCN. 2017.086535.

[14] I. A. Zualkernan, A. R. Al-ali, M. A. Jabbar, I. Zabalawi and A. Wasfy, “InfoPods: Zigbee-based remote information monitoring devices for smart-homes,” IEEE Transactions on Consumer Electronics, vol. 55, Aug. 2009, pp. 1221-1226, doi: 10.1109/TCE.2009.5277979.

[15] Z. Zhipeng, X Jiaoyuan, W Binquan, G Yiping, “f Hierarchically designed nanocomposites for triboelectric nanogenerator toward biomechanical energy harvester and smart home system,” Nano Energy, vol. 95, Feb. 2022, pp. 107047, doi:10.1016/j.nanoen. 2022. 107047.

Deep Learning Based Recognition of Lepidoptera Insects

Chao He
School of Computer Science and Engineering
Xi'an Technological University
Xi’an, 710021, China
E-mail:1061410265@qq.com

Pingping Liu
School of Computer Science and Engineering
Xi'an Technological University
Xi’an, 710021, China
E-mail: 1341369601@qq.com

***Abstract*—The successful application of cutting-edge computer vision technology to automatic insect classification has long been a focus of research in insect taxonomy. The results of this research have a wide range of applications in areas such as environmental monitoring, pest diagnosis and epidemiology. However, there is still a gap between the current techniques used in automatic insect classification and the latest computer vision techniques. The research in this paper is conducted on Lepidoptera, a class of insects that are widely infested, including butterflies and moths. The study focuses on the application of deep learning algorithms in image processing of Lepidoptera insects. In order to improve the recognition rate for Lepidoptera insect recognition, this paper uses a detection model based on deep neural networks to realize the recognition of Lepidoptera insects in complex environments. Specifically, the yolov7 algorithm is adopted as the basic model for this experiment, and the reasons for using this model are explained in terms of the splicing of network modules, loss function, positive sample allocation strategy, and the merging of convolution and normalization, respectively. Through experiments, it is proved that the algorithm can effectively improve the gesture recognition rate, the recognition accuracy reaches 79.5%, and the recognition speed is as high as 33.08it/s.**

Keywords-Deep Learning; Deep Neural Networks; Yolov7; Recognition of Lepidopteran Insects

I. INTRODUCTION

Agriculture has always been a major development industry in China, and the upgrading and transformation of agricultural technology is one of the important elements in the modernization of science and technology. Especially in the fields of plantation and forestry, which are important components of agricultural production and sustainable ecological environment construction, there is an urgent need for various technological innovations. Lepidoptera is one of the main pests faced by plantation and forestry and one of the most numerous pest categories, including moths and butterfly insects.

Effective and accurate identification of Lepidoptera is crucial for pest control of agricultural and forestry crops. In the past, monitoring of lepidopteran insects was usually done by manual multi-point sampling and relying on naked eye observation and subjective experience for identification; however, there are some problems with this method of manual judgment. Image judgment relying on the human eye is prone to misjudgment. Moreover, the results of manual judgment are not only unstable in accuracy, but also cannot be carried out dynamically in real time, which also wastes a lot of human and financial resources. Therefore, it is necessary to introduce computer technology and deep learning and other techniques for lepidopteran insect recognition. Through techniques such as image processing, feature extraction and pattern recognition [1], automated and accurate insect recognition can be realized. This method not only improves the accuracy and stability of recognition [2], but also realizes real-time dynamic monitoring and saves human and financial resources to meet the needs of actual production.

In recent years, with the development of deep learning technology, deep learning methods, such as convolutional neural networks, have also been applied in China to achieve more accurate and automated insect recognition. These methods can automatically learn the features of insects from raw images through end-to-end training, and perform classification and recognition. However,

DOI: 10.2478/ijanmc-2023-0073

in practical application scenarios, the detection speed of the target is more important than the detection accuracy of the target. Since Lepidoptera insects belong to the target detection of small objects and they are in constant motion, it is necessary to constantly capture the position of the insect in order to detect it accordingly. Therefore, in this paper, YOLOv7, which has been performing better in the field of target detection recently, is used as the network structure to complete the research on the recognition of lepidopteran insects.

II. Related Works

The country has paid more and more attention to the development of agriculture. Target detection of insects is a hot research category in the field of computer vision combined with agriculture. This kind of research can be applied to the cultivation of agricultural products, environmental detection, agricultural insects and pest prevention and other disciplines. It can effectively save the time of agricultural researchers and better understand the benefits and hazards of insects on agricultural products at different times and environments, which is of great significance to promote agricultural development and pest control. In addition, although computer vision [3] based insect identification [4] is available today, most of its applications still lag behind the cutting-edge computer vision technology.

Traditional insect recognition algorithms mainly classify and identify insects by their color, stripe, shape and other features [5]. However, the actual effect of such algorithms is not good, because the variety of insects is very large, and some insects have very similar appearance characteristics, which makes it easy to recognize insects into the wrong category. Although the target detection algorithm based on deep learning has been developed for nearly a decade, most of its applications are used in traditional industry and manufacturing, based on the natural ecology and agriculture is still in a developmental stage, in order to better develop agriculture on a large scale, so the use of computer technology to assist in the development of agriculture, the environment and other fields is very necessary.

In recent years, artificial intelligence has developed and made significant breakthroughs in the field of target detection [6], and a variety of network structures have appeared, which have shown good performance in both large-volume and microbial target detection. Most of the network structures are based on deep learning for deep convolution of images [7], extracting large features from large images[8], and then downsampling on the basis of large images to extract features based on small images. Finally the features are fused and the positive sample with minimum loss function is selected as the final output value.

III. Introduction to Algorithms

A. Introduction to Target Detection Algorithms

Target detection algorithms are mainly categorized into two types, one is traditional target detection algorithms and the other is deep learning based target detection algorithms [9]. Deep learning based target detection algorithms are further divided into two stages, One-Stage and Two-Stage. Two-Stage has better accuracy, such as the RCNN algorithm [10], whose main method of target detection is to find out a bunch of similar targets by selective search first, and then classify the similar targets. And One-Stage has faster speed, such as YOLO algorithm [11], which mainly solves the regression problem, unlike Two-Stage which solves the classification problem. The main purpose of this paper is to detect the target of Lepidoptera in a complex environment, which requires high real-time monitoring, so the YOLOv7 algorithm, which has a faster detection speed, is used in this paper.

B. Network structure of YOLOv7

YOLOv7 is proposed in 2022 based on YOLOv4 and YOLOv5 [12], which is the current target detection algorithm with high detection speed and detection accuracy in the field of target detection. Its model framework mainly consists of four modules: input, backbone network, feature fusion, and prediction head, as shown in Fig.1, this model can better realize real-time target detection.YOLOv7 model, as a One-Stage target detection algorithm, can be processed by direct regression once to obtain the target area, location information, and category of the corresponding object. Compared with the Two-Stage target detection algorithm, it can locate the target area more quickly and its accuracy is relatively balanced, which lays a good foundation for the recognition of lepidopteran insects in this paper.

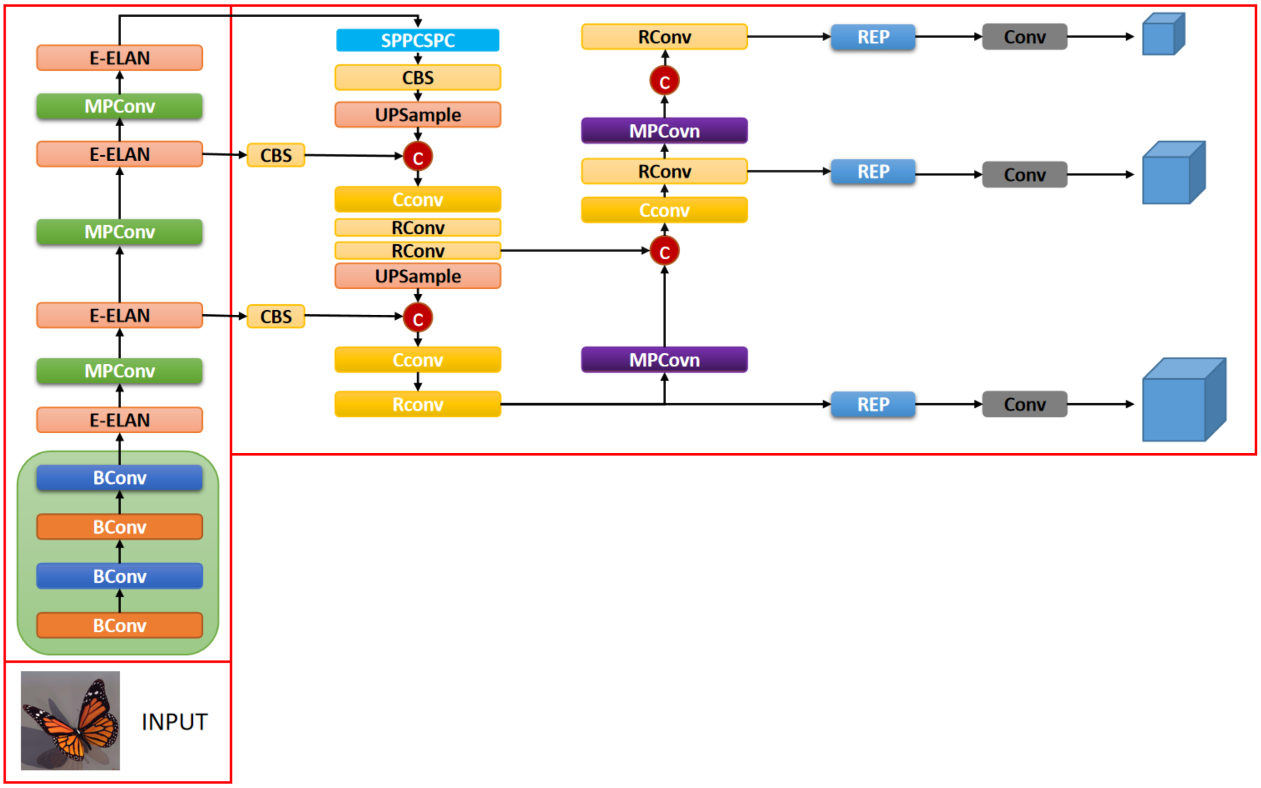


Figure 1. Network structure of YOLOv7.

C. Feature Splicing

The YOLOv7 network structure follows the YOLOv5's feature map splicing, which is performed in both the input network module and the output network module. As shown in Fig. 1, a large number of E-ELAN and MPConv layers are embedded in the network structure after image input, and the main operation of these layers is to perform feature map splicing. The advantage of this is that more comprehensive feature information can be obtained in the training phase, making the trained feature information richer. Since there are many splicing operations in this network structure, this paper mainly introduces the E-ELAN layer and MPConv layer.

In ordinary convolutional structures, basically the results of the previous layer convolution are used as the input value of the next layer convolution, as shown in Fig. 2, while in the E-ELAN layer of YOLOv7, the results of the convolution of the first 1, 3, 5, and 6 layers are merged, and the value is used as the input value of the next layer convolution. The purpose of this operation is that in the training process, it is not certain that the results of the upper layer of convolution must be better than the results of the previous convolution, so a few results of the upper layer of convolution are merged and used as the input value of the lower layer, which makes the feature maps more comprehensive and avoids the loss of important features in the downward training.

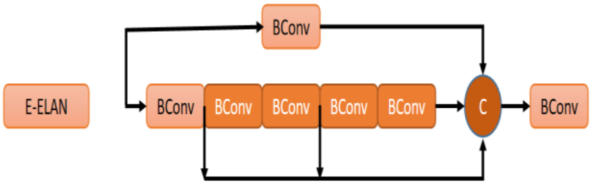


Figure 2. Network structure of the E-ELAN layer.

In the pooling layer, there is also a splicing operation similar to E-ELAN. For ordinary network structures, basically only MaxPooling is used in the pooling layer to compress their feature maps. However, in the YOLOv7 network model, the same feature map is compressed twice. As shown in Fig. 3, the first time the feature map is convolved after MaxPooling. Since convolution of larger feature maps with smaller convolution kernels can also result in feature map compression, 2 convolution kernels are used in the second compression to compress the original feature map. After that, the feature maps compressed by the above two methods are merged. This operation can make the compressed feature map more balanced than the traditional pooling operation without knowing which method is better for compressing the feature map.

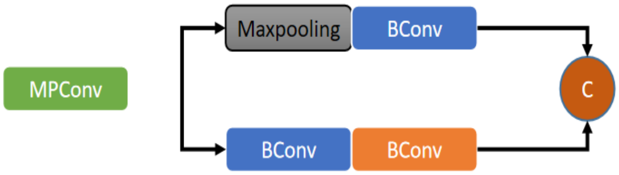


Figure 3. Network structure of the MPConv layer.

D. Loss function

This studyThe loss function used in this experiment is the loss function of YOLOv7 model, which mainly includes three parts. The first part is the target category lossclass loss, this part of the loss function is used to calculate the prediction accuracy of the model on the target category, generally use the cross-entropy loss function to calculate the prediction error of the target category, the formula is shown in equation 1. The second part is the bounding box position loss box loss, this part of the loss function is used to calculate the model's prediction accuracy of the target bounding box position, usually using the Smooth L1 loss function Smooth L1 Loss to calculate the positional difference between the predicted bounding box and the real bounding box, and its formula is shown in equation 2. The third part of the target confidence loss object loss, this part of the loss function is used to calculate the prediction accuracy of the model on the target confidence, usually using the two-classified cross-entropy loss function to categorize the target and the background, and its formula is shown in equation 3.

$$L = -\sum i = 1n \left[yi * ln \ \left(pi \right) + \ \left(1 - yi \right) * ln \ \left(1 - pi \right) \right] \quad (1)$$

$$SmoolthL_1 = \begin{matrix} |x| - 0.5, |x| > 1 \\ 0.5x^2, |x| < 1 \end{matrix} \quad (2)$$

$$Loss = - \ i = \sum_1^n \left(yilog \left(yi \right) + \ \left(1 - yi \right) log \left(1 - \ yi \right) \right) \quad (3)$$

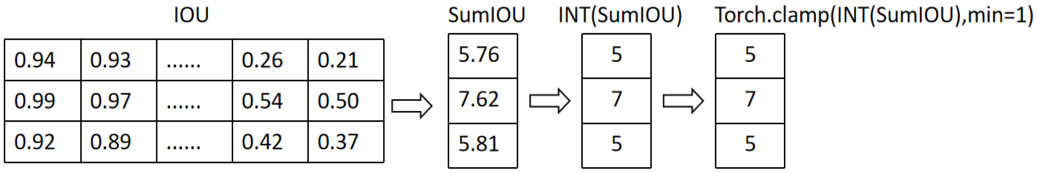
When calculating the total loss function, the loss values of the three parts need to be weighted and summed, where the weights of the target category loss and the bounding box position loss can be adjusted according to the actual task requirements. By using these three parts of the loss function, YOLOv7 can optimize the target category prediction, bounding box position prediction and target confidence prediction at the same time, thus improving the accuracy of target detection. Compared to the YOLOv5 model loss function module, YOLOv7 may improve the comprehensive performance by making improvements to the loss function. It may optimize the weights of the loss function, the way it is combined, or the calculation method based on the experience and feedback from previous versions to further improve the detection and localization capabilities of the model.

E. Positive sample allocation strategy

The more the number of positive samples for target detection, the better it is for model training. However, in most networks for target detection, the number of negative samples is much larger than the number of positive samples due to the different allocation strategies [13]. For example, in YOLOv3, YOLOv4 versions, after the ground truth is IOU'd with each anchor, the anchor with the largest IOU value is selected as a positive sample candidate. The number of candidate frames in an anchor is fixed, so it can only select the appropriate positive samples from the candidate frames in this anchor, which leads to the number of positive samples is too small, and the results of model training may not be stable enough. In YOLOv7, when selecting the positive samples, firstly, it continues the characteristics of YOLOv3 and YOLOv4 to select the anchor with the largest IOU value, and secondly, it will offset the ground truth by 0.5 in each of the four directions, and then it will carry out the IOU operation, and select the two anchors with the largest IOU value, which results in three anchors, and the positive samples will be selected from this anchor. The number of positive samples is expanded by selecting the 2 anchors with the largest IOU values.

After expanding the number of positive samples, not every positive sample can be the final output, so this gives rise to the positive sample allocation strategy. The allocation strategy of positive samples in YOLOv7 is mainly divided into three parts, firstly, the comparison of the aspect ratio of ground truth and anchor is screened, secondly, it is screened by IOU [14], and finally, it is screened by calculating the category prediction loss.

For the screening of aspect ratio and the screening of category prediction loss is relatively simple. So this section focuses on the computation of loss of IOU. The screening of IOU in YOLOv7 is done by adding and rounding the IOU values of all candidate boxes of an anchor, and then selecting the first integer IOU values as the values for the next screening. For example, the need to predict the number of positive samples for the 3, an anchor of the candidate box for 10, the IOU calculation will produce 10 IOU values, by adding the 10 IOU values and rounding, you can get this anchor can be the next screening of the number of candidates, as shown in Figure 4.



0.94	0.93	......	0.26	0.21
0.99	0.97	......	0.54	0.50
0.92	0.89	......	0.42	0.37

SumIOU	INT(SumIOU)	Torch.clamp(INT(SumIOU),min=1)
5.76	5	5
7.62	7	7
5.81	5	5

Figure 4. IOU loss calculation process.

F. Merging Convolution and Normalization

Since the algorithm used in this paper is One-Stage and the application scenarios are more complex, the speed of target detection is an important factor. Compared to the previous versions of YOLO, in the testing stage, it is generally after the feature convolution of the target object[15], and then each channel of the convolution of the normalization operation, such a process will need to spend nearly twice the time. However, if the feature convolution and normalization are combined into a new convolution then only one convolution operation is needed to complete the two operations, which greatly saves the testing time and thus improves the speed.

The operation to normalize the data of a batch in the RepConv layer in YOLOv7 is equation 4.

$$\hat{x}_i = \gamma\frac{x_i-\mu}{\sqrt{\sigma^2+\varepsilon}}+\beta = \gamma\frac{x_i}{\sqrt{\sigma^2+\varepsilon}}+\beta-\frac{\gamma\mu}{\sqrt{\sigma^2+\varepsilon}} \quad (4)$$

The normalization operation of the RepConv layer is disassembled to obtain equation 5.

The $\hat{F}_{C,i,j}$ in equation 5 represents the result after normalization for each channel, $F_{C,i,j}$ represents each channel, $\frac{\gamma 1}{\sqrt{\hat{\sigma}_C^2+\varepsilon}}$ represents the weights by which the normalization scales the features, and $\beta_C-\gamma_C\frac{\hat{\mu}_C}{\sqrt{\hat{\sigma}_C^2+\varepsilon}}$ represents the offsets. After observing equation 5, its form similar to the convolution form of WX+B.

$$\begin{pmatrix} \hat{F}_{1,i,j} \\ \hat{F}_{2,i,j} \\ \vdots \\ \hat{F}_{C-1,i,j} \\ \hat{F}_{C,i,j} \end{pmatrix} = \begin{pmatrix} \frac{\gamma 1}{\sqrt{\hat{\sigma}_1^2+\varepsilon}} & 0 & \cdots & \cdots & 0 & \\ 0 & \frac{\gamma 1}{\sqrt{\hat{\sigma}_2^2+\varepsilon}} & 0 \cdots & \cdots & 0 & \\ \vdots & \cdots & \ddots & \cdots & \vdots & \\ 0 & \cdots & \cdots\ \cdots & \frac{\gamma 1}{\sqrt{\hat{\sigma}_{C-1}^2+\varepsilon}} & 0 & \\ 0 & \cdots & \cdots & \cdots & \frac{\gamma 1}{\sqrt{\hat{\sigma}_C^2+\varepsilon}} & 0 \end{pmatrix} \bullet \begin{pmatrix} F_{1,i,j} \\ F_{2,i,j} \\ \vdots \\ F_{C-1,i,j} \\ F_{C,i,j} \end{pmatrix} + \begin{pmatrix} \beta_1 - \gamma_1 \frac{\hat{\mu}_1}{\sqrt{\hat{\sigma}_1^2+\varepsilon}} \\ \beta_2 - \gamma_2 \frac{\hat{\mu}_2}{\sqrt{\hat{\sigma}_2^2+\varepsilon}} \\ \vdots \\ \beta_{C-1} - \gamma_{C-1} \frac{\hat{\mu}_{C-1}}{\sqrt{\hat{\sigma}_{C-1}^2+\varepsilon}} \\ \beta_C - \gamma_C \frac{\hat{\mu}_C}{\sqrt{\hat{\sigma}_C^2+\varepsilon}} \end{pmatrix} \quad (5)$$

Since the merging of convolution and normalization is performed in the testing phase after the training is completed, not only the convolution kernel for feature convolution and each input channel can be obtained in this phase. The convolution kernel for the normalization operation and the channels that have undergone feature convolution can also be obtained. The expression for fusing feature convolution and normalization can be written as equation 6. Where $W_{conv} \bullet f_{i,j} + b_{conv}$ generation is the feature result after convolution of the input channel. Then multiplying its feature result by the normalized weight W_{BN} and adding the offset b_{BN} gives the result of normalizing the feature convolution for that channel $\widehat{f}_{i,j}$.

$$\widehat{f}_{i,j} = W_{BN} \bullet W_{conv} \bullet f_{i,j} + (W_{BN} \bullet b_{conv} + b_{BN}) \quad (6)$$

By looking at the expansion of Eq. 3 one can merge the normalized convolution kernel with the feature convolution kernel to obtain a new convolution kernel. The input channel is always unchanged. The new bias is obtained by the normalization operation. Thus the merging of convolution and normalization is done and can be written as equation 7.

$$\widehat{f}_{i,j} = W_{BN,Conv} \bullet X_{f_{i,j}} + b_{BN,Conv} \quad (7)$$

IV. EXPERIMENTAL SETUP AND ANALYSIS

After the text edit has been completed, the paper is ready for the template. Duplicate the template file by using the Save As command, and use the naming convention prescribed by your conference for the name of your paper. In this newly created file, highlight all of the contents and import your prepared text file. You are now ready to style your paper.

A. Data set construction

Since most of the lepidopteran insects belong to the pest type at present, the IDADP Agricultural Pests and Diseases Research Atlas was used in this paper. However, since it does not have a separate database for Lepidoptera, this paper adopts its butterfly and moth related datasets, and collects some related images of other Lepidoptera by web crawling. The images are labeled with LabelImg, the export type is YOLO type, and the categories are Islepidoptera and Nolepidoptera, which finally form a lepidopteran dataset with 5000 pictures.

B. Experimental platforms

The experimental server for this experiment is CPU for AMD Ryzen 9 7940H w, GPU for NVIDIA GeForce RTX 4050/PCle, operating system is windows 11, memory is 16GB, training environment is pytorch. Data is randomly divided according to 8:1:1, put into the grid for training iteration 300 epochs, batch_size16, training image size set to 640×640. The data was randomly divided according to 8:1:1, and put into the grid for 300 epochs, batch_size16, and the training image size was set to 640×640.

C. Evaluation indicators

In this experiment, in order to evaluate the performance of the model, the average accuracy mAP, recall, precision, and specific mathematical formulas are used as shown in the following equation. Where c is the number of categories, APi is the precision rate of the first category, TP is the positive sample of correctly predicted samples, and FP is the negative sample of incorrectly judged positive samples.

$$mAP = \frac{\sum_{i=1}^{c} APi}{c} \tag{8}$$

$$\mathrm{Re}\,call = \frac{TP}{TP + FN} \tag{9}$$

$$\mathrm{Pr}\,ecision = \frac{TP}{TP + FP} \tag{10}$$

The performance of this paper's YOLOv7-based algorithm is tested using the test set divided in the lepidopteran insect dataset, and comparative experiments with YOLOv5m6 and YOLOv5s6 algorithms, respectively, so as to validate the performance of this paper's YOLOv7-based algorithm in the detection and identification of lepidopteran insects, which is mainly the comparative experiments based on the test set. In this experiment, in order to evaluate the performance of the model, the average accuracy mAP, recall, precision, and specific mathematical formulas are used as shown in the following equation. Where c is the number of categories, APi is the precision rate of the first category, TP is the positive sample of correctly predicted samples, and FP is the negative sample of incorrectly judged positive samples.

After training the algorithm on YOLOv5m6 and YOLOv5s6 with the same dataset, validation set, and epoch respectively, the above algorithmic models are tested with the same test set and the average accuracy value as shown in equation 8 and the number of iterations per second of the model to process the data as shown in equation 11 are used as the evaluation metrics. Where iterationNum represents the number of this iteration and Time represents the time spent on this iteration.

$$Speed = \frac{iterationNum}{Time} \tag{11}$$

The results after testing are shown in Table 1.

TABLE I. YOLOV5M6, YOLOV5S6, YOLOV7 PERFORMANCE COMPARISON

Arithmetics	**mAP/%**	**Speed**
YOLOv5m6	78.6%	22.36it/s
YOLOv5s6	75.8%	24.69it/s
YOLOv7	79.5%	33.08it/s

A comparison of the YOLOv5m6 and YOLOv5s6 algorithmic models shows that YOLOv7 has improved its average accuracy. In YOLOv5, the convolution and normalization operations were performed separately to generate nearly twice the time for these two operations, and YOLOv7 achieved the merger of convolution and normalization in the test phase, so its iteration speed has been greatly improved, so that it can be effectively used in complex real-world scenarios.

D. Loss and accuracy analysis

Because the background of lepidopteran insect recognition in the actual environment is generally more complex, and there may be multiple insects interfering with the recognition in the same background. By observing the graphs of precision and recall, it can be found that the algorithm model has a good performance in both the training set and the validation set, which proves that the algorithm model has a high practicability in real scenarios.

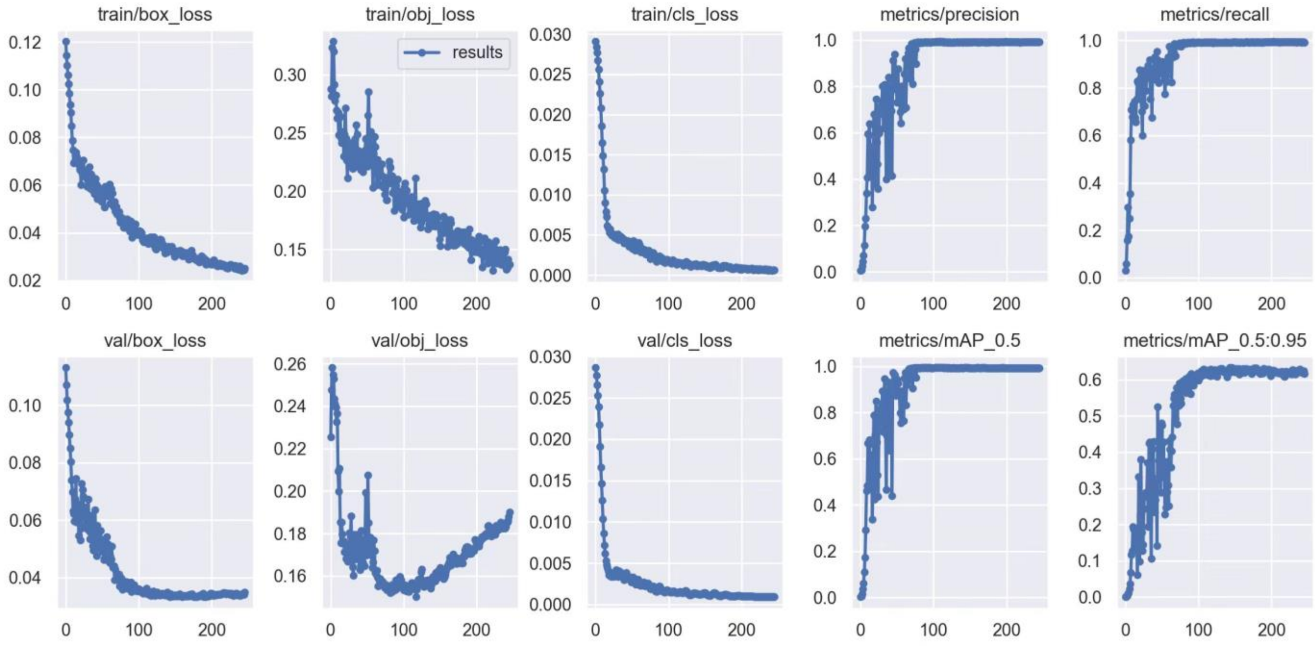
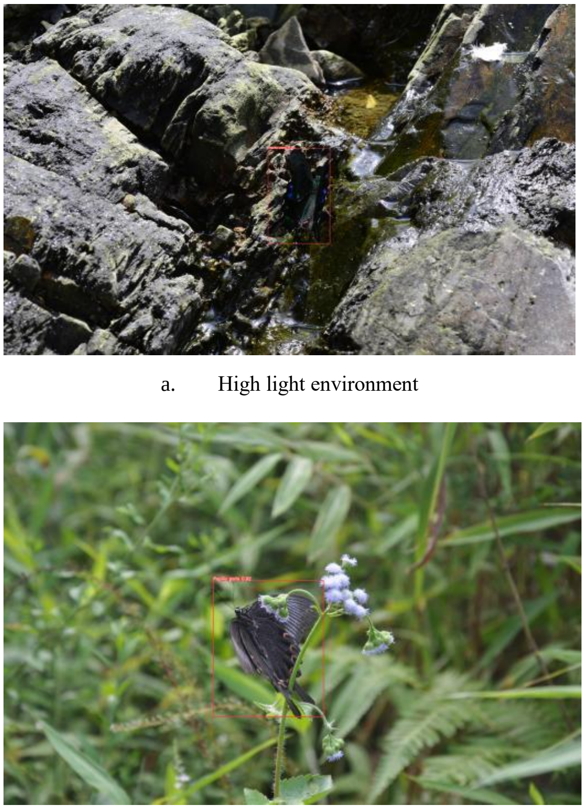


Figure 5. Loss, Precision, Recall, mPA0.5 and mAP0.5-0.95 curves

E. Comparison of test results

In order to more intuitively verify the inspection effect of YOLOv7 on Lepidoptera, in this paper, The respectively tested Lepidoptera in several groups of complex situations, including detection in bright light, detection in ordinary environment and detection in dark light situation, as well as detection in the case of smaller targets and farther distance.

As shown in Figure 6, the model trained by YOLOv7 has better test results no matter in complex situations such as strong light or dark light, and its targets almost always fall in the correct candidate box, with few false checks and fewer misses, and the detection confidence is also relatively high. It proves that the detection ability of the model as well as the feature extraction ability of the YOLOv7 model in such complex situations presents a good performance.



a. High light environment

b. High light environment

Figure 6. Lepidoptera Insect Detection Effect

V. Conclusions

In this paper, a deep learning algorithm is used to accomplish the task of detecting lepidopterous insects. In the preliminary experiments, it is considered that this kind of target detection is mainly used in complex and high real-time environment, so the YOLOv7 model with faster detection speed is used, and according to the analysis of the results of the comparative experiments, it can be seen that the algorithm model is indeed much faster than the other models in the detection speed, which can reach 33.08it/s, but the improvement of the accuracy is not obvious. Misdetection of Lepidoptera is easy to occur in some occasions where insects are gathered, but occasional misdetection in the actual agricultural production scenarios will not affect its overall use, and the speed can be used to detect insects several times, thus enhancing the correct detection of the target.

Based on the analysis of the experiments, its main problem lies in two aspects, lack of sufficient samples and background interference. For the Lepidoptera detection task, a limited number of Lepidoptera species and samples in the training dataset may result in the model performing poorly on unseen Lepidoptera classes or variants. The solution is to collect more Lepidoptera image data and ensure that the dataset contains diverse Lepidoptera species and postures. Secondly, Lepidoptera may appear in a variety of complex backgrounds, such as flowers and leaves. For target detection algorithms like YOLOv7, if the background is similar to the color and texture of lepidopteran insects, it may lead to false or missed detection. One improvement is to use semantic segmentation networks or other image processing techniques to extract regions of butterflies and use them as inputs for target detection to minimize background interference. For the above problems, it is advisable to continue to review the information and references later on to improve the settings of the network structure to enhance the performance of the detection of lepidopterous insects.

References

[1] YANG Hongwei, ZHANG Yun. Progress in the application of computer vision technology in insect identification [J]. Bioinformatics, 2005, (03):133-136.

[2] Yu M C, Yu T, Wang S C, et al. Big data small footprint: the design of a low-power classifier for detecting transportation modes [J]. Proceedings of the Vldb Endowment, 2014, 7(13):1429- 1440.

[3] Redmon J, Divvala S, Girshick R, et al. You only look once: unified, real-time object detection [C]. IEEE Conference on Computer Vision and Pattern Recognition, 2016: 779－788.

[4] Zhu Leqing, Zhang Daxing, Zhang Zhen. Image recognition of lepidopteran insects based on color names and OppcmmtSIFT features [J]. Journal of Entomology, 2015, 58(12).

[5] ZHANG Lei, CHEN Xiaolin, HOU Xinwen et al. Construction and testing of an automatic digital image recognition system for insects of the genus Drosophila in the family Psittacidae [J]. Journal of Entomology, 2011, 54(2):184-196.

[6] Redmon, 3oseph, and Ali Farhadi. YOL09000: Better; Faster, Stronger. Conference on Computer Vision and Pattern Recognition, 6517-6525, 2017.[4].

[7] Girshick R. Fast R-CNN [C]. Proceedings of ICCV 2015, 2015.

[8] Girshick R, Donahue J, Darrell T, et al. Rich feature hierarchies for accurate object detection and semantic segmentation[C]. Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, 2013:580-587.

[9] He, Kaiming et al. Deep Residual Learning for Image Recognition [C]. 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2016): 770-778.

[10] W. Zhang, S. Wang, S. Thachan, J. Chen and Y. Qian. Deconv R-CNN for Small Object Detection on Remote Sensing Images [J]. IGARSS 2018-2018 IEEE International Geoscience and Remote Sensing Symposium, 2018:2483-2486.

[11] K. S. Htet and M. M. Sein. Event Analysis for Vehicle Classification using Fast RCNN [C]. 2020 IEEE 9th Global Conference on Consumer Electronics (GCCE), 2020:403-404.

[12] Xing Shanshan, Zhao Wenlong. A review on UAV target detection in complex scenes based on YOLO series algorithms [J]. Computer Application Research, 2020, 37(S2):28-30.

[13] Szegedy C, Wei L, Jia Y, et al. Going deeper with convolutions [C]. 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). IEEE, 2015.

[14] Matthew D. Zeiler and Rob FergusD. Fleet et al. Visualizing and Understanding Convolutional Networks [C]. ECCV 2014, Part I, LNCS 8689, pp. 818‑833, 2014.

[15] Chen, Yaqian. Research on Quantization and Deployment of Convolutional Neural Networks Based on Nor Flash [D]. University of Science and Technology of China, 2021.

The Application of Whale Optimization Algorithm in Array Antennas

Long Qin
School of Information and Engineering
Xi'an Industrial and Commercial College
Xi'an, 710072, China
E-mail: 18092298157@163.com

Fan Yu
School of Information and Engineering
Xi'an Industrial and Commercial College
Xi'an, 710072, China
E-mail: yffshun@163.com

Sihang Yu
Imaging Marketing Department
Canon Medical Systems (China) Co., Ltd.
Xi'an, 710072, China
E-mail: yusihang@canon-medical.com.cn

***Abstract*—With the continuous improvement of various radio system performance indicators, the research work on antenna has become particularly important. According to different scenarios and requirements, practical projects also need the corresponding antennas to produce different radiation patterns. By reasonably setting the parameters of the array antenna, the target radiation pattern can be obtained to meet real life applications. When the array antenna has a large number of basic units and the expected far-field pattern is complicated, the design of the array antenna becomes a complicated optimization problem. To solve this problem, Whale Optimization Algorithm (WOA) is proposed. WOA is not only simple and fast, but can also get the global optimal solution. Therefore, WOA has developed rapidly in recent years. However, the application of this algorithm in the field of antenna design is still relatively rare, thus using WOA to solve the optimization problem of array antenna design is very valuable.**

Keywords-Multi-source Data Fusion; ICP Algorithm; IMU; Three-dimensional Reconstruction

I. INTRODUCTION

The field of antenna design is still relatively rare. There With the continuous improvement of various radio system performance indicators, the research work on antenna has become particularly important. According to different scenarios and requirements, practical projects also need the corresponding antennas to produce different radiation patterns. By reasonably setting the parameters of an array antenna, the target radiation pattern can be obtained to meet real life applications.

In the early stage, analytical methods and traditional engineering optimization methods were used to solve such problems. However, these solutions not only suffer from long calculation time and lack of precision, but were also unable to solve the synthesis of high dimensional complex demand radiation pattern.

With the emergence and rapid development of various intelligent optimization algorithms, array antenna researchers found intelligent optimization algorithms very suitable for solving such complex optimization problems, and started using various intelligent optimization algorithms to solve array antenna optimization problems. As a new intelligent optimization algorithm, WOA has not been widely studied. This paper will study the characteristics of WOA and apply it to the optimization problem of an array antenna. The algorithm mimics the three steps of hunting behavior: encircling prey, spiral bubble-net attacks and searching prey respectively. The three steps of WOA achieve the following functions: first,

DOI: 10.2478/ijanmc-2023-0074

"encircling prey" enables the whales to swim to the nearest location, which improves the search ability; second, "spiral bubble-net attacks" can improve the convergence speed and local search in a spiral way, which improves the search efficiency of whales; third, "searching prey " is the behavior that whales search for the prey randomly according to the position of each other so as to enhance the global search ability. Since WOA searches globally for optimal solutions, it is considered an effective global optimizer [8]. As a result, WOA has developed rapidly in recent years. However, the application of this algorithm in before, using WOA to solve the optimization problem of array antenna design can be very valuable.

II. BASIC PRINCIPLES OF WOA

A. *Encircling Prey*

Whales need to determine the target location first, and then surround and hunt. Whale optimization algorithm assumes the current optimal or near-optimal position as the target position. After the optimal candidate solution is established, the positions of other whale individuals are iteratively updated to gradually approach the optimal search for local search. The specific process can be shown in the following formulas (1) and (2):

$$\mathrm{D} = \left| C \cdot X^{*}(t) - X(t) \right| \tag{1}$$

$$X(t+1) = X^{*}(t) - A \cdot D \tag{2}$$

Update the positions of the other searches as shown in Equations (3) (4) (5) below:

$$A = 2\alpha \cdot r - \alpha \tag{3}$$

$$C = 2r \tag{4}$$

$$\alpha = 2\left(1 - \frac{t}{\max_t}\right) \tag{5}$$

The above equations are commonly used where $\max_t$ is the maximum number of iterations and α is an important variable of WOA. As shown in the equations, the change in convergence factor α will affect the value of the coefficient vector A and indirectly control the activity of the whale. The shrinking encircling mechanism is achieved through changing the value of convergence factor α $Fig.1$ is a schematic describing the mechanism of encircling prey.

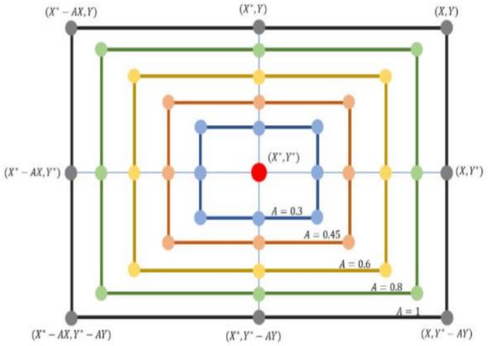


Figure 1. Schematic of encircling prey

B. *Spiral Bubble-net Attack*

The shrinking encirclement mechanism is that the whale's encirclement of food will gradually shrink in the process of hunting. Whales, while contracting and encircling the food, also swim spirally to the food, which is the spiral position update.

The implementation of the shrinking encircling mechanism mainly depends on changing the value of convergence factor α . When the value of convergence factor α is small, the value of the coefficient vector A will also become smaller, and the value of the coefficient vector A will affect the search ability of search agents. By increasing the value of A, the search range of search agents will grow to a larger range, so that the global search range of the group will be expanded. The global search ability will be enhanced and less likely to be trapped by a local optimum. When the value of A decreases, the search area of search agents will be smaller, hence increasing the local search ability of the group as well as the search speed. Over the course of the entire iteration, α decays linearly from 2 to 0, making A changes within the interval $[-\alpha, \ \alpha]$. When the value of A is set between $(-1,1)$, it means that the positions of search agents in the next iteration may be anywhere between the current position and the current optimal position. Therefore, search agents are still active in the shrinking bubble-net. When random vector r is between $\left[0,1\right]$, the entire process of the shrinking encircling mechanism can be represented by the model shown in $Fig.2$:

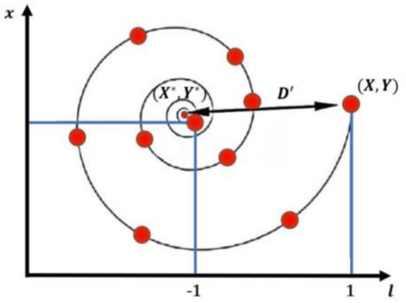


Figure 2. Spiral bubble-net attack of prey

The mechanism of spiral updating position can be represented by the following equation (6):

$$X\left(t+1\right)=D'\cdot e^{bl}\cdot\cos\left(2\pi l\right)+X\left(t\right) \qquad (6)$$

Where $D'=\left|X^*\left(t\right)-X\left(t\right)\right|$ and represents the distance between the optimal location and the current search agent, b is a constant used to define the shape of spiral movement and l is a random variable ranging between $\left[-1,1\right]$. When l reaches 1, it indicates that the current whale is the farthest away from the optimal position. When the value of l is -1, it indicates that the current agent is the closest to the optimal position. The symbol "·" is element-by-element multiplication and e represents the base of the natural logarithm.

As can be seen from the figure above, the distance between the current agent and the optimal target position needs to be calculated before the spiral position is updated. The figure above is the mathematical model of motion obtained by simulating the spiral updating position mechanism of humpback whales.

When humpback whales shrink and encircle prey, they also update their spiral positions. It is necessary to assume that each behavior has a certain probability if we were to use a mathematical model to describe these simultaneous behaviors.

The mathematical model is as follows (7):

$$X\left(t+1\right)=f\left(x\right)=\begin{cases}X\left(t\right)-A\cdot D & P<0.5\\ D'\cdot e^{bl}\cos\left(2\pi l\right)+X\left(t\right) & P\geq 0.5\end{cases} \qquad (7)$$

C. *Searching Prey (global exploration)*

In the actual hunting process of humpback whales, the current searched fish school may not be the optimal fish school in the hunting space. Therefore, humpback whales will also change their positions according to the positions of other whales. As shown in $Fig.3$, global random search is performed for the best fish school in the space. This random search mechanism is simulated by random variables A. In the algorithm, when $0<|A|<1$, the whale launches an attack on the prey. When $|A|>1$, the whales will carry out a global random search for prey, where each humpback whale updates its position according to a search agent randomly selected in the global space. This mechanism increases the population diversity of the algorithm and significantly improves the global search ability of WOA. In the equations below, D is the distance between the position of the current whale and the position of any random search agent, $X_{rand}\left(t\right)$ is the position of a random agent in the population. The specific mathematical model is shown in Equations (8) and (9) below:

$$D=\left|C\cdot X_{rand}\left(t\right)-X\left(t\right)\right| \qquad (8)$$

$$X\left(t+1\right)=X_{rand}\left(t\right)-A\cdot D \qquad (9)$$

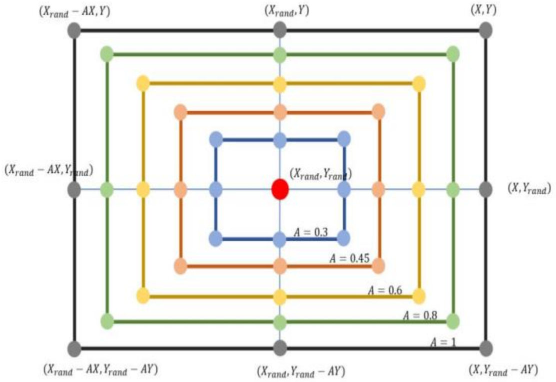


Figure 3. Schematic of searching prey (global exploration)

III. Improving WOA

In the aspect of global search, the WOA algorithm performs global search through the current point and the current optimal point.

A. *Logistic Mapping*

At present, there are usually many random variables in algorithms based on crowd behavior research. These random variables are generally adjusted through probability. However, these parameters obtained by probability are too random, which are likely to slow the convergence speed of the algorithm and affect the accuracy of the solution. Many researchers now use logistic mapping instead of random probability to solve this problem. Logistic mapping is a typical model for studying the behavior of complex systems such as dynamic systems with discrete time, chaotic and fractal dimensions [4]. It is a nonlinear iterative equation described as follows:

$$x_{n+1} = \mu x_n \left(1 - x_n\right),\ \ \mu \in \left(0,4\right),\ \ x_n \in \left[0,1\right] \quad (10)$$

In this equation, n represents the number of iterations, x_n represents the insect-population in the n_{th} generation and $\left(1-x_n\right)$ represents the influence of environmental factors. μ is a system bifurcation control parameter closely related to the dynamic characteristics of chaotic logistic mapping system. Different values of μ will have different effects on the system. When $\mu < 1$, this suggests the insect-population decreases and, when $\mu > 1$, it means the insect-population increases. The impact of μ on the distribution of logistic mapping is described in $Fig.4$:

where,

a) When $0 < \mu \le 1$, no matter the initial value of the system and the number of iterations, the final system trajectory will converge to 0;

b) When $1 < \mu \le 3$, there will be two steady-state solutions: 0 and $1 - \dfrac{1}{\mu}$, and after a number of iterations, the result will converge to either one of them;

c) When $3 < \mu \le 4$, the system will start to exhibit some periodic trajectories;

d) When $3.569945972 \le \mu \le 4$, the system is in a chaotic state, where, the result generated by iterations has pseudo-randomness, as well as a strong sensitivity towards the state of the initial value;

e) When $\mu = 4$, the distribution of x becomes uneven. A U-shaped relationship is observed with highest frequency at the two extremes.

$Fig.4$ shows the distribution of logistic mapping with different values of μ. It can be seen that as μ is closer to 4, the system becomes more evenly distributed. While when $\mu = 4$, the distribution of x is more frequent at the two extremes and less frequent in the middle. This suggests the closer the value of μ is to 4, the better the outcome. Therefore, in this paper, we establish the logistic mapping using $\mu = 3.99$. $Fig.5$ is the logistic mapping based on $\mu = 3.99$, where T is the number of iterations, the $Y-axis$ represents the value of x.

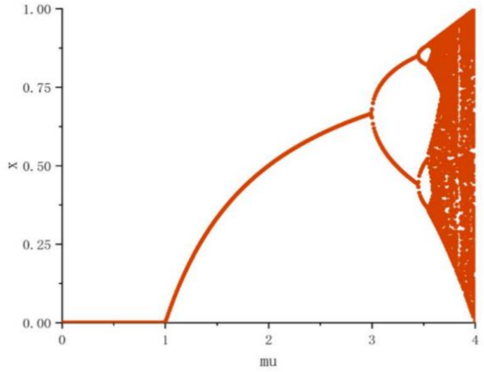


Figure 4. The distribution of logistic mapping with different values of μ

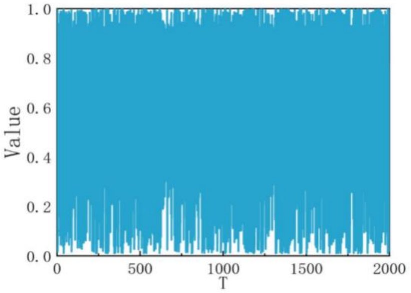


Figure 5. The distribution of logistic mapping when μ = 3.99

B. Inertia Weight

Inertia weight is a concept first appeared in Particle Swarm Optimization (PSO), where the changes of particle coordinates are related to the inertia weight in the iterations of PSO. When the value of inertia weight is large, the step size for the search becomes relatively large, which improves the global search ability of the algorithm. When the value of inertia weight is small, the local search ability of PSO will be better, and the accuracy of the optimal solution will also improve, but the search may be trapped by a local optimum. In this section, we introduce inertia weight into WOA, and applies inertia weight to the two steps: encircling prey and spiral bubble-net attack. Weight is added to the global optimal candidate solution, and the next group of whales will search according to the historical optimal information with added weight [5][8]. This process is updated as equations (11) and (12).

$$x(\mathrm{t}+1)=\omega(t)\cdot x_*(t)-A\cdot D \qquad (11)$$

$$x(t+1)=D'\cdot e^{bl}\cdot\cos(2\pi l)+x(t)\cdot\omega(t) \qquad (12)$$

Where $\omega(t)$ is the inertia weight, which adjusts the step size of the search. According to its characteristics, this paper chooses equation (13) as the iterative update of the inertia weight, where $\mu = t/\max\_t$, $\omega_{\max}=0.9$, and $\omega_{\min}=0.4$.

$$\omega(t)=\omega_{\max}-(\omega_{\max}-\omega_{\min})\cdot 2\mu\cdot sqrt(1-\mu^2) \qquad (13)$$

Fig.6 Shows the relationship between the inertia weight and the number of iterations $(t\le 1000)$. Because of the fast convergence speed of WOA, it is easy to fall into the local optimum, and it is basically stable at 700 iterations. At this time, the inertia weight increases rapidly, and the step size is expanded again to carry out the global search, and finally the optimal solution is determined. Hence, by introducing the inertia weight, we can better balance the global search and local search ability of WOA.

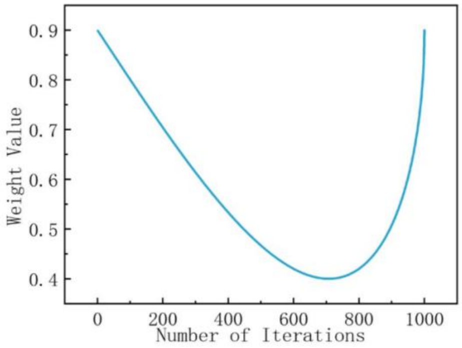


Figure 6. The relationship between the inertia weight and the no. of iterations

IV. CHEBYSHEV PATTERN SYNTHESIS

If the array antenna has a high sidelobe, the strong scattering points at the sidelobe will produce strong reflection of energy. This may cause the radar to mistakenly believe that there is a target in the main lobe direction of the antenna and may miss the target in the main lobe, making the radar to fail to work properly. Therefore, low sidelobe array antennas can not only help radars to perform normal target detection function, but can also improve the battlefield survivability of radars. According to the fundamental theory of antenna array, applying appropriate excitation amplitude on all the basic antenna units will help us obtain lower sidelobe levels. Chebyshev pattern synthesis and Taylor synthesis are methods commonly used in the synthesis of low sidelobe array antenna patterns [1][2][6]. Antenna pattern synthesis is the inverse process of pattern analysis. Pattern synthesis is to calculate the number, position, excitation current amplitude and phase of antenna array elements according to the given pattern conditions (sidelobe level, beam width, pattern shape, etc.). In this section, we will introduce the analytical method used to solve the optimal pattern synthesis of linear array antennas, namely, the Chebyshev pattern synthesis technique. This method solves the contradiction between low side lobe and narrow main lobe of an array antenna. The definition of Chebyshev polynomials (14) is as follows:

$$T_n(x)=\begin{cases}(-1)^n ch(n\ ch|x|) & x<\text{-}1\\ \cos(\text{n}\arccos x) & \text{-}1<x<1\\ \text{ch}(\text{n}\arccos x) & x>1\end{cases} \quad (14)$$

Let $x=\cos\ \delta$, then $T_n(\cos\ \delta)=\cos(n\delta)$. By using the trigonometric functions, $\cos(n\delta)$ can be expanded into the power polynomial of $\cos(\delta)$. Then substitute $x=\cos\ \delta$ into the equation, we can prove that $\cos(n\arccos x)$ is the power polynomial of x. Chebyshev's recurrence equation is then given by equation (15):

$$T_{n+1}(x)=2xT_n(x)-T_{n-1}(x) \quad (15)$$

Chebyshev polynomials have the following three properties:

i. Even-order polynomial has the characteristic of an even function, the polynomial curve is symmetric about the vertical axis, and that is, when n is even， $T_n(-x)=T_n(x)$. While odd-order polynomial has the characteristic of an odd function, that is, when n is odd, $T_n(-x)=-T_n(x)$.

ii. All the polynomials above pass through point $(1,1)$. When $-1\le x\le 1$, the value of all polynomials oscillates between -1 and 1, and the absolute value of polynomials are less than or equal to 1.

iii. All the zeros of the above polynomials are located at the interval $-1\le x\le 1$, and the values of the polynomials at the interval $|x|\le 1$ either monotonically increase or monotonically decrease.

Since all the characteristics of Chebyshev polynomials are consistent with the characteristics required in sidelobe patterns, the array factor can be expressed in the form of Chebyshev polynomials.

Given that the Chebyshev polynomials has only side lobes in the interval $[-1,1]$ and the main lobe is outside this interval, we need the variation range of the matrix factor outside the interval $[-1,1]$, which is given by equation (16) :

$$x=x_0\cos\left(\frac{\phi}{2}\right) \quad (16)$$

In the above equation, let $f_a(\phi)=T_{n-1}\left(x_0\cos\frac{\phi}{2}\right)$, then the following equations (17) and (18) can be obtained at the angle of maximum radiation $\theta_0=90^0$:

$$\phi=\beta d\cos 90^0=0,\quad x=x_0\cos\left(\frac{\phi}{2}\right)=x_0 \quad (17)$$

$$f_a(\phi=0)=T_{n-1}(x_0)=R \quad (18)$$

Where R is the true value of the desired sidelobe level SLL, as shown in equation (19). Therefore, according to equation (18) and the value of R, x_0 can be calculated.

$$R=10^{-SLL/20} \quad (19)$$

The Chebyshev polynomials and the matrix factor have the following correspondence, that is, when x_0 changes to $x_0\cos\left(\frac{\beta d}{2}\right)$, the true value of the matrix factor is the value of Chebyshev polynomial. The independent variable of Chebyshev polynomial is described as follows:

$$f_a(\phi)=T_{N-1}\left(x_0\cos\frac{\phi}{2}\right) \quad (20)$$

It can be seen that the independent variable of the matrix factor in the Chebyshev polynomial varies in the range $x_0\to x_0\cos\left(\frac{\beta d}{2}\right)$, and its range depends on spacing d and x.

With the given sidelobe level and the number of units, we can use the Chebyshev pattern synthesis method to calculate the excitation current corresponding to the optimal pattern. The comprehensive steps of deriving the excitation current are as follows: step one, get the value of R based on the value obtained by the above Equation (18) and the known sidelobe level, and then calculate x_0 . In order to simplify the calculation, the above equation can be converted to equation (21):

$$x_0 = \frac{1}{2}[(R+\sqrt{R^2-1})^{\frac{1}{N-1}}+(R-\sqrt{R^2-1})^{\frac{1}{N-1}}] \quad (21)$$

Step two: derive the excitation current of each basic unit. If the matrix factor is equal to the Chebyshev polynomial of order $N-1$, then the coefficients $\cos\frac{\phi}{2}$ of the same power on both sides of the equation should also be equal, and the excitation current of each basic unit can be calculated.

Step three: calculate the radiation pattern of the array antenna.

V. Verification of WOA and Chebyshev Pattern Synthesis

The optimal radiation pattern can be obtained by Chebyshev pattern synthesis method. For a given sidelobe level, Chebyshev synthesis method can achieve the narrowest zero-lobe width and main lobe width. For a given zero-lobe width, the Chebyshev synthesis method can obtain the lowest sidelobe level. In order to verify the feasibility of WOA in solving array optimization problems, this section attempts to use the algorithm to solve the optimal radiation pattern of a given array antenna, and compares the optimization results with Chebyshev pattern synthesis method, so that the effectiveness and accuracy of WOA can be verified.

Combining the characteristics of the Dolph-Chebyshev radiation pattern, the following two aspects should be considered when designing the objective function: one is that the main lobe beamwidth should be close to the expected beamwidth, and the other is that the side lobe level should meet the design requirements. We will use the two-mask function to build the target radiation pattern, and the optimal radiation pattern should be between the upper function $Mask_U$ and the lower function $Mask_L$, as shown in $Fig.7$. In the main lobe region, the main lobe width of the actual pattern and the target pattern should be equal. In the sidelobe region, the optimal sidelobe level should be equal to the designed sidelobe level. Therefore, the fitness function in this section can be expressed as equation (22):

$$fitness = \sum_{i=0}^{N} \begin{cases} |f(\theta)-Mask_U(\theta)|, & f(\theta) > Mask_U(\theta) \\ 0, & Mask_U(\theta) > f(\theta) > Mask_L(\theta) \\ |f(\theta)-Mask_L(\theta)|, & f(\theta) < Mask_L(\theta) \end{cases} \quad (22)$$

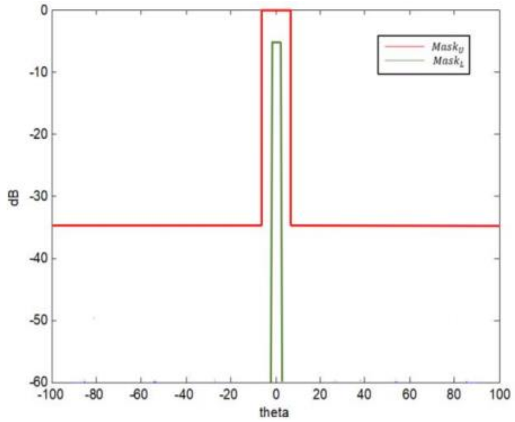


Figure 7. Schematic of two-masks function

Where θ is the radiation angle, $f(\theta)$ is the actual radiation pattern, $Mask_U(\theta)$ is the upper bound of the expected radiation pattern, and $Mask_L(\theta)$ is the lower bound of the objective function. Fitness function is used to represent the error function between the actual pattern and the expected pattern. The smaller the error function is, the closer the actual pattern is to the expected pattern.

In this section, a uniformly arranged linear array of 30 elements is selected as an example. The elements are ideal point source, the spacing between each element is $d=\lambda/2$, and the expected sidelobe level is -35dB. Each element is excited in phase, and the optimization variable is the excitation amplitude I_n of each basic unit. The value of the excitation amplitude is located at the interval $[0,1]$. Since the amplitudes of the 30 array elements are centrally symmetric, the dimension of the optimization variable is half of the array size, which is 15. In this example, the step interval of the radiation pattern is 0.1 degrees, and the angle range of the radiation pattern is $\left[-90^0, 90^0\right]$. According to the equation of uniform linear array, the array radiation pattern $f(\theta)$ corresponding to any optimized variable can be calculated. The selection of the objective evaluation function of the desired radiation pattern is shown in $Fig.4$. Where $Mask_U(\theta)$ and $Mask_L(\theta)$ are the upper and lower bounds of the desired radiation pattern. In this example, the parameters of WOA are set as follows: the number of populations $N = 200$, and the maximum number of iterations is 300.

Therefore, Chebyshev amplitude distribution can be obtained and the radiation pattern of array factors can be calculated. $Fig.8$ shows the comparison of low sidelobe pattern obtained by Chebyshev synthesis and WOA algorithm, and $Fig.9$ compares the amplitudes obtained by Chebyshev pattern synthesis and WOA. As can be seen from the comparison results in $Fig.8$, WOA can obtain a better radiation pattern than Chebyshev synthesis, with the same lobe width and the same amplitude and position of sidelobe levels. $Fig.9$ Compares the excitation amplitudes and it shows that the distribution and trend, as well as the excitation amplitudes obtained by WOA and Chebyshev synthesis are basically the same. The specific differences between the two are shown in Table 1Conclusion.

This paper focuses on the application of WOA in the optimizing the radiation pattern of array antennas. Firstly, WOA is used to solve the amplitude distribution for optimal radiation pattern of uniform linear array with Chebyshev distribution. Then the optimization results of WOA are compared and analyzed with the amplitude distribution obtained by Chebyshev synthesis method, which verifies the effectiveness and accuracy of WOA to solve the optimization problem of array antennas.

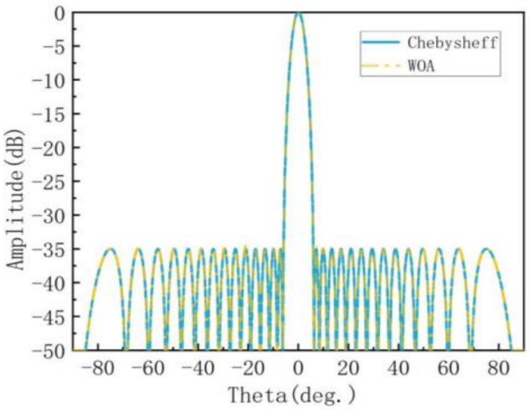


Figure 8. Comparison of low sidelobe radiation pattern obtained by Chebyshev synthesis and WOA

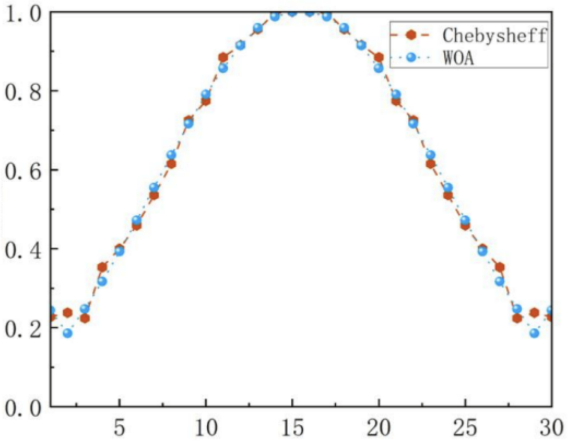
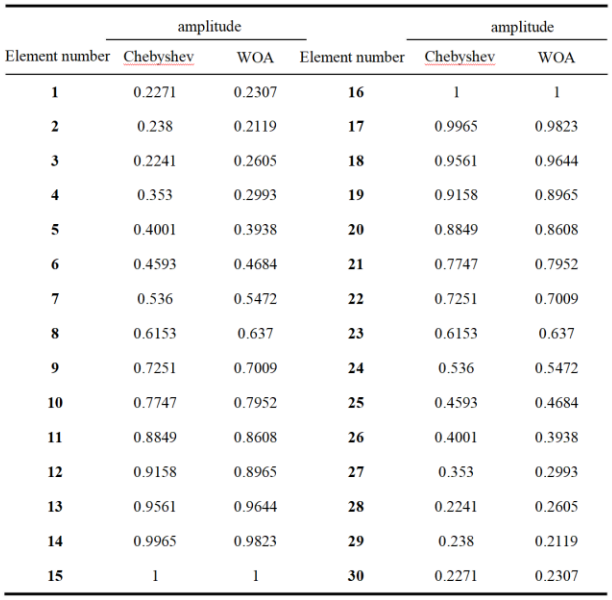


Figure 9. Comparison of distribution of excitation amplitudes obtained by Chebyshev synthesis and WOA

TABLE I. COMPARISON OF OPTIMIZATION RESULTS BETWEEN CHEBYSHEV DISTRIBUTION AND WOA



	amplitude			amplitude	
Element number	Chebyshev	WOA	Element number	Chebyshev	WOA
1	0.2271	0.2307	**16**	1	1
2	0.238	0.2119	**17**	0.9965	0.9823
3	0.2241	0.2605	**18**	0.9561	0.9644
4	0.353	0.2993	**19**	0.9158	0.8965
5	0.4001	0.3938	**20**	0.8849	0.8608
6	0.4593	0.4684	**21**	0.7747	0.7952
7	0.536	0.5472	**22**	0.7251	0.7009
8	0.6153	0.637	**23**	0.6153	0.637
9	0.7251	0.7009	**24**	0.536	0.5472
10	0.7747	0.7952	**25**	0.4593	0.4684
11	0.8849	0.8608	**26**	0.4001	0.3938
12	0.9158	0.8965	**27**	0.353	0.2993
13	0.9561	0.9644	**28**	0.2241	0.2605
14	0.9965	0.9823	**29**	0.238	0.2119
15	1	1	**30**	0.2271	0.2307

REFERENCES

[1] Chen Shuiqing, and Huang Lihui. Smart antenna and its applications in wireless communication [J]. China New Telecommunications, 2020, 22(5):1. (In Chinese).

[2] Chen Biran, and Wang Ye. Digital array antenna wideband radiation pattern synthesis technology [J]. Ship board Electronic Countermeasure, 2019, 42(4):5. (In Chinese).

[3] Wang Wei, Wang Qinzhao, Liu Gangfeng, Cheng Hui, Tao Yi, and Guo Aobing. Countering unmanned ground system: A review of key technologies [J]. Acta Aeronautica et Astronautica Sinica, 2022, 43(7). (In Chinese).

[4] Jing Yang, Fan Xuhui, and Liang Junli. Comprehensive design of array antenna radiation pattern without template [J]. Aeronautical Science & Technology, 2019, 30(6):7. (In Chinese).

[5] Bi Xiaokun, Zhang Xiao, Wong Saiwai, et al. Synthesis Design of Chebyshev Wideband Band-pass Filters with Independently Reconfigurable Lower Passband Edge [J]. IEEE Transactions on Circuits and Systems II: Express Briefs, 2020, 67(12).

[6] Li Yang, Lei Zhu, Wai-Wa Choi, et al. Wideband Balanced-to-Unbalanced Bandpass Filters Synthetically Designed With Chebyshev Filtering Response [J]. IEEE Transactions on Microwave Theory and Techniques, 2018, 66(10).

[7] Wei Li, Gai-Ge Wang, Amir H. Gandomi. A Survey of Learning-Based Intelligent Optimization Algorithms [J]. Archives of Computational Methods in Engineering, 2021, 28(5).

[8] Kumari*, K. Karuna, and Dr. P. Sridevi. Sidelobe Level Optimization of Rectangular Microstrip Patch Antenna Array Using Binary Coded Genetic Algorithm [J]. International Journal of Innovative Technology and Exploring Engineering, 2020, 9(4)

Research on Multi-Person Pose Estimation Technology

Hongyan Wang*
School of Computer Science and Engineering
Xi'an Technological University
Xi'an, China
E-mail: whyanon@163.com
*corresponding author

Changyuan Wang
School of Computer Science and Engineering
Xi'an Technological University
Xi'an, China
E-mail: Cyw901@163.com

***Abstract*—Human pose estimation is a hot topic of computer research in recent years, which promotes the progress of society and brings many conveniences to people's lives. Fom traditional methods to the mainstream deep learning-based methods, the primary approach in deep learning involves the use of convolutional neural networks to reduce computational complexity and improve network accuracy, but because the network structure is too deep to improve the accuracy, the trained model parameters are also very large, and it is very dependent on the input of hardware equipment. At this time, the lightweight human pose estimation can solve this problem very well. This paper mainly describes the knowledge of convolutional neural network in detail and compares it with traditional image algorithms. The OpenPose model is a classic model based on convolutional neural network that can well achieve single-person and multi-person human pose estimation model, but because the convolution kernel in its network structure is too large to increase the amount of calculation, this paper proposes three improvements to the network structure of the conventional OpenPose model. Finally, the precision of the model is improved by about 40%, which verifies the feasibility of lightweight human body posture estimation research.**

Keywords-Convolutional Neural Network; Network Structure; OpenPose Model; Multi-person Human Pose Estimation

I. INTRODUCTION

In recent years, computer vision related technologies have developed rapidly. Human posture estimation [1] involves the most basic method of human posture research, and has become a hot spot for many scholars. Its primary objective is to identify the spatial coordinates of human joints and key body parts [2][3] within an image, enabling the extraction of partial or comprehensive limb information. This information is crucial for assessing and interpreting human body movements and behaviors. Due to many factors such as shooting angle, lighting, and environment, it is difficult for traditional human pose estimation algorithms to achieve satisfactory results. Given the extensive exploration of deep learning, particularly convolutional neural networks the utilization of such networks in human pose estimation has progressed swiftly, the traditional manual extraction of features is replaced by the method of learning features by convolutional neural networks, thereby realizing end-to-end optimization [4], although the current method based on convolutional neural networks has emerged as the mainstream method for human pose estimation, but there are still some core problems that have not been solved. For example, the existing research work mainly focuses on enhancing the accuracy of human pose estimation methods, resulting in progressively intricate network models. However, this emphasis on accuracy sometimes neglects the crucial trade-off between precision and processing speed in the network.

II. RELATED WORKS

A. Survey of Pose Estimation

There are two ways to estimate two-dimensional pose, one is to measure all the heads, left and right hands, knees, etc. from the bottom up [5][6], and then connect all the joints with the human body and then combine them together. The second is from the top down, transforming the pose estimation of multiple people into the pose estimation of multiple individuals, typical of CPM, Hourglass, CPN, Simple Baselines, HRNet,

DOI: 10.2478/ijanmc-2023-0075

MSPN, etc [7]. OpenPose is currently the most popular method of bottom-up multiplayer pose assessment [8]. First, the OpenPose network places multi-person photos on the previous level for feature extraction, and then inputs the features to two parallel branches, and first obtains a set of credibility maps from one branch, each confidence map represents the key points of the human body bone composition [7][8]. The second branch is used to predict the importance of other parts. All that remains is to refine the predictions of each branch, make a bipartite map of the credibility of each part, and then use the PAF value to stitch together the weaker parts of the dichotomy to obtain a rough outline of the human body and piece it together into a person.

With the advancement of computer equipment and human pose estimation technology, the utilization of conventional convolutional neural network algorithms in deep learning has been caused by complex convolutional convolution and large computational capacity to meet the requirements of the people and market demand under the current social form [9].

B. The OpenPose Mode

Cao et al. [10] introduced the OpenPose model, employing a technique known as the "local affinity field". This method effectively encode the position and orientation of limbs so that the connections between key points can be appropriately combined. The motion classification model uses neural networks to deeply fuse and classify human body key points output by the human posture estimation model, thereby achieving real-time monitoring of human motion.

C. Currently

At present, how to maintain the high performance of the model and make it achieve lighter mass has become a hot spot in current research. The OpenPose model extracts the feature map in the VGG network [11], a convolutional neural network is employed to analyze the reliability of the key points and the affinity vector field associated with these key points, and uses the Hungarian algorithm to match and optimize the key points [12][13]. Although the network complexity is reduced, the performance improvement is not significant, and the amount of operation is also increased, so in order to reduce the network complexity, reduce the network parameters, reduce the amount of calculation, on the basis of OpenPose, a lightweight model light OpenPose is proposed to realize the lightweight human pose algorithm [9][13], so that the human pose estimation in practical applications on the basis of liberating manpower to achieve better results. The components of lightweight human pose estimation are illustrated in Figure 1.

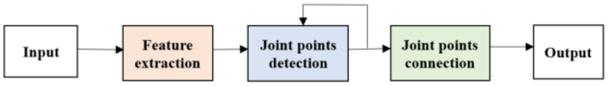


Figure 1. Components of lightweight human pose estimation

III. METHODS AND MATERIAL

A. Human Skeletal Coding

The distinctive attributes of different body parts can be condensed into either 18 or 25 key feature points. The human skeleton, formed by these feature points, effectively portrays the body's posture. The accuracy of human movement can be assessed based on the angles between the joints of the human body, and will not be affected by factors such as human body type, skin color, and clothing. This article is to use this feature to identify motion.

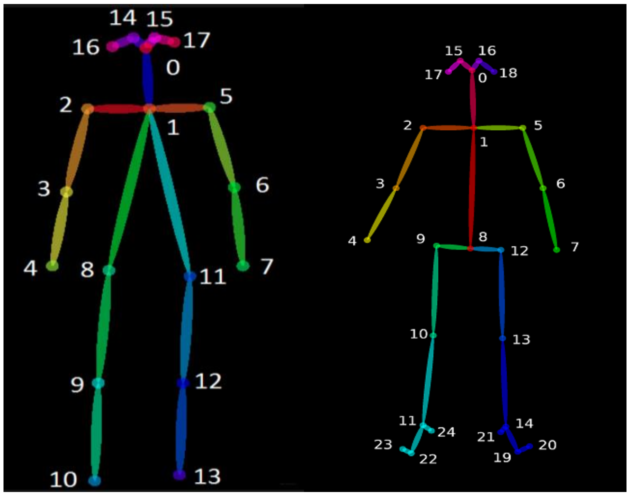


Figure 2. COCO (left) and OpenPose (right) coding diagrams

Deep learning serves as the foundation for conducting numerous descriptions of human body posture information. Currently, within the realm of human pose estimation, the labeling of data is also the marking of bone as the key point. The coding method in this article is mainly shown in Figure 2.

B. *Based on OpenPose Network Structure Design*

1) *PCM and PAF*

The crucial points of the human body can be expressed by a heat map, which is simulated using a Gaussian model, on the basis of which the values of each point represent the probability of a key point in the data of that point.

Part confidence map (PCM): This method is used to represent the Gaussian response asscociated with a pixel on the joint point. When the pixel is far away from the joint, the value of the response will increase.

Joint affinity field (part affinity fields, PAF): used to describe the spatial constraint connection between key points [14], that is, the alignment of the skeleton position and the orientation of pixels on the skeleton are crucial factors. The proximity of the predicted Part Affinity Field (PAF) to the actual PAF determines the closeness of the connection between the two nodes. Expressed by PCM there are C class vector fields, and the vector field of each limb is expressed by two feature maps, which represent the direction vectors of x and y, for a total of 19 classes, so the output of the convolutional network is 38 feature maps.

$$L^{*} = (L_1^{*}, L_2^{*}, ..., L_c^{*}), L_c^{*} \in R^{w \times h \times 2}, c \in \{1, ...c\} \quad (1)$$

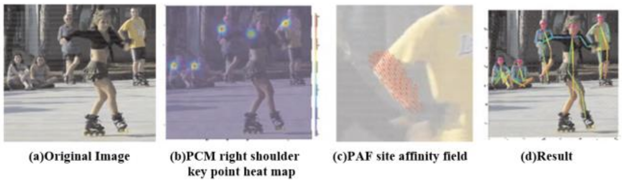

(a)Original Image
(b)PCM right shoulder key point heat map
(c)PAF site affinity field
(d)Result

Figure 3. The process of image inspection

Basic process: As shown in Figure 3, first enter a picture (a), go through the network, get a bunch of heat maps (b) and PAF sets (c), and obtain the parsed diagram (d) after matching the dichotomous diagram.

2) *OpenPose network structure design*

OpenPose is a convolutional neural network-based model that undergoes enhancements for real-time, multi-person human keypoint detection within a supervised learning framework. The primary architecture of the original network is depicted in Figure 4, which is divided into two components [15]. Initially, feature extraction is accomplished through the traditional convolutional neural network VGG19 (the first 10 layers), resulting in the acquisition of feature map F. This feature map is then fed into a two-branch multi-stage network. The upper branch focuses on predicting Partial Affinity (PAF), capturing positional and directional information between key points. Simultaneously, the lower branch is dedicated to predicting a Partial Confidence Map (PCM), which characterizes the location of key points.

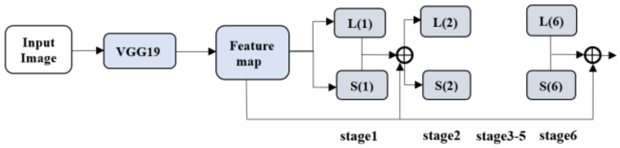


Figure 4. OpenPose's network structure diagram

The internal structure of a network for predicting partial affinity and confidence is illustrated in Figure 5. This network employs multiple stages to extract semantic information between key points.

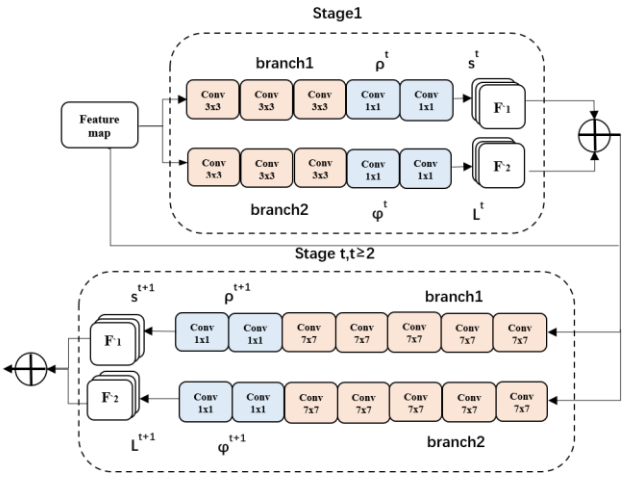


Figure 5. Diagram of the forecast internal structure

In addition to the first stage, multiple 7x7 convolution kernels are used in all other stages, which can obtain larger receptive fields [15], towards the conclusion of each stage, the forecasted values of the two subnetworks are connected with the initial characteristic curve F and used as input for the next step, In order to enhance the fusion of deep feature information without overlooking surface features, the equation description is as follows (2)(3):

$$S^t = \rho^t(F, S^{t-1}, L^{t-1}), t \geq 2 \qquad (2)$$

$$L^t = \Phi^t(F, S^{t-1}, L^{t-1}), t \geq 2 \qquad (3)$$

C. Optimization

1) Imporve feature extraction

TABLE I. LIGHTWEIGHT BACKBONE PERFORMANCE COMPARISON

	AP, %
MobileNet v1 (cut to conv4_1)	37.9
Dilated MobileNet v1 (cut to conv5_5)	42.8
Dilated MobileNet v1 (cut to conv5_6)	43.2
Dilated MobileNet v2 (cut to conv6_3)	39.6

Lightweight OpenPose replaces VGG as the backbone by using MobileNet_v1[16]. However, the MobileNet network structure lacks depth, as when returning to a skeletal point, attention must be given not only to the immediate vicinity but also to a broader context. This approach allows for accurate localization of the skeletal point even in the presence of occlusion, and if you simply use MobileNet, the effect is not good. Insufficient depth in the network structure hampers the attainment of a broader receptive field, consequently impacting the performance of the receptive field, so that the accuracy of bone positioning will be reduced, according to the test of convolution performance in 2D multi-person human pose estimation, as shown in Table Ⅰ, where AP represents the average accuracy and GFLOPs represent the model complexity.

Therefore, in order to improve the receptive field to get better results, you can use MobileNet (Dilated MobileNet v1) with void convolution, and the main function of hollow convolution is to expand the receptive field and obtain contextual information.

2) Improve the number of branches

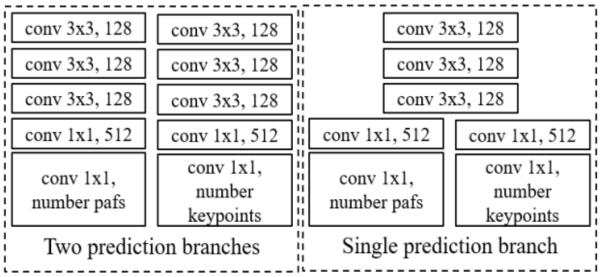


Figure 6. Merge branching diagram

The OpenPose model has two parallel branches, as shown in Figure 3.4 (left), utilized for forecasting the keypoint heatmap (keypoint map) and the keypoint affinity field (PAF). Because the two branches are the same in structure, but the number of output results is different, so lightweight OpenPose considers merging the two branches into one branch. Figure 6 (right) directly combines the original two prediction stages into one stage, and only needs to use 1*1 convolution in the last output stage to separate the two stages as output.

3) Modify the convolution kernel size

Figure 4 shows the internal network structure of OpenPose, where two prediction branches are composed of multiple convolution kernels concatenated, and 7×7 convolution kernels are frequently used in the prediction network stage. Although larger convolutional kernels can obtain larger receptive fields, their computation is also large.

Therefore, on this basis, lightweight OpenPose uses 1*1, 2*3*3 size convolution cascade, opting for a small convolution kernel instead o f a larger one significantly decreases the computational workload, in order to obtain the same feeling field with 7*7 convolution kernel, add a hole convolution with an expansion parameter size of 2 to the last piece of 3*3 convolution [15][16], because the kernel of the hole convolution is not continuous, therefore, the residual connection structure is used for each piece, as shown in Figure 7.

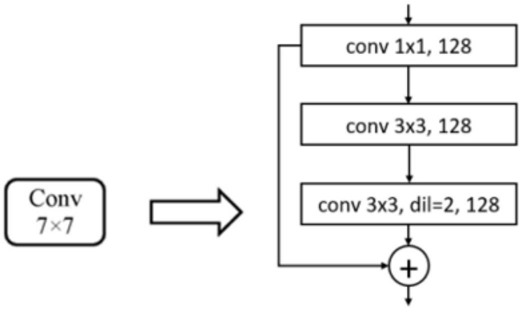


Figure 7. Improved convolution kernel structure

IV. RESULTS AND DISCUSSION

A. Model performance evaluation

Table Ⅱ shows the experimental results of the original OpenPose model and the improved OpenPose model under the COCO dataset, using AP value as the model evaluation index.

TABLE II. IMPROVING OPENPOSE NETWORK EVALUATION RESULTS

Model	AP, %
OpenPose	48.6
Optimization	86.3

B. Testing on COCO Datasets

1) Analysis of human bone point detection results

Since there are 3 people in the detected picture, a total of 3 people's key points are generated, and the OpenPose model has a total of 25 key points, of which 24 points are marked for Body; The improved OpenPose model in this paper uses a COCO dataset with a total of 19 points and 18 joint points, and the last point of which is labeled with the image background.

Figure 8 and Figure 9 show the confidence distribution line plot based on OpenPose model and lightweight OpenPose model to realize the key points of multi-person human posture, respectively, from which it can be seen that the distribution trend of confidence between the improved model and the original model has not changed greatly.

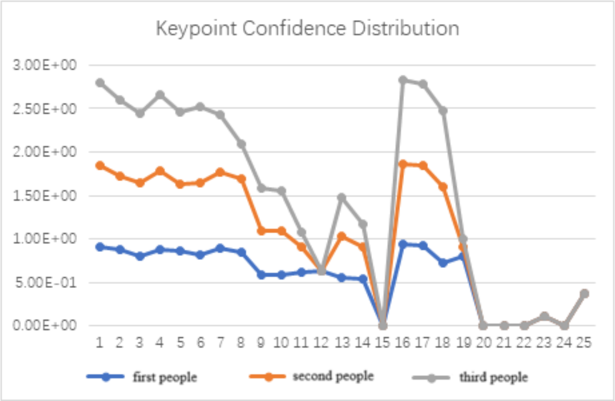


Figure 8. OpenPose model key point confidence distribution chart

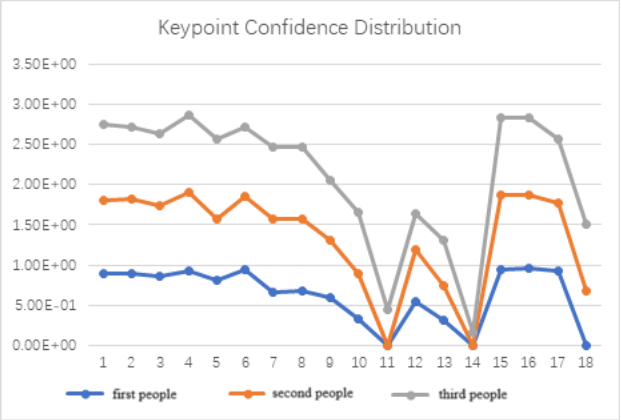


Figure 9. Lightweight OpenPose model key point confidence distribution chart

Through the confidence analysis of the key point position of the human body in Table III, the accuracy of detecting the position of the human body from OpenPose to lightweight OpenPose has no particularly large impact, and even improved, as shown in Table III.

TABLE III. COMPARISON OF AVERAGE CONFIDENCE

Average confidence of person i	1	2	3
OpenPose method	0.572	0.473	0.473
Lightweight OpenPose method	0.625	0.718	0.779

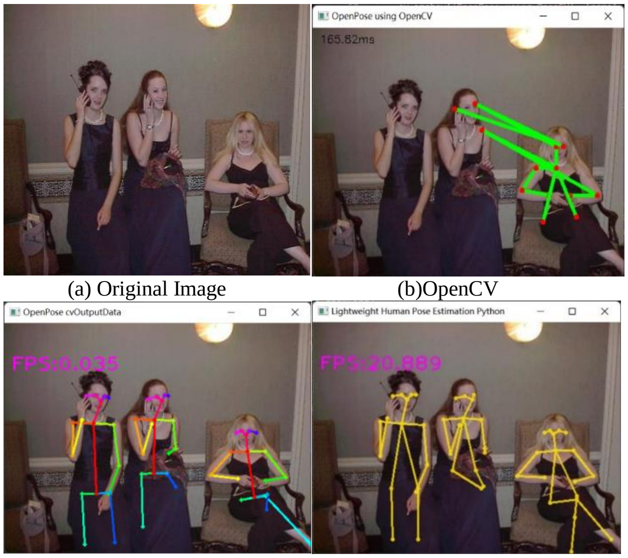
2) Analysis of attitude detection rate results

The comparison of OpenPose model and improved Openpo in terms of frame number, as shown in Table Ⅳ, can assess the accelerated speed of the model proposed in this study compared to the original model, indicating the feasibility of the improved OpenPose model for estimating and recognizing human posture.

TABLE IV. COMPARISON BEFORE AND AFTER NETWORK STRUCTURE IMPROVEMENT

Type	FPS
OpenPose	0.035
Lightweight OpenPose	20.889

3) *Multi-Person Pose Estimation Results*



(a) Original Image (b)OpenCV

(c) OpenPose (d) Lightweight OpenPose

Figure 10. Multi-person pose recognition results

As shown in Figure 10, analysis of multi-person pose recognition results, (a) is the original figure, (b) is a multi-person human pose recognition image not implemented by OpenPose, compared with (c) is a multi-person human pose recognition image implemented by OpenPose, which can obviously conclude the feasibility of OpenPose to recognize multiple human postures; (d) is the improved multi-person human posture estimation, adding a human frame for each human body in the picture, (c) is compared with (d), although (c) can identify multiple human postures well, but its model is complex, and the speed is higher when implemented, and the improved model is significantly faster when testing. This implies that the improvement not only focuses on enhancing performance but also achieves a significant enhancement in the efficiency of model execution. This comprehensive optimization provides the model with stronger and more efficient support across various applications.

V. CONCLUSIONS

Human pose estimation and recognition are widely used in modern society, for example, it is extremely convenient for people's lives, so the research on the lightweight of pose estimation has the significance of promoting the development of society and meeting the needs of people and the market. As science and technology progress, a lightweight algorithm for human pose estimation is introduced, which is more conducive to our research on pose estimation, so this paper then studies the OpenPose bottom-up convolutional neural network method proposed in 2017 and improves its model and network structure to achieve lightweight multi-person human pose estimation. According to the results realized, it is also verified that lightweight pose estimation reduces the amount of computation and reduces the complexity of the model compared with traditional OpenPose pose estimation, and is more suitable for mobile hardware devices. However, for the current mainstream lightweight human pose estimation, there are still many areas for improvement in this study:

- A splitter can be used to divide each recognized action, and each recognized action is type-output.
- When extracting features, more accurate and fast methods such as residual network ResNet18 can be used.
- You can also train your own model by collecting your own data set and apply human pose estimation to specific applications, such as: intelligent monitoring with the function of detecting falls, AI fitting to meet social fear and deep home, etc.
- Expand the study of 2D human posture to 3D research.
- Connected with software, a system can be created to detect the type of human movement.

REFERENCES

[1] A. Newell, K. Yang, and J. Deng. Stacked hourglass networks for human pose estimation. In ECCV, 2016.

[2] P. F. Felzenszwalb and D. P. Huttenlocher. Pictorial structures for object recognition. In IJCV, 2005.

[3] M. Andriluka, S. Roth, and B. Schiele. Monocular 3D pose estimation and tracking by detection. In CVPR, 2010.

[4] K. He, X. Zhang, S. Ren, and J. Sun. Deep Residual Learning for Image Recognition. In CVPR, 2016

[5] L. Pishchulin, E. Insafutdinov, S. Tang, B. Andres, M. Andriluka, P. Gehler, and B. Schiele. Deepcut: Joint subset partition and labeling for multi person pose estimation. In CVPR, 2016.

[6] E. Insafutdinov, L. Pishchulin, B. Andres, M. Andriluka, and B. Schiele. Deepercut: A deeper, stronger, and faster multi person pose estimation model. In ECCV, 2016.

[7] Cao,Z,Simon,T,Wei,S,et al.Realtime multi-perpson 2d pose estimation using part affinity fields [A].// Proc of the IEEE Conference on Computer Vision and Pattern Recongnitio n [C], Honolulu, HI, USA: IEEE, 2017:1302-1310.

[8] M. Kocabas, S. Karagoz, and E. Akbas. MultiPoseNet: Fast multi-person pose estimation using pose residual network. In ECCV, 2018.

[9] S. Hong, B. Roh, K.-H. Kim, Y. Cheon, and M. Park. PVANet: Lightweight Deep Neural Networks for Real-time Object Detection. In arXiv preprint arXiv:1611.08588, 2016.

[10] Z. Cao, T. Simon, S. Wei, and Y. Sheikh. Realtime Multi-Person 2D Pose Estimation using Part Affinity Fields. In CVPR, 2017.

[11] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. In arXiv preprint arXiv:1704.04861, 2017.

[12] B. Xiao, H. Wu, and Y. Wei. Simple Baselines for Human Pose Estimation and Tracking. In ECCV, 2018.

[13] A. Toshev and C. Szegedy. Deeppose: Human pose estimation via deep neural networks. In CVPR, 2014.

[14] U. Iqbal and J. Gall. Multi-person pose estimation with local joint-to-person associations. In ECCV Workshops, Crowd Understanding, 2016.

[15] Li Yifan, Yuan Longjian, Wang Rui. Improved Lightweight Human Action Recognition Model Based on OpenPose % J Electronic Measurement Technology [J]. 2022, 45(01): 89-95.

[16] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. In ICLR, 2015.

Research on Object Detection in Animal Images Based on Convolutional Neural Networks

Yuxin Niu
School of Computer Science and Engineering
Xi'an Technological University
Xi'an, 710021, China
E-mail: niuyx07l@163.com

Zhongsheng Wang
State and Provincial Joint Engineering Lab. of Advanced Network, Monitoring and Control
Xi'an Technological University
Xi'an, 710021, Shaanxi, China
E-mail: wzhsh1681@163.com

***Abstract*—Object detection is the use of computer to find out all the objects of interest in the image, determine their categories and locations, is one of the core problems in the field of computer vision. Traditional animal image target detection usually adopts the sliding window method, but due to the different sizes of the input images, this method has some problems such as insufficient training samples, low detection accuracy and slow speed. In order to solve such problems, based on the development of deep learning in recent years, this paper proposes an object detection algorithm based on convolutional neural network. YOLOv5 was used to effectively distinguish, identify and mark animal categories, which accelerated the training of the model and greatly improved the accuracy of target detection. Through the analysis of experimental data, it was concluded that the algorithm studied in this paper had good performance and good target detection results. Finally, the key problems of object detection research are summarized, and the future development trend of this field is prospected. When the number of training rounds is 30, the accuracy rate has reached about 70%, and after 50 rounds of training, some accuracy can reach 90%, which is excellent and better than other traditional algorithms.**

Keywords-Deep Learning; Object Detection; YOLO Algorithm; Animal Recognition

I. INTRODUCTION

Nowadays, computer information technology has become an indispensable part of human life, such as mobile phone payment, takeaway food that can be eaten without leaving home, convenient and practical smart furniture at home and powerful Visualize system, etc. With the continuous development of network technology and the wide application of monitoring system, a large number of visual information data spread rapidly on the network. These huge data make a variety of imaging equipment gradually replace human vision, and use the powerful computing power of computers to analyze the data [1] , so as to more accurately recognize people and things in the real world. In recent years, convolutional neural network-based methods have been widely used in the field of image recognition. The visual information data required for processing and analyzing information is very complex, which also poses a huge challenge to the progress of computer vision technology.

Animal resources are as valuable as ecological resources and have high scientific, economic and medical value. For human beings, protecting animal resources is to protect human beings themselves. Animal image target detection can be used to estimate the biological richness of an area, help biologists and conservationists understand animal resources, and can help people assess the richness and diversity of wildlife, recognize endangered and non-endangered species, and establish better conservation plans and management mechanisms. Animal image target detection can be used to detect the disease situation of wild animals, real-time monitoring and prevention of wild animal outbreaks. For example, in the endemic areas of African rinderpest and foot-and-mouth disease, the use of animal image target detection technology can realize the rapid identification and isolation of sick animals in order to control and contain the spread of the epidemic. Animal image target detection can also be used to study animal behavior, such as the interspecies interaction of birds, the habitat, migration and

DOI: 10.2478/ijanmc-2023-0076

predation of animals, and the interaction between animals and humans. Through the analysis of animal image target detection results, we can better understand the ecological habits of animals, behavior patterns and the interaction with the environment. Animal image target detection has important practical value and research significance in the fields of ecology, zoology and conservation biology, and is expected to be widely used in practical ecological protection and animal protection.

The research on object detection of animal images based on convolutional neural networks helps to strengthen the understanding of object detection in deep learning and computer vision, deepen the understanding of convolutional neural networks, help to promote the research progress of deep learning and animal detection, and pave the way for further combining deep learning with all aspects of life in the future.

II. RELATED WORK

In the rapid development of computer vision, object detection technology plays a key role, providing basic support for image recognition and understanding. This paper will discuss the overall status of object detection research and the remarkable progress made in this field. It also studies the latest progress of animal target detection, and pays attention to the efforts of domestic and foreign scholars in using deep learning technology to solve the problem of animal recognition. This research direction not only has practical significance for protecting ecological environment and realizing intelligent agriculture, but also provides new challenges and opportunities for the future development of computer vision.

A. Research status of target detection

Computer vision, which was born around 1960, refers to the image recognition technology based on computer algorithms and is the result of the combination of artificial intelligence and cognitive neuroscience. Visual processing also helps us better understand the workings of the human visual system, especially the brain. In the past 20 years, computer vision has experienced two major stages of development, with 2012 as the dividing point. Before 2012, it was mainly the traditional object detection stage, and after 2012, it was the deep learning-based object detection stage [2]. However, due to the limitation of image processing technology and the lack of effective image representation technology at that time, scholars in the traditional object detection stage can only design some complex or non-obvious images to check the computing resources at that time. In 2012, R. Gershick et al. made use of the characteristics of CNN [3] to break through the barriers in the field of target detection, bringing the target detection technology into a new period. In the detection process, algorithms that use deep learning for target detection can be roughly divided into two main lines of development. One is the two-stage algorithm, that is, the detection process is divided into two successive stages. The former uses a network to generate the proposed region, while the latter uses another network to send the proposed region to the classifier for classification [4]. The second is the one-stage algorithm [4] that is, the detection process is only divided into One Stage, and bounding box and classification label are output directly through a network. As a fundamental part of the field of computer vision, object detection has made great progress in the past few years.

Object detection algorithms have made great progress under the deep learning framework, integrating more algorithms and methods to solve real-world problems, and have gradually achieved a balance in terms of speed, accuracy and effectiveness. There are still many difficulties and challenges to overcome in the future, such as the application of small target detection, edge computing and other fields, higher precision and need to face higher complexity of the network structure, but in general, target detection will still be the core field of computer vision.

B. Current status of animal recognition research

At home and abroad, researchers have adopted various methods to explore and improve the animal target detection technology. At present, there are few researches in the field of animal image object detection. In terms of research status at home and abroad, in China, some researchers use deep convolutional neural networks (CNN) to identify wild animals, and some researchers use transfer learning to solve the problem of livestock target detection in agricultural images. However, these methods have some problems in practice, such as insufficient training samples, low detection accuracy and slow speed. Foreign researchers have proposed a variety of animal object detection methods based on deep learning. For example, methods such as Faster R-CNN, SSD, and YOLO are widely used in the field of animal identification. For example, the European ZooScan project is dedicated to the development of automated aquatic organism classifiers [5] , and uses machine learning algorithms and computer vision technology to achieve fast and accurate classification of biological samples. In addition, National Geographic magazine has published an article introducing a deep learning-based animal object detection technology that can identify dogs, cats, rabbits and other animals.

With the continuous development of convolutional neural network technology, foreign scholars have gradually carried out in-depth research on cats and dogs, and applied it to more complex biometric identification field. In 2017, Tibor Tmovszky [6] used shallow convolutional neural networks to classify and identify animals, such as deer, Wolf, pig, fox, etc., with an accuracy rate of 98%. In 2018, Schneider et al. [7]applied Faster R-CNN to the Reconyx Camera Trap and Snapshot Serengeti groups, and achieved good results of 93.0% and 76.7%. In 2019, Li Anqi [8] proposed an image feature extraction method combining ROI and CNN, which was superior to FastR-CNN to a certain extent. Cheng Zhe'an [9] used the existing wildlife database of the Inner Mongolia Horse Racing Reserve to build a deep residual network containing 6 species, including white stork, wild boar, quagga, deer, mink and spores, and modified the balance loss function of Faster R-CNN, with the mAP value reaching 92.2%.

Animal target detection is an important research direction both at home and abroad, and methods to study this problem are constantly emerging. Scholars are insisting on exploring more efficient and accurate animal target detection methods to cope with more complex and changeable practical application scenarios.

III. TECHNICAL MODEL

A. Convolutional neural network

Convolutional neural network (CNN) is a kind of deep learning neural network for processing high dimensional information such as speech, image, video, and natural language.

CNN network generally uses convolution and pooling operations to preprocess the input data to extract the spatial structure information and timing information in the data. The core idea of CNN is that through the neuronal connections between layers, the network can automatically learn the features of the input data, so as to provide more accurate prediction results when the model makes predictions. As shown in Figure 1.

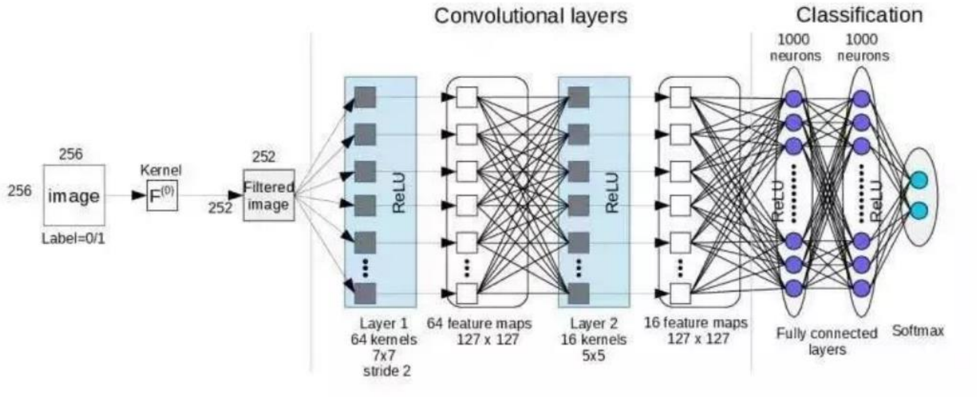


Figure 1. CNN network architecture

B. Object detection algorithm

Target detection is mainly through the input of images or videos, and then use the algorithm to identify the target inside, and determine its position and size, target detection algorithms can usually be divided into the following categories, based on traditional methods; Methods based on deep learning, representative algorithms include Faster R-CNN, YOLO, SSD, etc. Two-stage detector, Faster R-CNN is the representative algorithm; The first stage detector, YOLO and SSD are well-known representative algorithms; Candidate region estimation algorithms, such as RCNN and Faster R-CNN.

C. YOLOv5 algorithm

At present, the YOLOv5 algorithm model is divided into four basic modules: input, Backbone, Neck network and output. Figure 2 shows the structure.

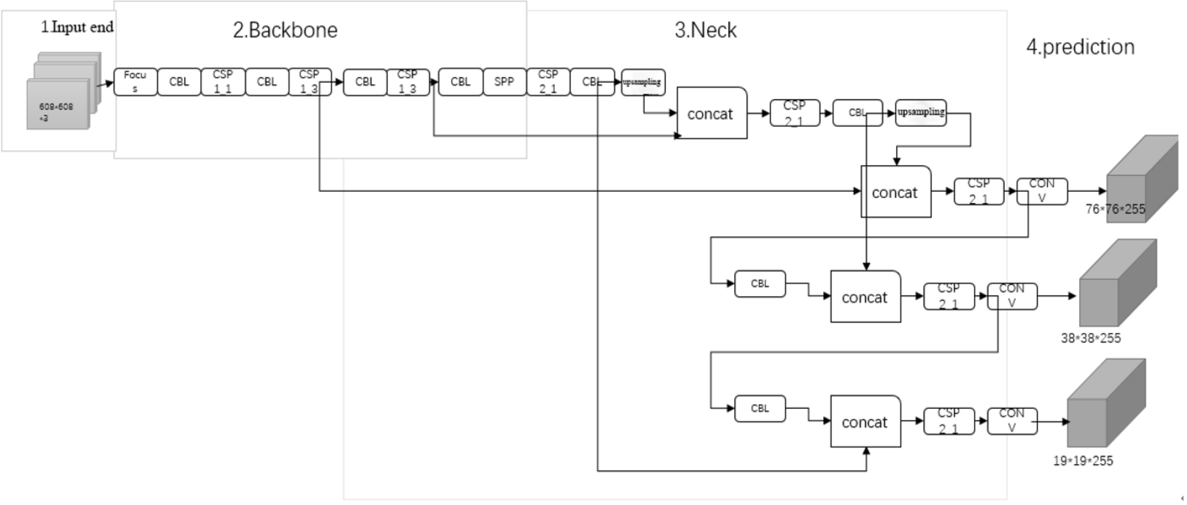


Figure 2. Structure of YOLOv5s

1) Input end

The input image size of YOLOv5 is 608×608, so it generally needs to be preprocessed accordingly. Usually, the original image scale is scaled first and normalized to the range of [0,1]. In the network training part, YOLOv5 not only introduced adaptive anchor frame calculation and adaptive image scaling, but also added Mosaic data enhancement algorithm.

Mosaic data enhancement algorithm. By means of random scaling, random clipping and random arrangement of the four pictures, the data set is enriched, and the training speed and accuracy are improved.

Adaptive anchor frame calculation. This method dynamically adjusts the default bounding box size and shape based on the size and distribution of the object. In the network training stage, the model will output prediction boxes based on these anchor boxes, calculate the difference between them and the real boxes, and then perform reverse updates to update the network parameters [10] . A proper initial anchor box is critical to the accuracy of the target detection algorithm, and for YOLO versions 3 and 4, separate programs need to be set up for different data sets. In YOLOv5, this function has been embedded in the code, which can automatically calculate the best anchor box according to the name of the data set [11] , and the user can turn this function on or off according to the needs during each model training.

Adaptive picture scaling. In order to match different target detection algorithms, we generally scale the image to be detected to a certain size, and then send it to the detection network. However, in the real scene, the aspect ratio of the image is inconsistent, so the commonly used scaling will increase the number of black edges due to the inconsistency of the image, resulting in large data duplication and reducing the reasoning efficiency of the algorithm. In order to further accelerate the inference efficiency of YOLOv5, this algorithm adopts an adaptive algorithm which can minimize the number of black edges added in the reduced image. Figure 3 shows the implementation effect.

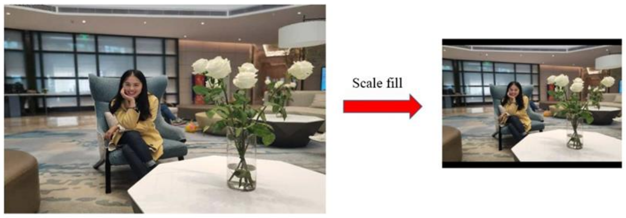


Figure 3. Adaptive picture scaling

2) *Backbone network*

Generally composed of multi-layer convolution and pooling layers, this module is used to extract some common feature representations. The base network of YOLOv5 uses the CSPDarknet53 and Focus structures. The Focus structure is used to slice the images before they enter the backbone network, and convolution joins the sliced images, thus avoiding the multi-convolution kernel structure and retaining important feature information, as shown in Figure 4. For example, in the network structure of YOLOv5s, the original image with input of 608×608×3 is processed by slicing the Focus structure, and then convolution is processed once by 32 convolution kernels to obtain a feature graph of 304×304×32 [12]. However, different versions of YOLOv5 use different numbers of convolution kernels.

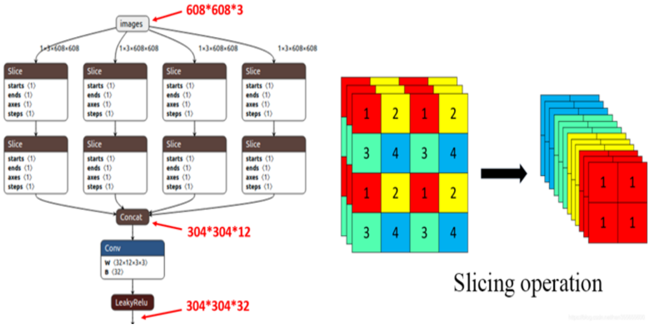


Figure 4. Focus structure

3) *Neck network*

Neck network is between the baseline network and the header network, which can enhance the diversity and robustness of features [13] . Although YOLOv5 also uses FPN+PAN module, as shown in Figure 3-5, the implementation details are slightly different. Specifically, the FPN structure forms a feature pyramid with multi-scale characteristics by gradually fusing feature maps from low to high levels, which enables the model to process objects of multiple scales at the same time, and improves the detection accuracy of small objects. The Path-aggregation part uses a novel Path-aggregation network for depth feature fusion. The feature maps of different resolutions are connected in series. Meanwhile, the PAN structure weights the feature maps of all scales to optimize the scale invariance. The detection of objects of different sizes and scales is realized, and the detection performance of YOLOv5 is effectively improved. FPN solves the problem of heterogeneity in size, PAN solves the problem of scale, FPN layers convey strong semantic features from the top down, and PAN layers convey positioning features from the bottom up.

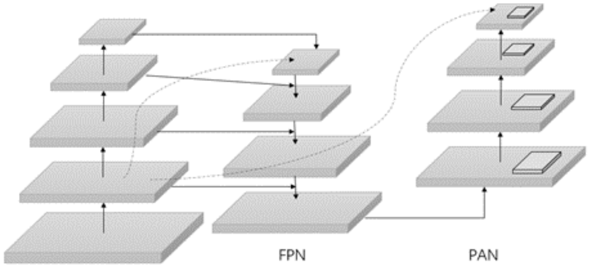


Figure 5. Structure of FPN+PAN

The structure of the whole Neck network is very simple and has very high computational efficiency. In addition, both SPP-Pyramid and Path-aggregation in Neck network can solve some problems existing in YOLOv4, such as false positives caused by unclear counting near edge anchor points and slow network convergence. Therefore, compared with YOLOv4, the Neck network of YOLOv5 not only has better detection performance, but also high computational efficiency and accuracy.

4) *Head end*

The Head end of YOLOv5 refers to the last layer of the network, which is mainly used to predict the location, category and confidence of the target. Compared with YOLOv4, the Head end of YOLOv5 has been slightly improved to improve detection performance and speed.

Specifically, the Head end of YOLOv5 adopts GIoU Loss, as shown in equation (1). First, the minimum closure area of the two frames is calculated (popular understanding: The smallest area of the predicted box and the real box), and then calculate the IoU, and then calculate the proportion of the area that does not belong to the two boxes in the closure area, and finally subtract this proportion from IoU to get GIoU. This loss function is dedicated to bringing the predicted box closer and closer to the real box, it is to put the calculated value of the IoU in a norm term that contains the position and shape information of the bounding box, and redefine the IoU so that it can quantify the "proximity" between the two boxes. In this way, the optimizer will backpropagate on the basis of the GIoU objective function, gradually adjusting the output of the network, balancing the algorithm between classification and reconstruction, and achieving better performance.

$$GIoU = IoU - \frac{|A_c - U|}{|A_c|} \qquad (1)$$

The Head end of YOLOv5 also uses a new predictive method called Dynamic Convolution, which can effectively adapt to the different shapes and sizes of each target, thus improving the detection performance. In addition, the Head end of YOLOv5 also uses a multi-scale prediction mechanism, that is, by using different sizes of anchor points to detect different sizes of targets. In the prediction, YOLOv5 uses the up-sampling method to fuse the feature maps with different resolutions, which can ensure the detection performance is not lost and improve the calculation speed. The non-maximum suppression (NMS) algorithm used in YOLOv5 is a technique used to remove duplicate boxes in detection results. The main idea is to merge boxes whose overlapping area is larger than a certain threshold, and eventually only one box is retained to represent the target.

The NMS is mainly implemented through the following steps.

Step1. Sort all prediction boxes from highest confidence to lowest

Step2. Traverse each prediction box from top to bottom to check whether the IOU value between this prediction box and all subsequent prediction boxes is greater than a certain threshold (generally 0.5 or 0.6). If the value is greater than the threshold, the prediction box is deleted. Otherwise, the prediction box is retained

Step3. Continue through the next prediction box and repeat the above steps until all prediction boxes have been traversed

Step4. The remaining prediction box is the result after the NMS processing, that is, each object corresponds to only one prediction box [14] .

The YOLOv5 algorithm improves the YOLOv4 algorithm. For example, YOLOv5 uses a new computer vision module, namely SPP module, which can make the difference in the size of the feature map irrelevant, which is conducive to enhancing the feature extraction capability. In addition, YOLOv5 also adopts a new Network design, namely CSP Net (Cross-Stage-Partial Network), which can effectively increase the capacity and feature expression capability of the network, so as to improve the detection accuracy. At the same time, YOLOv5 also applies Mix Up and Cut Mix and other technologies in the training process, which can enhance the generalization ability of neural network. Whether it is a new module design or a new training strategy, these improvements have enabled YOLOv5 to make significant progress in the field of object detection.

The YOLOv5 algorithm has the advantages of higher efficiency, faster speed, higher accuracy, and can be trained on a smaller GPU memory, so it is suitable for target detection scenarios of lightweight devices. At the same time, due to its innovative design, YOLOv5 achieved first place in several target detection competition tasks.

IV. EXPERIMENT AND ANALYSIS

A. Experimental content

Animal image datasets are used to evaluate the proposed target detection method. This dataset contains images of cats, dogs, birds, horses and sheep. We conducted experiments on this dataset to evaluate the performance of our method under different rounds of training. Experimental results show that the proposed YOLO algorithm has a good effect in the task of animal target detection. Compared with traditional image-based methods and some previously proposed convolutional neural network methods, the proposed YOLO algorithm shows better results in terms of detection accuracy and processing speed.

The environment used in this experiment was Windows 11 system, PyCharm Community Edition 2020.2.1, Python 3.8 and LabelImg.

B. Experimental process

The general flow chart of the experiment is shown in Figure 6.

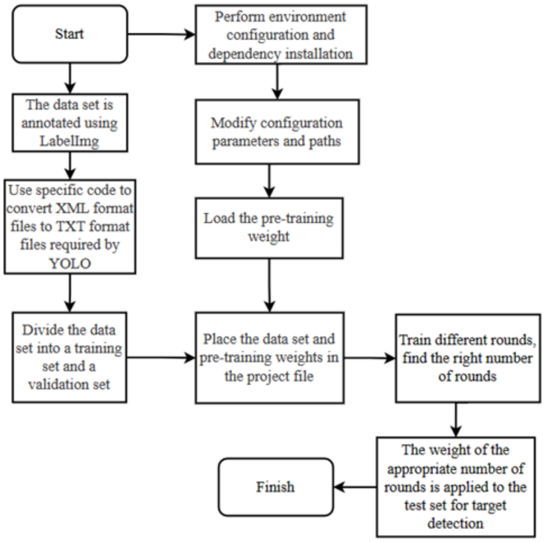

Start
Perform environment configuration and dependency installation
The data set is annotated using LabelImg
Modify configuration parameters and paths
Use specific code to convert XML format files to TXT format files required by YOLO
Load the pre-training weight
Divide the data set into a training set and a validation set
Place the data set and pre-training weights in the project file
Train different rounds, find the right number of rounds
The weight of the appropriate number of rounds is applied to the test set for target detection
Finish

Figure 6. Flowchart of the experiment

First, the target detection data set is annotated using the LabelImg tool. Set VOCdevkit and VOC2007 folders in the project, create JPEG Images in VOC2007 to store unannotated images, store annotated file for Annotations, and create txt file to store the name of the target class. The data set contains five categories and is saved to the Annotations folder after labeling images using LabelImg. Next, convert the XML format to the TXT format required by YOLO. The data set is divided into a training set and a validation set, and the 8:2 ratio is used in this project. Configure the environment, install dependencies, modify parameter paths, and load yolov5s.pt pre-training weights. Download the weights to the project folder, train several times, select the best weights, and train further. Finally, the optimal weight is used to detect the target and the detection result is obtained.

In this experiment, we selected some images from COCO dataset (including horses, sheep and birds) and Cats vs Dogs dataset. COCO dataset is a large and rich object detection, segmentation and captioning dataset, and its images include 91 types of objects, 328,000 images and 2,500,000 labels. There are 80 categories provided, with more than 330,000 images, of which 200,000 are annotated, and the number of individuals in the entire dataset exceeds 1.5 million [15], which are commonly used to train and evaluate advanced computer vision models such as object detection, instance segmentation, and key point detection. The Cats vs Dogs dataset is mainly used for image classification tasks, in which the goal is to distinguish the images of cats and dogs. It is relatively small, including the training set and the test set. The number of pictures of cats and dogs in the training set is 12,500 and sorted in order, and the mixed pictures of cats and dogs in the test set are 12,500 [16]. Used primarily for entry-level computer vision projects to demonstrate how to build and train basic image classification models.

The experiment ran 3, 10, 30 and 50 rounds respectively. After the model is trained, a runs folder will appear, which contains the folders with weights, best is the best weight, last is the weight of the last training, it will automatically do validation, parameters and so on. Next, for inference, first paste the copied and trained best weights into the weights folder, then modify the paths and parameters in the file where detect.py is deduced, and finally put the image path to be detected into the folder in the inference path and run detect.py.

C. Experimental result

The specific data obtained from the experiment are shown in the following figures, including the accurate information of a series of curves during the whole training process and the values of different loss functions, which are visually displayed in the form of matrix and discount.

The experimental confusion matrix is shown in Figure 7, where each row represents the predicted category and each column represents the true belonging category of the data. The following figure shows that there is little difference between the predicted results of cats, dogs, birds and horses and the true category, but the predicted results of sheep category are not good.

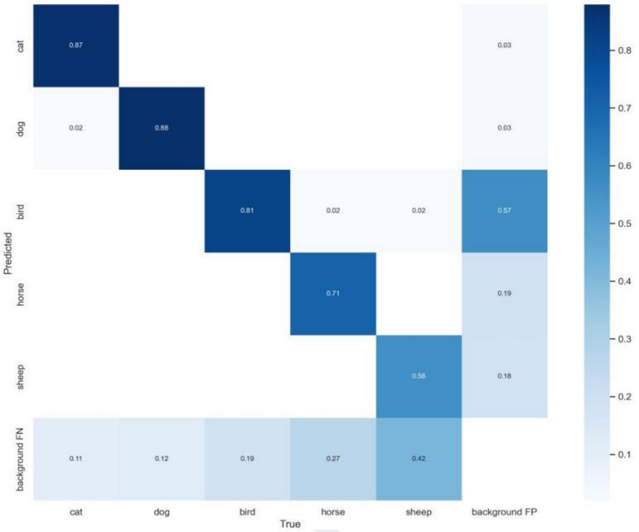


Figure 7. Confusion matrix

As shown in Figure 8, the experimental PR diagram draws a curve between the accuracy rate and the recall rate, showing the performance of the model under different thresholds. The recall rate, that is, the probability that the correct category in the sample is predicted correctly, is shown in formula (2), where TP indicates that the correct category is predicted as the correct category number, and FN indicates that the correct category is predicted as the negative category number.

$$recall = \frac{TP}{TP + FN} \qquad (2)$$

The experimental RCC curve is shown in Figure 9, which is a visual representation of the change of recall rate under different confidence thresholds. It is shown in the figure that except for sheep category, other categories can still maintain a high recall rate after filtering out the prediction box with low confidence, indicating that the model has a good performance.

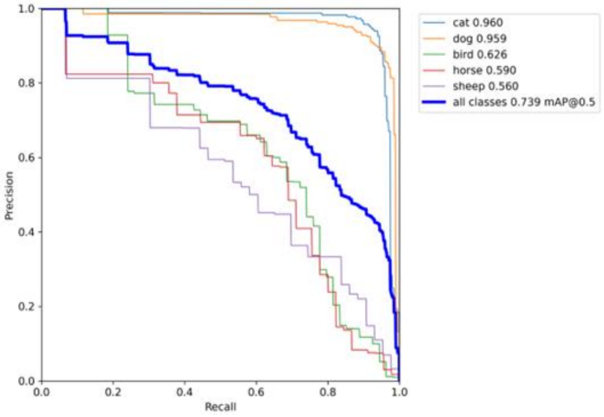


Figure 8. PR curve

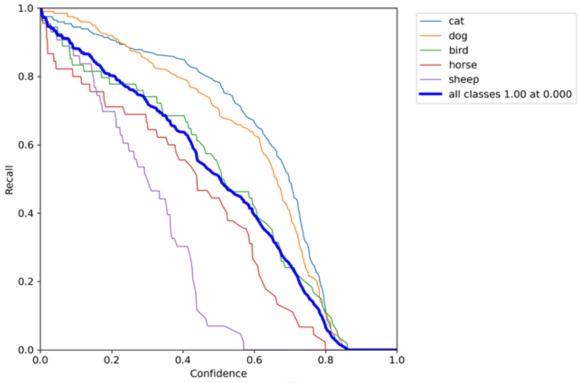


Figure 9. RCC curve

The general figure of the training results of the algorithm on the data set is shown in Figure 10, which comprehensively evaluates the performance of the classification model through different indicators.

The experimental results are shown in Figure 11 and Figure 12, showing the test results of different kinds of animals.

Different rounds of training were conducted in the experiment, and the accuracy, mAP50 and MAP50:95 values were different, as shown in Table 1.

TABLE I. DIFFERENT ROUNDS OF TRAINING

Epoch	Recall	mAP50	mAP50:95
3	0.062	0.009	0.015
10	0.259	0.220	0.119
30	0.507	0.520	0.248
50	0.753	0.739	0.350

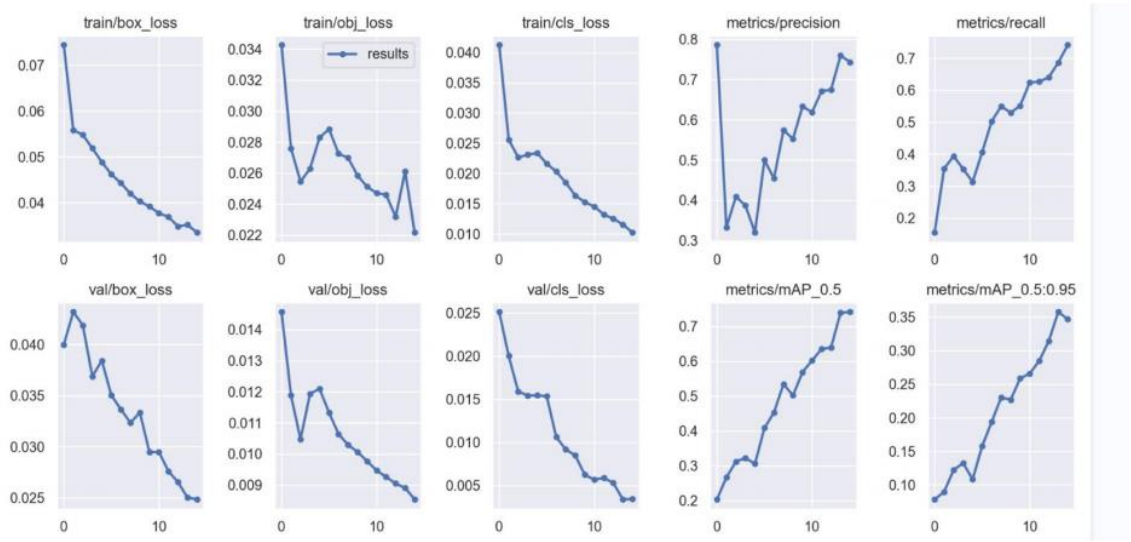


Figure 10. Training results of the algorithm on the data set

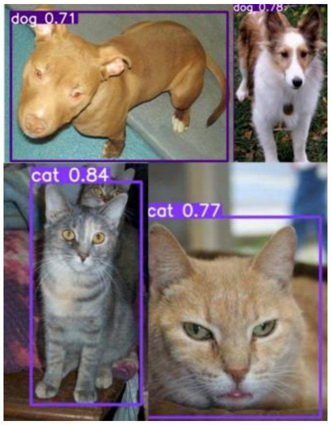


Figure 11. Test results of different types of animals(1)

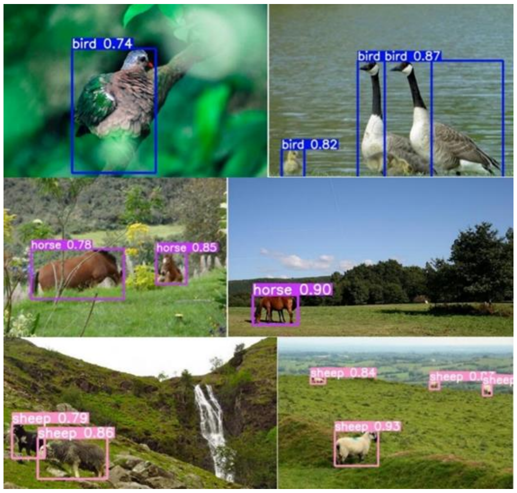


Figure 12. Test results of different types of animals(2)

The above are the results of animal image detection based on YOLOv5 algorithm. This experiment requires different rounds of training, but the increasing number of training rounds will lead to the increase of network pressure, and the training is relatively time-consuming, so this experiment only trains 3 rounds, 10 rounds, 30 rounds and 50 rounds. The accuracy rate, recall rate and loss rate of each round of training are observed. It can be observed that when the number of training rounds is 50, the detection effect is the best, the average accuracy is more than 70%, and the accuracy of some categories can reach 90%. In the later stage, the category of data set and the number of training rounds can be gradually increased to find out whether there is a better detection result.

V. Conclusions

The traditional target detection algorithm has some disadvantages, such as tedious process, fast convergence and difficult learning, which lead to poor target detection results. The algorithm in this paper is based on a specific model for detection. In order to solve the problems caused by the traditional algorithm, the algorithm will be trained and learned in advance. Through multiple training with different rounds, the training results are compared, and the appropriate number of rounds is selected for image detection. The algorithm in this paper still has some problems, such as insufficient sample class richness, variable target scale and Angle, etc. In order to further improve the detection accuracy and practical application, future research can consider the introduction of multi-scale training, image enhancement and other technologies, and combine more features and information to improve the detection effect of animal targets in complex scenes in terms of model performance, application expansion, interpretability research and other aspects.

REFERENCES

[1] Gao Hui. Research on Video Object Detection Algorithm Based on Deep Learning [D]. University of Electronic Science and Technology of China, 2021.

[2] Zhang Xin. Image Recognition System of Engineering Vehicle Equipment Based on Deep Learning [D]. Xidian University, 2021.

[3] Zhao Yongqiang, Rao Yuan, Dong Shipeng et al. Review of Deep learning object detection methods [J]. Journal of Image and Graphics, 2019, 25(04):629-654.

[4] Fan Lili, Zhao Hongwei, Zhao Haoyu, et al. Optics and Precision Engineering, 2020, 28(05):1152-1164.

[5] Wang Yuzheng, Cheng Yuan, Bi Hai et al. Marine single-cell algae recognition algorithm based on deep learning VGG network model [J]. Journal of Dalian Ocean University, 2021, 36(02):334-339.

[6] Trnovszky T, Kamencay P, Orjesek R, et al. Animal recognition system based on convolutional neural network [J]. Advances in Electrical and Electronic Engineering, 2017, 15(3):517.

[7] Schneider S, Taylor G W, Kremer S. Deep learning object detection methods for ecological camera trap data [C]. IEEE Conference on Computer and Robot Vision, 2018: 321-328.

[8] Li Anqi. Research on automatic recognition of wildlife monitoring images based on Convolutional neural networks [D]. Beijing Forestry University, 2020.

[9] Cheng Z A. Automatic recognition of terrestrial wildlife in Inner Mongolia based on deep convolutional neural networks [D]. Beijing Forestry University, 2019.

[10] Ma Linlin, Ma Jianxin, Han Jiafang et al. Research on Object Detection Algorithm based on YOLOv5s [J]. Computer Knowledge and Technology, 2021, 17(23):100-103.

[11] Zhou Wenhui, JIA Yonghong, Jiao Yang. Research on detection method of Chinese sturgeon in underwater video [J]. Computer Science and Applications, 2022, 12(8): 1998-2005.

[12] Fan Youchen, Ma Xu, Ma Shuli et al. Evaluation method of laser interference effect based on deep learning [J]. Infrared and Laser Engineering, 2021, 50(S2): 39-45.

[13] Lin Sike, Chen Jinwei, Huang Sihua. Research on Student Behavior Detection based on Deep Learning [J]. China Journal of Multimedia and Network Teaching (Mid-day), 2022(06):237-240.

[14] Jin Y. Research on pig quantity monitoring method based on machine vision [D]. Jiangxi Agricultural University, 2021.

[15] He Yuzhe, He Ning, Zhang Ren et al. Research on Training Unbalance of Deep Learning-oriented object detection Model [J]. Computer Engineering and Applications, 2022, 58(05):172-178.

[16] Xu Bo. Research on Object Detection and Semantic Segmentation Algorithms based on Deep Learning [D]. Northeastern University, 2019.

Object Localization Algorithm Based on Meta-Reinforcement Learning

Han Yan
Xi'an Technological University
School of Computer Science & Engineering
Xi'an, China
E-mail: 18713877573@163.com

Hong Jiang
Xi'an Technological University
School of Computer Science & Engineering
Xi'an, China
E-mail: 249479898@qq.com

Abstract—**When the target localization algorithm based on reinforcement learning is trained on few-sample data sets, the accuracy of target localization is low due to the low degree of fitting. Therefore, on the basis of deep reinforcement learning target localization algorithm, this paper proposes a target localization algorithm based on meta-reinforcement learning. Firstly, during the initial training of the model, the meta-parameters were classified and stored according to the similarity of the training tasks. Then, for the new target location task, the task feature extraction was carried out and the meta parameters with the highest similarity were matched as the initial parameters of the model training. The model dynamically updated the meta parameter pool to ensure that the optimal meta parameters of multiple different types of features were saved in the meta parameter pool, so as to improve the generalization ability and recognition accuracy of multiple types of target location tasks. Experimental results show that in a variety of single target localization tasks, compared with the original reinforcement learning target localization algorithm, under the same data set size, the model converges under a small number of training steps with the meta-parameters in the matching meta-parameter pool as the initial training parameters. Moreover, the training speed of the meta-reinforcement learning method based on MAML-RL is increased by 28.2% for random initial parameters, and that of the meta-reinforcement learning method based on this paper is increased by 34.9%, indicating that the proposed algorithm effectively improves the training speed, generalization performance and localization accuracy of object detection.**

Keywords-Meta-reinforcement Learning; Meta-Parameter; Target; Generalization Ability; Deep Reinforcement Learning

I. INTRODUCTION

Humans can quickly find a new object in the field of vision without too much complicated process, because humans have mastered the ability to learn quickly. This is very difficult for computers, especially for the object localization process, which often requires a large number of datasets and computational costs for training new tasks. This leads to a low degree of model fit and accuracy of target localization for real-world few-sample data. It is particularly important to improve the convergence speed of the model for new tasks by storing and learning the model's historical experience.

With the introduction of reinforcement learning, the accuracy of target positioning has been improved to a certain extent [1], and the typical algorithms are RAM [2] and UR-DRQN positioning models [3]. This kind of algorithm regards the process of target localization as the process of Agent constantly interacting with the task environment, getting positive and negative rewards to update the model parameters, and finally locating the target. Such algorithms need to train agents according to different task objectives and require a large amount of labeled data, so they will have the problems of low fitting degree and slow convergence speed when facing few-sample tasks [4].

Meta reinforcement learning is an important field of machine learning research. It is a method that enables agents to learn and converge quickly when facing new tasks by training a global optimal parameter as the initial parameters of model

DOI: 10.2478/ijanmc-2023-0077

retraining. Among them, MAML-RL [5] is the most classical method in meta-reinforcement learning. Its core idea is to optimize the model through multiple kinds of tasks to train an initial parameter, so that the model can quickly converge on a new task with only a small number of samples or a few gradient updates. The existing target localization algorithms based on meta-reinforcement learning improve the generalization ability of the model by training the optimal parameters of the task, but with the improvement of the task type, its learning ability will decrease. In view of this, this paper sets up a memory storage module to save the training parameters of historical tasks according to similarity, and conducts meta-learning operations on them. It can significantly improve the convergence speed of the model for new tasks and alleviate the problem of reduced learning ability of the model.

II. RELATE WORKS

Meta-reinforcement learning has achieved great success on many complex and high-dimensional tasks [6]. Although reinforcement learning provides a new solution for object localization algorithms, it mainly focuses on the localization efficiency under a certain task, rather than generalization in multiple scenarios and rapid adaptation to few-shot tasks. The meta-reinforcement learning method effectively learns new tasks through agent learning in reinforcement learning environment t [7-8]. Existing meta-reinforcement learning methods mainly focus on model-free methods [9-10]. These algorithms tend to have more complex training pipelines than non-meta reinforcement learning methods, making it difficult to apply to real-world applications. Moreover, the existing model-free methods [11] tend to ignore the attenuation of learning ability for new tasks, which reduces their training efficiency when the types of tasks increase.

Some meta-reinforcement learning designers improve the learning ability of the model by designing the architecture of the model or designing new optimization algorithms and update rules. The typical meta-reinforcement learning algorithm MAML-RL [5] obtains a set of initial parameters of the model through training, so that the model can maximize the performance of a new task by only one or a few gradient updates on a small number of samples. On this basis, a storage and replay memory pool is designed to classify and update the meta-reinforcement learning parameters according to tasks, so that the parameters in the memory pool have the maximum generalization performance within the range of task types. In addition, our method allows the model to match the appropriate historical memory according to the new task, and allows the model to automatically adapt to the leap of task types with large differences, thereby reducing the amount of data required for the model to learn new tasks.

III. MODEL

In the target localization algorithm of reinforcement learning, a large number of data sets are usually required for training. However, in real life, there are many kinds of tasks with few samples, for which the bottleneck of localization accuracy is easy to be reached [12]. In object localization algorithms based on meta-reinforcement learning, a set of initial parameters that can converge quickly on new tasks is trained by learning the commonality of task types. This paper combines meta-reinforcement learning on the target localization framework based on reinforcement learning, and learns the optimal parameters of tasks with high similarity by setting a storage mechanism.

Figure 1 shows the framework diagram of the proposed algorithm model, which is mainly composed of three parts: the target localization module, the feature mapping module and the meta-parameter pool module. In the feature mapping module, the improved VGG-16 [13] network extracts the features of the task, and classifies and maps them into the corresponding feature space. The model first uses the Training Data set to train in the reinforcement learning target localization model, records the convergence parameters and loss gradient of each task type, and updates the gradient of the meta-parameters after the training of each task type. The updated model parameters are stored in the meta-parameter pool module according to the mapping area of the feature mapping module. The parameters in the meta-parameter pool are updated by using the meta-parameter update function, and the updated parameter a Meat$_\theta^i$ shows the global optimum in the feature region i.

The purpose of meta-reinforcement learning is to make the model learn the commonality under multiple task types, and then master a learning ability to quickly converge under new tasks. For few-shot data in reality, the proposed model uses the feature mapping module to match the meta-parameters in the meta-parameter pool as the initial training parameters. The meta-parameters preserve the historical exploration experience of the model, and the gradient correction of the few sample data can have better positioning accuracy for the new target.

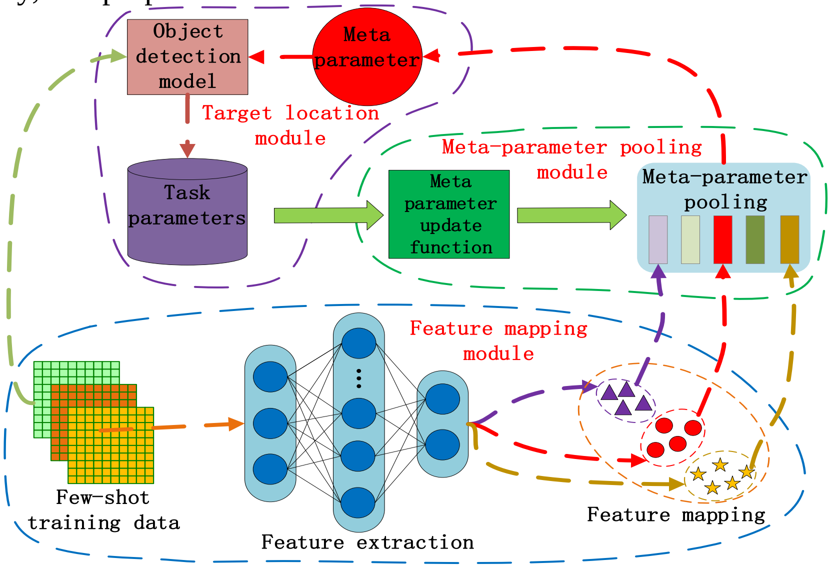


Figure 1. Process of target localization in meta-reinforcement learning

A. *Target positioning module*

As shown in Figure 2, this paper uses the reinforcement learning target localization model with joint action network and regression network as the task training model of the model. It mainly consists of three parts: feature extraction network, action network and regression network. The feature extraction network is the improved GAP-VGG16 network. The model matches the task to the feature space corresponding to the meta-parameter pool according to the feature values extracted by the feature extraction network, and stores the updated parameters after completing a batch of training. At the same time, the feature vector extracted by the feature is fused with the memory vector and sent to the action network. The action network is responsible for taking adjustment actions according to the current environment state, until the stop action is generated, the regression network is used for regression operation, and the final positioning result is output.

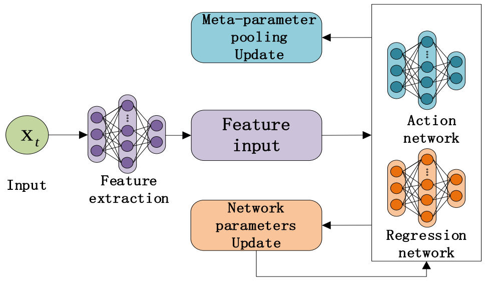


Figure 2. Object localization model

For the few-shot object localization task, the model extracts a few samples from the task for feature extraction, and matches the meta parameters with the largest similarity from the meta parameter pool according to the feature value type extracted by the task as the initial parameters of the few-shot object localization task for retraining. The parameters of the retrained model are updated in the corresponding meta-parameter pool to retain the newly learned task memory. Through the method of this paper, the model has a certain ability to learn the task at the beginning, avoiding the agent to explore in a completely unfamiliar environment. The model regards each image input as a reinforcement learning environment, and selects the exploration action according to the fusion state of the input and historical exploration. The model gives feedback according to the pre-designed reward function to judge the quality of the model action selection, and updates the network parameters of the model through the process of circulation to improve the accuracy of target localization. The detailed design of the model states, actions, and rewards is as follows.

B. State

The process of human searching for the target is not only related to the current visual field, but also involves the memory of the past historical exploration in the brain. Human beings realize the accurate recognition of the target by combining the brain memory with the current visual field. The state S of this paper is the procedural simulation of this process, which is represented by a tuple $\mathrm{s}_{\mathrm{t}} = (\mathrm{o}_{\mathrm{t}}, \mathrm{h}_{t})$ related to time t, which represents the fusion information of o_{t} and h_{t} , and the agent makes the next action selection according to this fusion state s_{t} .

C. Actions

The action taken by the Agent acts on the adjustment of the candidate box, which is divided into horizontal movement (left and right), vertical movement (up and down), scale transformation (horizontal expansion, horizontal reduction, vertical expansion, vertical reduction), and stop action). Each action is adjusted discretized according to the multiple of. Among them, the output termination action indicates that the target is in the field of view of the agent. The specific classification of actions is shown in Figure 3.

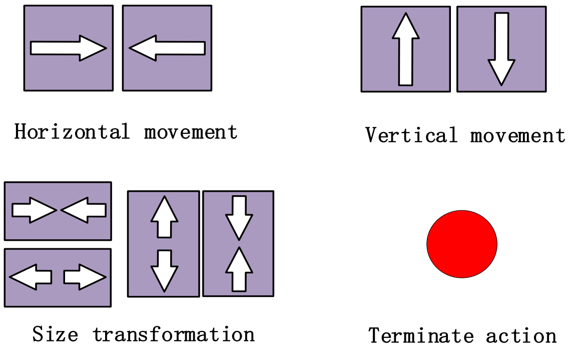


Figure 3. Action diagram

The model selects the actions of the agent through the DQN network, and uses $\varepsilon - \text{greedy}$ [15] strategy to make the agent explore new actions [16], so as to ensure that the agent takes the optimal action under long-term exploration. $\varepsilon - \text{greedy}$ strategy is shown in (1).

$$\pi(\mathrm{a}|\mathrm{s}) \leftarrow \begin{cases} 1\text{-}\varepsilon + \dfrac{\varepsilon}{|\mathrm{A}(s)|}, if \quad \arg\max_{a} Q(s,a) \\ \dfrac{\varepsilon}{|A(s)|}, if \quad \arg\max_{a} Q(s,a) \end{cases} \tag{1}$$

Where s is the current state the Agent is in, and a is the action taken by the Agent based on the current state. A(s) is the set of actions that the Agent can choose at states, and |A(s)| denotes the number of actions that can be chosen. $\varepsilon \in [0,1]$ Is the exploration factor, and $\pi(a \mid s)$ is a policy, which represents the probability distribution of possible actions taken by the agent at a given state s. For the state at a given time in the policy, the agent selects the action corresponding to the output with the maximum probability to adjust the attention field.

D. Rewards

The good or bad of the action taken by the agent can be intuitively seen through the reward function, and in this paper, the reward function is set by the IOU change of the attention field after the state change s after the agent takes the action. As shown in (2), where b is denoted as the visible area of the Agent and g is the real labeled area of the target object.

$$IOU(b,g)=\frac{area(b\cap g)}{area(b\cup g)} \tag{2}$$

At each time step, after taking an action, the agent will obtain a new visual area. By calculating IOU between this area and the real area, the reward value of the agent's state change after taking an action can be obtained, which is defined by the reward function shown in (3).

$$\mathrm{R}_a(a,s\rightarrow s')=sign(IOU(b',g)-IOU(b,g)) \tag{3}$$

This function indicates that after the agent's state has changed. If the value of IOU increases upward, it means that the agent has obtained positive feedback, and the model will store this state and action tuple (s,a,r,s',b,g) as experience in the experience pool, which is used as a reference for the agent to explore the target position. On the contrary, if the IOU decreases after the state change, it indicates that the action is poor and negative feedback is obtained. For the determination of stop action, when the IOU value rises above 0.6 after the agent makes an action, it is determined that the target is in the field of view of the agent, and the stop action is taken, and the regression network is selected to take a smaller step to fine-tune the field of view frame. The reward function for the stop action is given in (4).

$$\mathrm{R}_t(s\rightarrow s')=\begin{cases}+\eta & if\ IOU(b,g)\geq\tau\\ -\eta & otherwise\end{cases} \tag{4}$$

In order to ensure the training efficiency, when the IOU of agent reaches 40 steps, it is determined that the exploration fails, and the regression network is not used for fine-tuning. (7) is used to update the parameters of the two networks.

E. Action Network Structure

The action network consists of two parallel fully connected networks with the same structure and different parameters. One produces the "predicted value" and the other produces the "target value". In the training phase, the "target value" is calculated to assist the learning of network parameters. In the testing phase, the "target value" is not calculated, but when the fusion information ht is received, a random action is selected with the help of the ε-greedy strategy with probability ε, as shown in Figure 4:

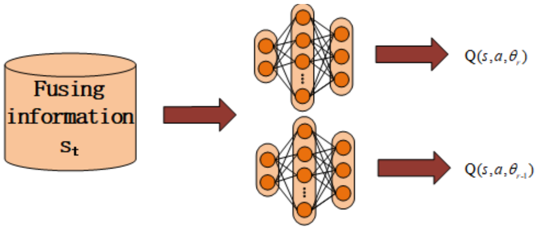


Figure 4. Structure of the location network

F. Regression network structure

The regression network is a fully connected network $fc(128*128*4)$. When the termination action is generated during the learning of an epoch, and the IoU between the visible area and the real marked area is greater than 0.6, the network will fine-tune the coordinates of the current visible area to obtain the offset that needs to be adjusted in the corresponding direction of the bounding box $(\nabla\mathrm{X},\nabla\mathrm{Y},\nabla W,\nabla H)$, as shown in Figure 5:

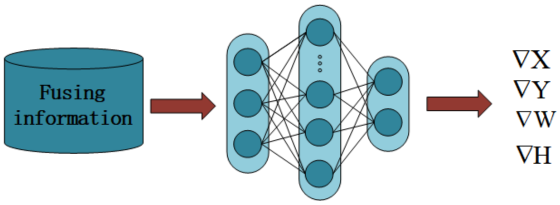


Figure 5. Structure of the regression network

G. Feature space mapping

The tasks of the training set are mapped to different positions in the feature space by means of feature mapping for the generation of meta-parameters. Due to the problem of parameter redundancy and computational complexity in the original VGG-16 network, in this paper, the original fully connected layer is replaced by the global average pooling layer, and GAP-VGG16 is constructed as the feature extraction network, as shown in Figure 6. The feature types of each task can be obtained through feature extraction, and the number of feature types is controlled by specifying the range of mapping (set to 10 in this paper), and each feature range corresponds to the storage space in a meta-parameter pool.

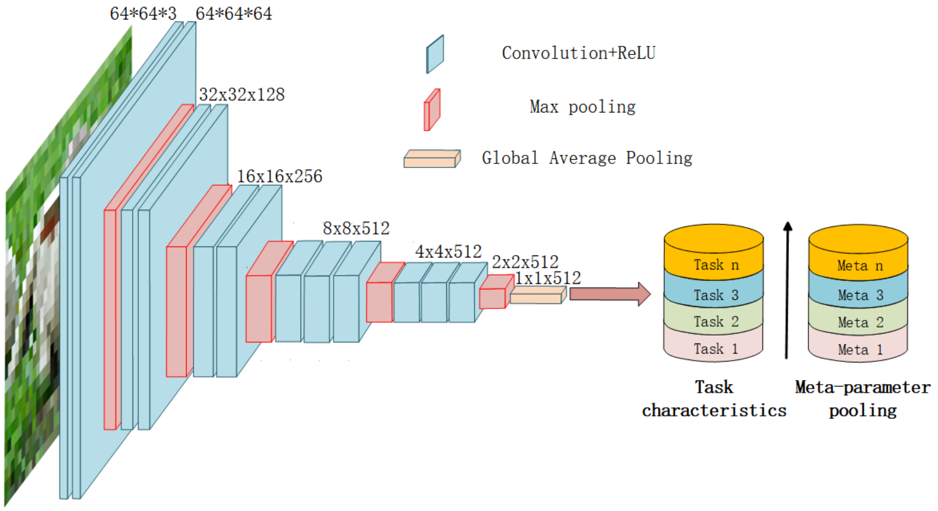


Figure 6. Feature network structure

The feature mapping result of few-shot task is vector form a_i. For each meta-parameter Meta_θ obtained by training, there is a feature vector b_i corresponding to it, and the corresponding meta-parameter Meta_θ is matched by measuring the difference $d(a_i,b_i)$ between vectors a_i and b_i .In this paper, the Euclidean distance between two vectors is used to judge the mapping space region, as well as the degree of similarity between tasks. The calculation formula is given in (5).

$$d(a_i,b_i)=\sqrt{\sum_{i=1}^{n}(a_i-b_i)} \qquad (5)$$

In this paper, the feature vector corresponding to the task of the first meta-parameter of model training is used as the benchmark vector, as the label of 1 meta-parameter stored in the meta-parameter pool position, the difference value κ is set, and the multiple of the difference value κ is set to select the meta-parameters in the model meta-parameter pool, that is, the feature extraction of a few-sample training task is carried out. The similarity between the extracted feature vector and the feature vector label corresponding to the first position of the meta-parameter pool is calculated, and the matched meta-parameter position is obtained by dividing it with κ and adding 1. As shown in (6).

$$meta_{\theta_i}=\frac{d(a_i,b_1)}{\kappa}+1 \qquad (6)$$

Here, $Meta_{\theta i}$ represents the initial meta-parameters matched during retraining of the i^{th} few-shot dataset, and b_1 represents the feature vector corresponding to the meta-parameter at the first position of the meta-parameter pool.

In order to avoid forgetting the historical tasks of the meta-parameters, this paper uses (7) to update the meta-parameters, and retains the previous memory at each update of the meta-parameters. As shown in (7).

$$\text{Meta}_{\theta n}=Meta_{\theta n}(1-n\lambda)+\lambda(\theta_1+\theta_2+....\theta_i) \qquad (7)$$

IV. MODEL TRAINING

The whole training process of the model includes the training of the parameters θ_t of the inner recurrent network in the meta-reinforcement learning process, and the update of the meta-parameter Meta_θ in the meta-parameter pool. In the outer loop, the model updates the meta-parameter pool according to the feature mapping region of the task. The same network architecture is used to update the action network and the regression network in the inner loop.

A. Meta-parameter pool training

The meta-parameter pool O stores the meta-parameters of N kinds of tasks and updates them. For the new training task, the region Oi in the meta-parameter pool is matched according to the way of feature mapping, and the corresponding meta-parameters Meta_θ are selected as the initial parameters for retraining. The retrained meta-parameters allow the retention of the previous memory, and the trained meta-parameter pool maintains the optimal loss value for tasks of the same task type. The update process of the meta-parameter pool is shown in Figure 7.

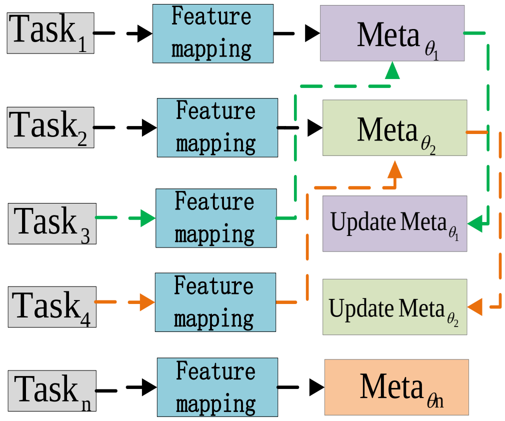


Figure 7. Training process of meta-parameter pooling

The meta-parameters in the meta-parameter pool are updated by continuous learning, and (7) is used to preserve the historical memory when the agent is updated. Where θ_i represents the meta-parameter after the i^{th} update and represents the learning coefficient, which is used to prevent the model distortion caused by too large parameter changes.

B. Training of target localization parameters

The parameters of the target localization model include the parameters of the action network and the regression network, namely $\theta = (\theta_a, \theta_g)$. The historical experience of the action network is represented by the tuple (s, a, r, s', b, g), as shown in Figure 3. There are multiple exploration tasks in the same environment, and each exploration task will generate an MDP sequence. Expressed as the strategy of the agent, the loss function of each task Ti is shown in (8).

$$\theta_i^* = \theta - \alpha \nabla L_{Ti}(f_\theta) \tag{8}$$

C. Loss function

The comprehensive loss of the target localization model includes the loss of the action network and the loss of the regression network and (7) is the loss function. The weighted sum of the losses of the action network and the regression network is used as the comprehensive loss of the target localization model. The mean square error loss function is used for the action network and the smoothL_1 loss function is used for the regression network. The overall loss function is defined as in (9) - (11).

$$L(s, a, t) = L_{action} + \lambda L_{reg} \tag{9}$$

Among them,

$$L_{\text{action}} = \frac{1}{N_{\text{action}}} \sum [(y_i - Q(s, a; \theta_i))^2] \tag{10}$$

$$L_{\text{reg}} = \frac{1}{N_{reg}} S(t_i - t_i^*) \tag{11}$$

Where N_{action} and N_{reg} are the number of execution steps of the action network and the number of execution steps of the regression network, and their losses are averaged as the loss of the exploration action and the regression action. $Q(s, a; \theta_i)$ is the predicted value derived from the "prediction branch" of the action network, and y_i is the target value derived from the "target branch" of the action network. In the regression network loss L_{reg} , $t_j = \{t_x, t_y, t_h, t_w\}$ denotes a vector of dimension size 4. Here, $\sum L_{Ti}(f_\Phi)$ and t_y are the center coordinates obtained by the regression network, respectively, and t_h , t_w are their corresponding heights and widths. t_x^* and t_y^* are the center coordinates of the true labeled regions of the regression network, respectively, and t_h^*, t_w^* are the corresponding heights and widths. Where S is the smoothL_1 function, see (12).

$$soomthL_1(x) = \begin{cases} 0.5x^2, if \ |x| < 1 \\ x| - 0.5, otherwise \end{cases} \tag{12}$$

a is the loss balance coefficient, which is used to balance the loss of the action network and the regression network, and s stands for the regression loss.

D. Meta-reinforcement Learning parameter training

Each RL goal localization task T_i contains an initial state distribution L_{T_i} and transition distribution $q_i(x_{t+1} | x_t, a_t)$, and the loss L_{T_i} corresponds to the negative reward function R. In this paper, we treat each object localization task as a Markov Decision Process (MDP) with an initial state S0, which allows the agent to retrace historical exploration trajectories and perform learning across tasks T_i the model is defined with respect to task T_i and loss $L_{Ti}(f_\varphi)$ as shown in (13).

$$L_{Ti}(f_\varphi) = -E_{x_t, a_t \sim f_\varphi,\ q_{\mathrm{Ti}}}[\sum_{t=1}^{H} \mathrm{R}_i(x_i, a_t)] \qquad (13)$$

For each of the k tasks Task1-k in the meta-parameter pool O_i , the model generates H exploration actions (x1,a1,... xH,aH) and the corresponding loss $\mathrm{L}_{Ti}(f)$, K accumulated losses $\sum L_{Ti}(f_\Phi)$ generated for the model are used for the meta-parameter $\mathrm{Meat}_\theta(\mathrm{T}_{1\text{-k}})$ corresponding to the task type in the meta-parameter pool O_i region, see (14).

$$\mathrm{Meta}_\theta = \theta\text{-}\beta\nabla_\theta \sum_{\mathrm{T}_{1-k} \sim p(T)} L_{T_i}(f_{\theta_i^{'}}) \qquad (14)$$

Using meta-parameters when training on a new task leads to convergence in fewer exploration steps. In this paper, meta-reinforcement learning includes two parts *InnerLoop* and *outerLoop* , and the agent uses network random parameters as initialization parameters in each task pair. In *InnerLoop* , the agent continuously explores the target to generate an *MDP* -sequence, which converges to θ_i^* through multiple training gradient updates. The outer loop of s is classified according to the feature types of the training tasks, and the corresponding meta-gradients of s are cumulatively updated to obtain the meta-parameters of multiple task types, that is, the meta-parameter $\mathrm{Meta}_\theta^{\mathrm{i}}$ maximizes the sum reward of multiple task rewards. The trained meta-parameter $\mathrm{Meta}_\theta^{\mathrm{i}}$ is stored in the meta-parameter pool module, waiting for matching and updating during retraining. Figure 8 shows the parameter training process of meta-reinforcement learning.

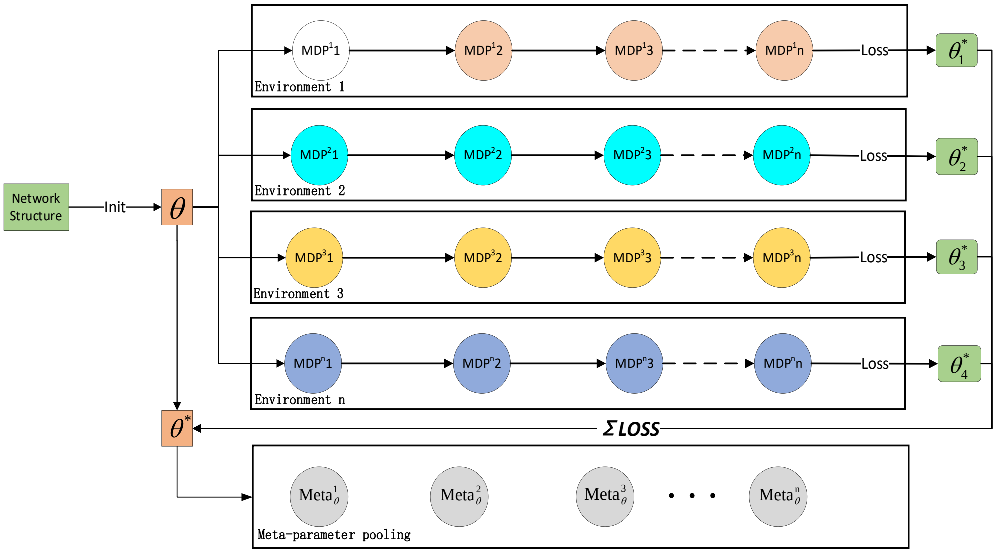


Figure 8. Parameter training process of meta-reinforcement learning

V. EXPERIMENT AND RESULT ANALYSIS

A. Experimental platform and parameter setting

In this paper, Torch deep learning platform is used to train model parameters with each kind of reinforcement learning object recognition task in the joint data set of VOC2007 +VOC 2012 as a task type, and the training of the object localization model is tested with the Test data set of VOC2007. In order to test the generalization performance and convergence efficiency of the proposed model for new tasks, six samples of different task types in VOC 2007+VOC 2012 dataset are selected for retraining. For the other 14 kinds of task datasets used for training update model meta-parameter pool. The experiment measured the convergence speed of the target localization model with different initial parameters, the localization precision (ap), the return rate (recall) of the trained model, and the number of required positioning steps.

B. Meta reinforcement learning Loss comparison plot

In this section, the random initial parameter θ_{random}, the MAML-RL based meta-parameter θ_{maml} and the adaptive meta-parameter θ_{meta} of this paper are compared in the convergence speed of retraining with few samples on VOC 2007+VOC 2012 datasets, respectively. In this paper, only one type of target is located in the target location part. As shown in Figure 9, the Loss value of the model with random parameter θ_{random} is 2.8583 in the initial training, and converges to 0.2 after 2000 training times. Using the meta-parameter θ_{maml} based on MAML-RL, the Loss value of the initial training of the model is 2.0525, indicating that the model has a certain learning ability, and converges to around 0.2 after 1000 training times. For the method in this paper, the model uses adaptive meta-parameters to have a low Loss value (1.3347) for few sample data in the initial case, indicating that the model automatically matches meta-parameters adapted to the new task as initial parameters, and the model reaches convergence in a few steps (500). In addition, with the same number of training steps, the few-shot task overall fits more than r and m through retraining. It shows that the proposed method has good generalization and learning ability for few-shot data training.

C. Results of target positioning

In this paper, six categories cat, bicycle, aeroplane, cow, tvmonitor and DOGin VOC 2007+VOC 2012 dataset are selected for testing with the meta-parameters based on the proposed method as initial parameters. By testing the test samples after the same training batch, it is found that the proposed method can make the target localization model have better recognition accuracy for the target in a few steps, and part of the samples can capture the target position in one action step. Figure 10 shows the test results of the test sample, where Iteration represents the number of steps explored by the model, and the blue wireframe represents the prediction of the model on the target field of view.

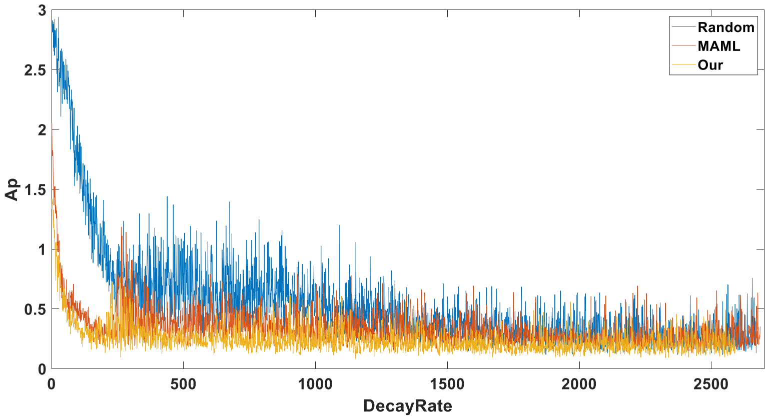
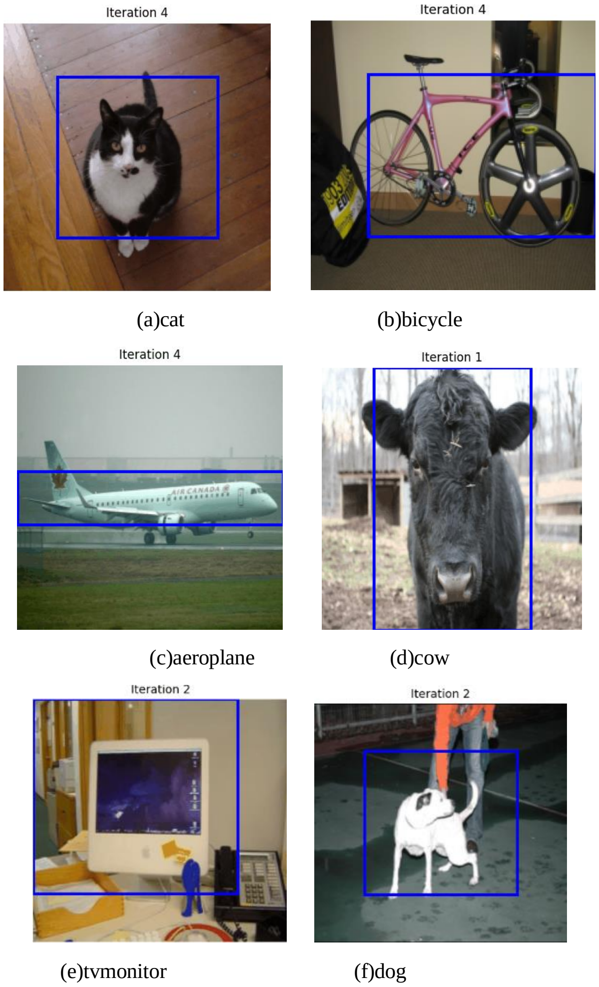


Figure 9. Comparison of training loss functions



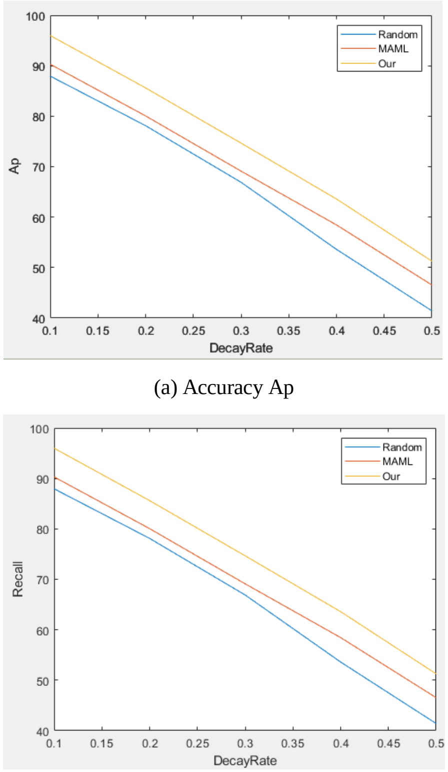
(a)cat (b)bicycle

(c)aeroplane (d)cow

(e)tvmonitor (f)dog

Figure 10. Results of ta

Figure 11 shows the precision rate (Ap) and Recall rate (Recall) of the model under the test data set and the proposed method under the VOC 2007 test set for the new tasks bird, motorbike, diningtable and train categories. The initial discount rate of the model is set to 0.5 and decreases to 0.1 in steps of 0.1, which shows the probability that the agent chooses the optimal action according to the model. It can be seen from Figure a and Figure b that the accuracy and recall of the model decrease when the learning rate increases from 0.1 to 0.5, and the accuracy and recall of the model for different tasks reach the highest when the learning rate is 0.1.



(a) Accuracy Ap

(b) Recall Rate

Figure 11. Comparison of precision and recall

VI. Conclusions

In order to overcome the shortcomings of low generalization ability caused by insufficient data in few-sample data sets and forgetting of meta-reinforcement learning, this paper proposes a meta-reinforcement learning target localization algorithm based on meta-reinforcement learning parameter playback. Firstly, the model uses MAML method to train various tasks to obtain local optimal parameters. Then, a meta-parameter pooling method is used to store and playback the task meta-parameters, and the optimal parameters for few-sample data training are retrained by feature matching to improve the generalization ability, training speed and target positioning accuracy of the model.

In the model training phase, the positioning model and meta-parameter pool are trained in stages to improve the positioning accuracy of the model, and the data utilization efficiency is improved by sharing the data in the meta-parameter pool. The experimental results show that the number of samples required for model training and the computational cost are effectively reduced by using the memory and replay method of the meta-parameter pool. The experiments on the target positioning data set verify the effectiveness and generalization of the method for practical problems. However, the test results in the process of multi-type object detection are not ideal, and there is a large room for improvement. For multi-target detection, target detection is realized by using yuan reinforcement learning way and the transfer between multiple targets is in-depth study in this part.

References

[1] Mathe S, Pirinen A, Sminchisescu C. Reinforcement learning for visual object detection[C]//Proceedings of the IEEE conference on computer vision and pattern recognition. 2016: 2894-2902.

[2] Zhou W, Lai J, Liao Y, et al. Meta-reinforcement learning based few-shot speech reconstruction for non-intrusive speech quality assessment [J]. Applied Intelligence, 2023, 53(11):14146-14161.

[3] Yao Hongge, Zhang Wei, Yang Haoqi et al. Joint return target depth of intensive study [J]. Journal of automation, 2023, 49 (5):1089-1098. The DOI: 10.16383 / j.a as c200045.

[4] Snell J, Swersky K, Zemel R. Prototypical networks for few-shot learning [J]. Advances in neural information processing systems, 2017, 30.

[5] Finn C, Abbeel P, Levine S. Model-agnostic meta-learning for fast adaptation of deep networks [C]//International conference on machine learning. PMLR, 2017:1126-1135.

[6] Gupta A, Mendonca R, Liu Y X, et al. Meta-reinforcement learning of structured exploration strategies [J]. Advances in neural information processing systems, 2018, 31.

[7] Thrun S, Pratt L. Learning to learn: Introduction and overview [M]//Learning to learn. Boston, MA: Springer US, 1998:3-17.

[8] Ajay, Anurag, et al. "Distributionally adaptive meta reinforcement learning." Advances in Neural Information Processing Systems 35 (2022):25856-25869.

[9] Duan Y, Schulman J, Chen X, et al. Rl $^ 2$: Fast reinforcement learning via slow reinforcement learning [J]. arXiv preprint arXiv:1611.02779, 2016.

[10] Al-Shedivat M, Bansal T, Burda Y, et al. Continuous adaptation via meta-learning in nonstationary and competitive environments [J]. arXiv preprint arXiv:1710.03641, 2017.

[11] Fakoor R, Chaudhari P, Soatto S, et al. Meta-q-learning [J]. arXiv preprint arXiv:1910.00125, 2019.

[12] Wang Y, Yao Q, Kwok J T, et al. Generalizing from a few examples: A survey on few-shot learning [J]. ACM computing surveys (csur), 2020, 53(3):1-34.

[13] Schoettler G, Nair A, Ojea J A, et al. Meta-reinforcement learning for robotic industrial insertion tasks [C]//2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). IEEE, 2020: 9728-9735.

[14] Garcia F, Thomas P S. A meta-MDP approach to exploration for lifelong reinforcement learning [J]. Advances in Neural Information Processing Systems, 2019, 32.

[15] Sutton R S, Barto A G. Reinforcement learning: An introduction [M]. MIT press, 2018.

[16] Yu T, Quillen D, He Z, et al. Meta-world: A benchmark and evaluation for multi-task and meta reinforcement learning [C]//Conference on robot learning. PMLR, 2020:1094-1100.

GreatFree as a Generic Distributed Programming Language and the Foundation of the Cloud-Side Operating System

Bing Li
GreatFree Research Lab
Jiangsu, China
E-mail: bing.li@asu.edu

***Abstract*—GreatFree is a generic distributed programming language to develop various distributed systems over the Internet-oriented computing environment. The fundamental characters of GreatFree are shaped by three essential techniques, including the message-passing, the physical-machine-visible, and the thread-visible. More important, GreatFree is equipped with three additional distinguished mechanisms, i.e., the distributed primitives, the distributed common patterns, and the distributed threads on the application level, which are sufficient to turn GreatFree into a generic distributed programming technology. To the best of our knowledge, compared with any others, GreatFree is the first one to achieve the goal. Thereafter, GreatFree is capable of exploiting distributed computing resources flexibly to adapt to any heterogeneous environments with a uniform solution. It indicates that GreatFree represents the common principles existed in various complicated computing circumstances over the Internet. That inspires that GreatFree is a proper technology to build a new concept of cloud computing environment, i.e., the cloud-side operating system, which dominates diverse distributed computing resources upon the common principles of GreatFree. Such a system is a generic development and running environment for any distributed systems. Without doubt, within the environment, GreatFree is the unique choice to program any distributed systems in a scalable manner.**

Keywords- Cloud-side Operating System; Generic Distributed Programming Languag; Distributed Primitives; Application-level Threading on Messaging; the Distributed Common Patterns

I. INTRODUCTION

GreatFree is a generic distributed programming language for the Internet-oriented computing environment. It has the three necessary characters to become such a technique, i.e., the message-passing between threads, the threading programmable, and the physical distributed node programmable. More important, three additional distinguished characters are sufficient to support GreatFree to become a generic and rapid distributed programming paradigm. Those characters consist of the distributed primitives, the distributed common patterns, and the application-level threading on messaging. With the emergence of GreatFree, it inspires the generation of the new concept of distributed development and running environment, i.e., the cloud-side operating system.

GreatFree is a generic programming paradigm for the Internet-oriented computing environment. Three technologies, i.e., the Distributed Primitives (DP), the Application-level Threading on Messaging (ATM), and the Distributed Common Patterns (DCP), are proposed to form the distinguished characters of GreatFree. The DP consists of a series of the distributed primitive application programming interfaces. Any distributed components and systems are originated from the DP through self-derivation without the support of any third-party distributed techniques. The ATM is the distributed, application-level, and asynchronous message-passing threading, which aims to implement the most fine-grained distributed concurrent systems to leverage computing resources in various distributed environments conveniently. The DCP unveils that the code of any heterogeneous distributed systems is constructed with a limited number of the common code-level design patterns through self-derivation. With the support of those techniques,

DOI: 10.2478/ijanmc-2023-0078

GreatFree not only becomes a generic and rapid paradigm of distributed programming but also establishes the basis of the first distributed programming language for the Internet-oriented computing environment.

The DP represents the most fundamental programming components for the development of various distributed systems. As one of the foundations of GreatFree, the DP is the basis of general-purpose techniques for distributed systems development. It shapes GreatFree to become a full coverage and rapid development technique in a sense that it not only hides tedious details but also exposes indispensable elements. The DP is equivalent to the most basic and mandatory distributed computing resources and mechanisms to construct any distributed systems. If any element of the DP is missed, it definitely results in the failure of programming with GreatFree to implement any systems. On the other hand, if any of the ones encapsulated by the DP is exposed, it raises the programming efforts and lowers the quality of developed systems obviously.

Different from other concurrent mechanisms [1~15] for distributed programming, the ATM threads are visible to developers, i.e., the threads can be manipulated by passing application-dependent messages without worrying any native characters of threads. Because of the complexity of the Internet-oriented computing environment, it is impossible to predefine an omni-potent concurrency instrument, which not only conceals every detail of threads for rapid programming but also adapts to sophisticated cases. Hence, it is required to allow developers themselves to create, monitor, reuse, and collect threads directly on the application level. Thus, the support of visible threads is necessary for a generic distributed programming paradigm. In addition, to isolate developers from the tedious synchronization workload of threading, the ATM is founded on the basis of the message-passing rather than that of the memory-sharing in traditional methodologies [16~22]. Using the ATM, developers are able to handle any number of remote threads on arbitrary distributed nodes to process intricate distributed tasks concurrently through message-passing only. Consequently, a single ATM thread is equivalent to a distributed node accomplishing scheduled tasks in a serial way such that developers can program with the ATM threads in the same way as distributed nodes to construct various complicated distributed systems in a higher quality. As a novel distributed concurrency technique, the ATM is another crucial foundation of GreatFree to become a generic distributed programming paradigm.

The DCP is a phenomenon existing natively in any distributed system instead of a contrived technique. It is evident that the DCP exists pervasively in all the code of distributed programs. With respect to the large amount of distributed programming experiences in various environments, it discovers that the code of heterogeneous distributed systems abides by a limited number of the homogeneous code-level design patterns. Although it is always necessary to derive diverse DAAs to raise the rapidness of programming complicated distributed systems, the types of code-level design patterns do not change with the new proposed APIs. Only the DCP is sufficient to adapt to any scenarios since the patterns for DAA are always the straightforward aggregations of the DCP and nothing else needs to be invented for DAA. Thus, when programming with DAA, the same code structures originated from the DCP are reusable for distinct distributed systems. The phenomenon reveals the truth that the code structures of a distributed system are independent of its distributed natures.

II. RELATED WORK

Distributed programming is an evergreen topic such that it contains plentiful solutions. According to their originally target computing environments by default, all of them are classified as the Sequential and Standalone Paradigm (SSP), the Distributed Frameworks Paradigm (DFP), and the Distributed Programming Paradigm (DPP). The DPP, the primary methodology GreatFree competes with, contains a variety of mutations aiming to become general-purpose solutions.

A. The Sequential and Standalone Paradigm

The SSP specifies the programming methodologies that implement a system running in the sequential and standalone manner by default. Most traditional high-level programming languages [23~34] belong to the category. When the SSP was invented, the primary effort was focused on replacing the machine-dependent code with the nature-language-like syntax and semantics. They do not take into account the issues of the concurrency and distribution. With the rapid development of computing technologies, it is required to program concurrent and distributed systems using those sequential and standalone languages. For that, the techniques of threading and networking are proposed to support the SSP programs to be executed concurrently in a network environment. When programming with those techniques, developers are required to transform the sequential and standalone instructions to the concurrent and distributed ones with those attached techniques. The procedure is notoriously difficult such that even proficient developers usually avoid doing that if alternative solutions are available.

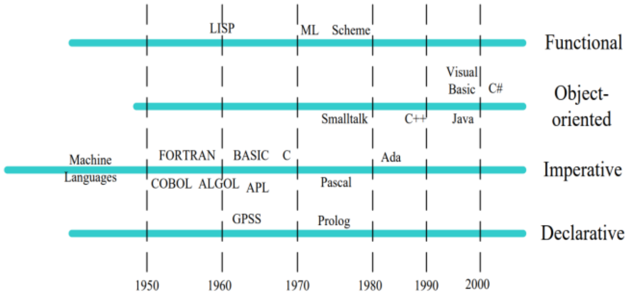


Figure 1. Sequential and Standalone Paradigm

Simply put, to program a server, i.e., one single physical distributed node, with the SSP, the effort is intolerable although such a server is the simplest component within various distributed systems. The programming efforts for the server include networking, serialization, message-passing, message scheduling, threading management, threading synchronization, threading scheduling, resource management and so forth. Additionally, the solutions to those issues always change for specific applications, such as lightweight or heavyweight, streaming or messaging, idle or busy, centralized or decentralized, stable or unstable, heterogeneous or homogeneous, machined or socialized, and so forth. Even though all the issues are resolved, it is still tough to extend them to implement a scalable large-scale system since the workload is always raised exponentially. Besides, the incompatible code structures also reflect the difficulty of programming distributed systems with the traditional languages. Even the same developer programs the same server with different code patterns if lacking for experiences and references. It brings forth the difficulty to manage, reuse and debug for further development and collaboration.

B. The Distributed Frameworks Paradigm

Since it is tough to implement distributed systems with the SSP, as the semi-constructed systems, the distributed frameworks are employed to simplify the development in most cases. Because the DFP resolves all the distributed issues and make them invisible in one specific domain, developers are able to work within a virtualized computing environment in which no concurrent and distributed issues need to be considered. Then, they are concentrated on programming upper level applications in a sequential and standalone manner. This approach is the most rapid such that it becomes popular nowadays.

However, the DFP is never a generic solution for the complexity of distributed computing environments. Instead, all the DFP solutions [35~60] are application-specific such that it hides developers from all the distributed issues for one particular scenario in the enterprise-level distributed computing environment. The current existing distributed frameworks cover the issues such as the distributed objects environment, the remote procedure call, the map/reduce concurrency, the clustering, the infrastructure for the enterprise environment, the data management, the streaming, the high-level scripting, and the customized applications. Unfortunately, it is impossible to establish a framework to make all the distributed issues transparent in the Internet-oriented computing environment. If one particular application is suitable to one of those distributed frameworks luckily, it results in low development efforts. If not, there is no way to make changes on those frameworks to adapt to specific requirements. A common case is that a bunch of distributed frameworks have to be accumulated in one specific application to fulfill respective scenarios. Such a system is always cumbersome in terms of management and resources consuming. Therefore, the DFP is far from perfect since the software development is degenerated from straightforward programming with a single full-fledged language to patching, scripting, configuring, or integrating multiple heterogeneous frameworks.

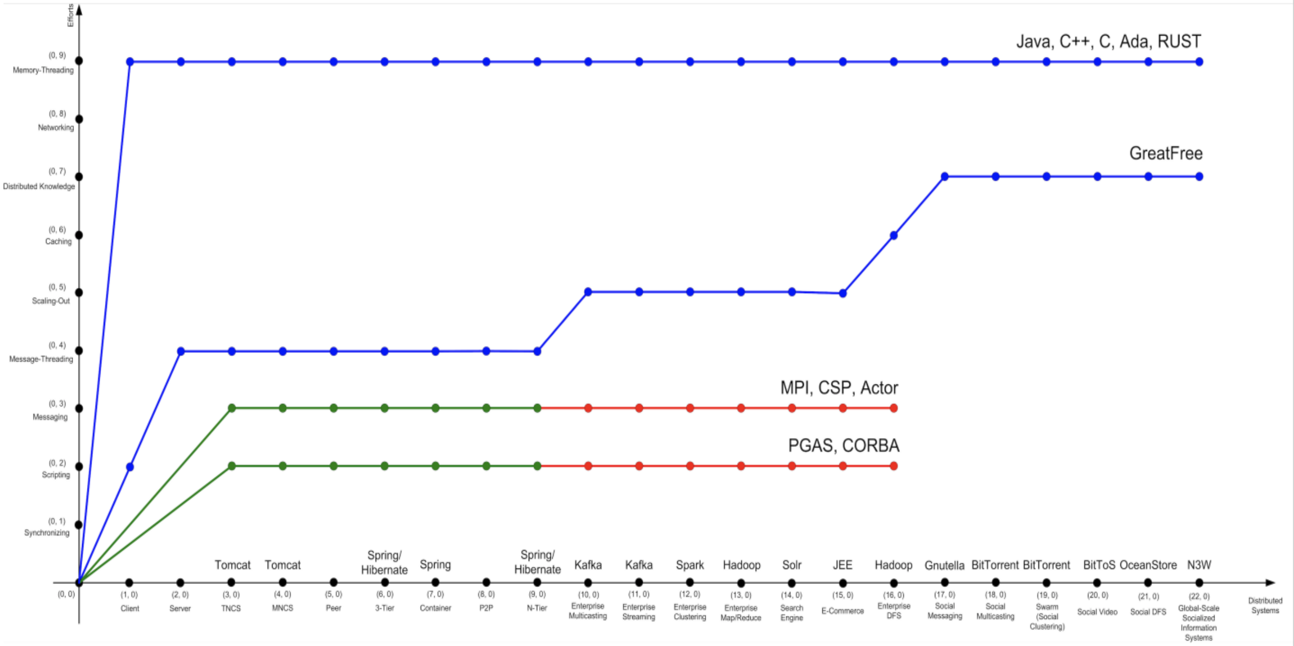


Figure 2. Distributed Frameworks Paradigm

C. The Distributed Programming Paradigm

The Distributed Programming Paradigm (DPP) [61~82] is defined as the methodology that aims to develop the systems in the computing environment in which multiple computers are connected through networking.

For the complexity of distributed computing environments, there are numerous mutations in the DPP. As any computing systems perform behaviors to manipulate data, it is appropriate to identify them upon their approaches of accessing and exchanging distributed data among distributed threads and processes. According to that, the DPP is categorized into the Memory-Sharing Paradigm (MSP) and the Message-Passing Paradigm (MPP). The MSP attempts to create a virtualized uniform memory space for a distributed computing environment. Hence, network locations are invisible, and data is retained in a unique memory space from a programmer's point of view. Because of that, a distributed computing environment is transformed to a standalone one. With the support of the MSP, it is unnecessary to take care of any distributed techniques to implement distributed systems. On the other hand, the MPP believes it is feasible for simple scenarios to construct such a homogeneous memory space. For complicated cases, it is impossible. Even though for those simple ones, it causes additional problems, such as heavy synchronization, low performance, and low scalability. Therefore, the MPP claims that multiple independent memory spaces are the foundation to process distributed data. To establish asynchronous, high performance, and scalable distributed systems, instead of sharing, data is passed as messages among distributed entities, including threads and processes, within isolated memory spaces.

In addition, as one of the most important components to program distributed systems, the concurrency implementation is another proper indicator to differentiate various paradigms. In accordance with the visibility of threading, the DPP is divided into the Threading Invisible Paradigm (TIP) and the Threading Visible Paradigm (TVP). Because of the difficulty to program with traditional threading, all the existing paradigms encounter the dilemma, i.e., they have to make a single choice between the adaptability to various scenarios and the rapidness of programming. Each of them either loses the adaptability to gain the rapidness or abandons the rapidness to obtain the adaptability. None of them wins both of them. The TIP hides threading to lower the difficulty to concurrency programming for distributed systems. That is the choice of most paradigms such that it proves the toughness of threading further. To do that, a concurrency pooling mechanism needs to be predefined to manage threads running asynchronously. Different from the TIP, the TVP exposes threading to adapt to various scenarios since it is impossible to create an omnipotent thread management mechanism to deal with unpredictable cases. For complicated systems, it is required for programmers to design the specific pooling for threading based on the certain domain knowledge. Therefore, the TVP declares that visible threading is a mandatory condition to accommodate to various contexts.

Finally, any distributed systems are constructed upon multiple computing devices. Thus, it is necessary to convert such a standalone device to a distributed node, which is able to interact with others. The technique of conversion is called the distributed modeling. To speed up the development of distributed systems, many variants of the DPP hide the modeling from programmers. They intend to create an abstract object to replace the physical heterogeneous distributed node. Programming with the logical entities instead of physical nodes, the tedious details of distributed environments are filtered out such that the efforts are focused on composing those homogeneous components. Then, the programming rapidness is raised obviously. Such a paradigm is called the Modeling Invisible Paradigm (MIP). However, because of the complexity of distributed computing environments, hiding physical distributed nodes results in the fact that programmers lose the possibility to access locations of distributed nodes, organize distributed nodes into one particular topology, and establish effective interactions among distributed nodes. Those issues are critical for a complicated distributed system and it is impossible to predefine them without taking into account requirements in one specific circumstance founded on those distributed nodes. For that, another approach, the Modeling Visible Paradigm (MVP) is proposed to overcome the drawbacks of the MIP.

On the other hand, all the paradigms can be classified roughly into the high-level one and the low-level one as well. The high-level paradigm strives to construct a logical prototype that is as independent of the physical distributed computing environments as possible to raise the rapidness of distributed programming. On the contrary, the low-level one aims to be closed to the physical distributed computing environments such that it is possible to guarantee the generality of distributed programming. With respect to the principle, the high-level one consists of the MSP, the TIP, and the MIP whereas the low-level one includes the MPP, the TVP, and the MVP. Moreover, among those approaches, data accessing determines others to a large extent since the functions of a computing system can be summarized as reading or writing data. For that, the DPP is mainly classified as the MSP and the MPP, which are the high-level and the low-level respectively. It is unreasonable to introduce the TVP and the MVP, which are low-level techniques compared with the TIP and the MIP, into the MSP since the paradigm conceals all the distributed details. Different from the MSP, as a low-level paradigm, the MPP is open enough to play the role of technical basis such that high-level ones are allowed to be established on it and low-level ones are employed to raise its generality.

TABLE I. THE CATEGORIZATIONS OF THE PPP INSTANCES

ID	Technique	MSP	MPP	TIP	TVP	Year of Birth
1	Id	Y	N	Y	N	1975
2	Sisal	Y	N	N	Y	1983
3	Occam	N	Y	N	Y	1983
4	Multilisp	Y	N	Y	N	1985
5	Newsqueak	N	Y	N	Y	1985
6	ParLog	Y	N	Y	N	1987
7	C*	Y	N	Y	N	1987
8	Joyce	N	Y	N	Y	1987
9	SequenceL	Y	N	Y	N	1989
10	Charm++	N	Y	Y	N	1989
11	Lustre	Y	N	Y	N	1991
12	HPF	Y	N	Y	N	1991
13	Alef	N	Y	N	Y	1992
14	ZPL	Y	N	Y	N	1993
15	SuperPascal	N	Y	N	Y	1993
16	OpenMP	Y	N	Y	N	1997
17	Titanium	Y	N	Y	N	1998
18	UPC	Y	N	Y	N	1999
19	BMDFM	Y	N	Y	N	2002
20	CnC	Y	N	Y	N	2004
21	XC	N	Y	N	Y	2005
22	Fortress	Y	N	N	Y	2006
23	Sequoia++	Y	N	Y	N	2006
24	Preesm	N	Y	Y	N	2008
25	Chapel	Y	N	Y	N	2009
26	C++AMP	Y	N	Y	N	2011

In addition to those classic ones, as one subset of the DPP, the Parallel Programming Paradigm (PPP) can be regarded as an early version to a special distributed computing environment. The PPP provides an abstract prototype for concurrent executions to attain high performance on a single standalone physical computer equipped with multiprocessors. Similar to the DPP, the PPP consists of the MSP, the MPP, the TIP, and the TVP as well. Neither MIP nor the MVP is associated with the PPP because the PPP supports the standalone computing device only. Compared with others of the DPP, the computing environment of the PPP is highly homogeneous since those multiprocessors within a computer are identical and each of them has an equivalent assignment of computer resources and capabilities such that there are no differences among those processors when exploiting them to accomplish multiple tasks concurrently. Therefore, it is easy to design highly abstract programming components to conceal low-level details. Because of that, most variants of the PPP are classified as the MSP and the TIP.

TABLE II. THE SUMMARY OF THE PPP

Paradigm	Proportion				
	26(100%)	1970s	1980s	1990s	2000s
MSP	18(69%)	1(4%)	5(19%)	6(23%)	6(23%)
MPP	8(31%)	0(0%)	4(15%)	2(8%)	2(8%)
TIP	18(69%)	1(4%)	5(19%)	6(23%)	6(23%)
TVP	8(31%)	0(0%)	4(15%)	2(8%)	2(8%)
MSP & TIP	16(62%)	1(4%)	4(15%)	6(23%)	5(19%)
MSP & TVP	2(8%)	0(0%)	1(4%)	0(0%)	1(4%)
MPP & TIP	2(8%)	0(0%)	1(4%)	0(0%)	1(4%)
MPP & TVP	6(23%)	0(0%)	3(12%)	2(8%)	1(4%)

According to the above discussions, as a typical methodology of the DPP, GreatFree is categorized into the MPP, the TVP, and the MVP. Aiming to be a generic paradigm, for the issues of data accessing, threading, and modeling, GreatFree always chooses the low-level solution rather than the high-level one. In other words, GreatFree has to propose distinct resolutions to avoid the inefficiency of programming. To break out the dilemma, GreatFree possesses the two distinguished characters, including the DP as the primitive distributed programming components and the DCP as the common homogeneous distributed code structures, to programming distributed systems rapidly. Moreover, it puts forward the distinct solution, the ATM, to the tough issue of threading. Therefore, GreatFree not only simplifies distributed programming mechanisms in the way to conceal those intricate techniques but also abstracts distributed resources and technical details to a degree to sustain the sufficient adaptability to various distributed computing environments.

III. GREATFREE AS A GENERIC DISTRIBUTED PROGRAMMING LANGUAGE

GreatFree is a generic distributed programming paradigm in the Internet-oriented computing environment. The three fundamental techniques, i.e., the Distributed Primitives (DP), the Application-level Threading on Messaging (ATM), and the Distributed Common Patterns (DCP), are proposed to achieve the goal to be a generic paradigm in the highly heterogeneous distributed computing circumstance.

The DP is the most fundamental elements that are sufficient and necessary to program any distributed systems. The ATM is a distributed concurrency programming technique distinguished from others by the mechanisms of the message-passing and the visible-threading on the application level. The DCP is a rapid distributed programming solution on condition that any heterogeneous distributed systems can be constructed with the common homogeneous code-level design patterns.

In general, GreatFree is the paradigm of the MPP, the TVP, and the MVP in the domain of distributed programming. Moreover, GreatFree is distinct from any others in the discoveries of the rudimentary and universal programming components, the application-level fine-grained concurrency model, and the homogeneous code structures. Thus, GreatFree becomes a new paradigm in the fashion of being individual-respected, messaging-oriented, threading-visible, and self-derivable such that it becomes unique as a generic distributed programming methodology in the heterogeneous computing environment of the Internet.

TABLE III. THE CATEGORIZATIONS OF THE MSP AND MPP INSTANCES

ID	Technique	MSP	MPP	TIP	TVP	Year of Birth
1	CSP	N	Y	N	Y	1978
2	Ada	Y	N	Y	N	1980
3	Emerald	Y	N	Y	N	1985
4	Linda	Y	N	Y	N	1986
5	Erlang	N	Y	Y	N	1986
6	LabVIEW	N	Y	Y	N	1986
7	Hermes	N	Y	N	Y	1986
8	SR	Y	N	Y	N	1988
9	Concurrent Smalltalk-90	N	Y	Y	N	1989
10	Haskell	Y	N	Y	N	1990
11	Janus	N	Y	Y	N	1990
12	CORBA	Y	N	Y	N	1991
13	MPI	N	Y	Y	N	1991
14	Oz	N	Y	N	Y	1991
15	SHMEM	Y	N	Y	N	1993
16	CML	N	Y	N	Y	1993
17	Glenda	Y	N	Y	N	1994
18	Limbo	N	Y	Y	N	1995
19	Millepede	Y	N	Y	N	1996
20	Joule	N	Y	Y	N	1996
21	E	Y	N	Y	N	1997
22	MPJ	N	Y	N	Y	1999
23	MPD	Y	N	Y	N	2000
24	SALSA	N	Y	Y	N	2001
25	CAL	N	Y	Y	N	2001
26	D	N	Y	N	Y	2001
27	X10	Y	N	Y	N	2004
28	JoCaml	N	Y	N	Y	2004
29	JCSP	N	Y	N	Y	2005
30	PyCSP	N	Y	N	Y	2006
31	Akka	N	Y	Y	N	2009
32	Go	N	Y	Y	N	2009
33	Axum	N	Y	Y	N	2009
34	Bloom	Y	N	Y	N	2010
35	Rust	N	Y	N	Y	2010
36	Ateji PX	N	Y	Y	N	2010
37	Elixir	N	Y	Y	N	2011
38	Julia	N	Y	N	Y	2012
39	Akka.NET	N	Y	Y	N	2013

TABLE IV. THE SUMMARY OF THE MSP AND THE MPP

Paradigm	Proportion				
	39(100%)	1970s	1980s	1990s	2000s
MSP	13(33%)	0(0%)	4(10%)	6(15%)	3(8%)
MPP	26(67%)	1(3%)	4(10%)	7(18%)	14(36%)
TIP	28(72%)	0(0%)	7(18%)	26(23%)	11(28%)
TVP	11(28%)	1(3%)	1(3%)	3(8%)	6(15%)
MSP & TIP	13(33%)	0(0%)	4(10%)	6(15%)	3(8%)
MSP & TVP	0(0%)	0(0%)	0(0%)	0(0%)	0(0%)
MPP & TIP	15(39%)	0(0%)	4(10%)	3(8%)	8(21%)
MPP & TVP	11(28%)	1(3%)	1(3%)	3(8%)	6(15%)

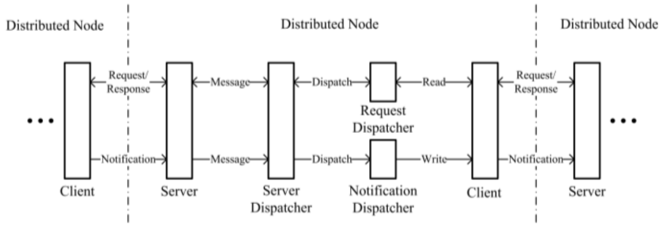


Figure 3. GreatFree Paradigm - DP

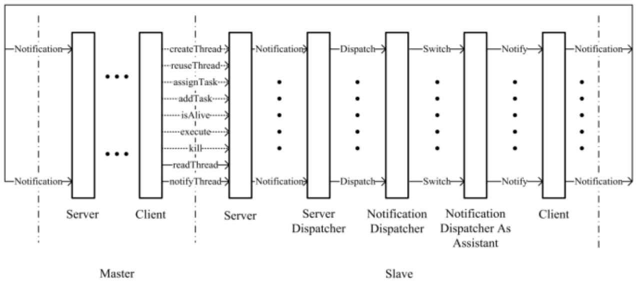


Figure 4. GreatFree Paradigm - AMTL

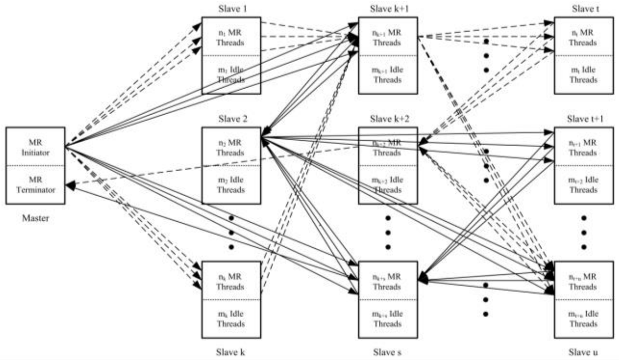


Figure 5. GreatFree Paradigm – AMTL for Map/Reduce

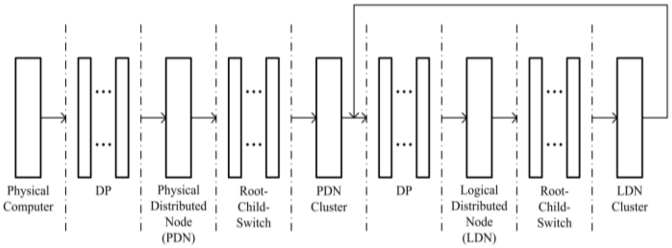


Figure 6. GreatFree Paradigm - SPRA

A. The DP

The DP represents the most fundamental distributed elements, which are sufficient and necessary to program any distributed systems in the Internet-oriented computing environment. The DP is the foundation of GreatFree to be a generic programming paradigm. The DP is made up of a series of the distributed primitive APIs. Programming with the DP only, it is rapid to create various distributed systems, such as the simplest ones, the distributed advanced APIs, and the distributed frameworks, in any environments. Even the most complicated one, the global scale socialized heterogeneous information system over the Internet, can be programmed with the DP. It demonstrates that the DP is also the basis of GreatFree as the self-derivable programming paradigm.

1) The Distributed APIs

The DP is the most elementary application programming interfaces to accomplish the intrinsic distributed functionalities. It consists of the three subsets, i.e., the distributed modeling, the distributed messaging, and the distributed dispatching. The distributed modeling transforms a single standalone physical computing device to one physical distributed node, i.e., one physical client or one physical server, such that the device can interact with any others within the Internet environment. The distributed messaging describes the interactions of requesting or eventing via the messages in the plain object-oriented form transmitted over the Internet. The distributed dispatching processes incoming messages concurrently in a scaling-up manner on a distributed node. As the most complicated component in the DP, the distributed dispatching includes the message dispatcher, the messaging thread pools, and the messaging threads. After a long-term experimenting, all of them are polished carefully to keep the balance between encapsulating underlying tedious technical details to lower programming efforts and exposing indispensable distributed components. The DP enables GreatFree to be adaptable enough to various distributed computing environments.

2) Programming the Simplest Systems

Programming with the DP directly, it is sufficient and necessary to build the most rudimentary distributed system, the Two-Node Client/Server (TNCS) one, which contains two physical distributed nodes and conforms to the interaction principle of the client/server model upon lightweight messaging. Furthermore, it is straightforward to increment the scale of the clients up to the capacity of the server. Then, a more complicated system, which contains multiple clients and a single server, is created. As the counterpart of the TNCS, it is called the Multiple-Node Client/Server (MNCS) system. It proves that the generality of GreatFree because of the axiom that any distributed systems are the aggregation of the TNCS or the MNCS.

3) Programming Distributed Advanced APIs

Based on the generality of GreatFree, besides programming the simplest distributed systems, another primary goal of the DP is used to program the Distributed Advanced APIs (DAA). Although the primitive APIs are generic, it is still expected to create powerful APIs, i.e., the DAA, to raise the programming productivity through further encapsulation. Any DAA is the direct or indirect encapsulation of the DP using the object-oriented technique. Additionally, the procedure is recursive, i.e., any new DAA is built upon programming with the DP or the existing DAA recursively.

As a general-purpose distributed programming paradigm, besides the DP, GreatFree provides additional two categories of DAA, including the distributed clustering and the distributed caching. The distributed clustering is the important programming component to construct scalable distributed systems. The distributed caching provides a large-scale high-performance storage mechanism in the interfaces of common data structures, such as the map, the list, the stack, the queue and so on. Similarly, all of them are derived through programming with the DP and existing DAA recursively.

4) Programming Distributed Frameworks

The motivation to create the DAA aims to program more complicated distributed frameworks rapidly through keeping on hiding low-level details that are unnecessary for one particular distributed context. Once if the DAA is available, it is more efficient to build sophisticated ones.

GreatFree is not an application-level programming paradigm such that it is focused on the establishment of distributed frameworks, which emphasize the semi-constructed distributed systems and ignore the implementation of upper-level applications. Working on a mature framework is currently the primary approach to develop distributed applications. For the complexity of the Internet-oriented computing environment, distributed frameworks have numerous mutations, such as the Peer-to-Peer (P2P), the 3-Tier, the n-Tier, the Map/Reduce, the streaming, the storage, the enterprise cluster, the cloud, and so forth. GreatFree-based frameworks have one more advantage. Different from the dedicated ones that can hardly be revised, the frameworks of GreatFree can be programmed further conveniently to accommodate to specific requirements on functions as well as performance with the support of the DP and the existing DAA. Over those frameworks, it is easy to establish a great many distributed applications such as chatting, file transmissions, e-commerce systems, gaming, financing, block-chains, and so forth to fulfill various circumstances.

In addition to the common frameworks, GreatFree can be used to program some important distributed frameworks for specific applications, such as the enterprise container, the search engine, the video streaming, distributed file systems, and the distributed data centers. Similarly, those frameworks are programmable further rather than the fixed or configurable ones only. The most complicated distributed systems are the ones dominated by human capital as well as social capital. Such systems emerge with the progresses of the Internet. One example is the World Wide Web (WWW), which is the global scale socialized heterogeneous information system over the Internet. Such a system conforms to the principles of human interactions in addition to those of machines. Thus, the system is highly heterogeneous with potentially infinite users and tremendously high workload. It is impossible to employ any existing techniques to implement it conveniently. Fortunately, because of the natures of the DP, it has already been utilized successfully in the project of the New World Wide Web (N3W). As a highly heterogeneous system, the N3W is one upgraded instance of the global scale socialized distributed system to resolve the drawbacks of the traditional WWW.

B. The ATM

The ATM is a novel concurrency mechanism for distributed programming. There is no way to establish a generic programming paradigm without properly designed threading. To achieve the goal, the ATM is distinct from others in its unique characters, including the visibility, the application-level, the distribution ability, the messaging orientation, and the programmability.

1) The Visible Threading

Threads are the major resource for any distributed programming paradigms since distributed systems are concurrent in nature. It becomes infeasible to conceal or degenerate threads for rapid programming when distributed computing environments become complicated. The characters of the Internet-oriented distributed computing environments result in utilizing resources concurrently in the most fine-grained granularity. Thus, it is required to control threads directly to implement high-quality concurrent algorithms rather than any other management mechanisms. The primary operations on threads include creating, task-assigning, interacting, monitoring, reusing, collecting, and so forth. Only if those functions are available to programmers, it is possible to program sufficiently fine-grained distributed concurrent algorithms to accommodate to the heterogeneity of various distributed circumstances. Following the principle, the ATM provides programmers with the full governance in terms of controlling a single thread in its entire lifecycle.

2) The Application-Level Threading

The application-level threading is defined as a concurrent programming mechanism that provides developers with the independently running threads, which abide by application-level instructions to change their behaviors rather than any system-level commands isolated from upper level scenarios. The application-level threading of the ATM alleviates the difficulty of programming with the system-level threads directly. Through the approach, the ATM threads are programmed via the simplified directives dependent on application progressing statuses rather than taking care of the raw characters of threads. From a programmer’s point of view, an instance of the ATM threads is dominated to accomplish various tasks for an application by messages of requesting or eventing until it is overloaded. In accordance with the dynamics of a specific application, programmers are offered the privilege to monitor their current states, evaluate the workload to be assigned to them, and even consider specific scenarios to administrate the thread reasonably. Luckily, all the efforts are focused on the concurrent strategies as well as the distributed solutions on the upper level instead of the management of the native threading on the lower level. In brief, the ATM is totally different from the system-level approaches that are independent of application scenarios in the SSP.

3) The Distributed Threading

As a large-scale distributed concurrent programming mechanism, the ATM is usually sustained by a scalable distributed cluster made up with multiple slave nodes, which are the ATM thread providers responsible for supplying sufficient ATM threads to fulfill one particular concurrent task. The cluster is accessed by any number of masters, who play the role of an ATM thread consumer. The count of the slaves depends on the computing requirements of specific distributed scenarios such that the scale of the cluster can be enlarged arbitrarily upon workload. To assign concurrent tasks, the master distributes its requirements via asynchronous messaging to the cluster such that the ATM threads originated from the slaves are created, reused, and composed together to accomplish all the subtasks. During the procedure to work on the subtasks, those ATM threads are still able to interact with each other following the commands from the master to deal with additional missions if needed. After one particular task is finished, the final result is gathered from all the ATM threads on the slaves to the master. In brief, rather than a naked thread running asynchronously on a physical standalone computing device in the SSP, an ATM thread is equivalent to a logical distributed node executing scheduled tasks independently in a serial fashion such that it can be exploited with others using various distributed strategies.

4) The Threading on Messaging

The ATM adopts the popular approach, asynchronous messaging, of the MPP to build loosely coupled distributed systems in the heterogeneous distributed computing environments. To guarantee the adaptability, the TVP is another character of the ATM. However, different from other TVP paradigms, with which threads can hardly be manipulated arbitrarily, the ATM is an approach that allows programmers to dominate threads fully on the application level. On the other hand, when programming with the ATM, the visible threading is absolutely not identical to that of the SSP, in which threads are naked for programming such that programmers are required to worry about each detail of threading on the system level. Rather, the ATM is in essence a concurrent mechanism that converts the system-level memory-sharing threading on a physically standalone computer to the application-level message-passing threading over a large-scale distributed computing environment. Programming with the ATM, developers are allowed to create, monitor, reuse and collect the threads from distributed nodes through asynchronous messaging. The messages contain the application-dependent instructions, tasks and states from a thread consumer to thread providers rather than any system-level directives that probably disrupt the upper-level distributed activities.

5) The Programmable Threading

The same as other complicated systems, the ATM is a distributed concurrent programming mechanism that is constructed completely through programming with the DP and the relevant DAA as well. That is another evidence that GreatFree is a generic distributed programming paradigm. In fact, the ATM is an instance of the distributed system implemented with GreatFree. A regular implementation of the ATM is established with a tree-structured cluster, which contains one single collaborator and a lot of children. It is possible that the cluster is overloaded in practice because of heavy tasks. If so, it is convenient to employ an auto-scaling-out cluster to tolerate the potentially high burden on the fly. It is also feasible to update the topology of the cluster for large volume accessing in a wide area. An extreme case is that each node of the cluster is turned from one physical computer to a logical cluster for heavy pressure workload using the DCP of GreatFree. Whatever the implementation is, only GreatFree techniques are sufficient and necessary. As a matter of fact, what can be seen from the perspective of programmers is always a vast number of the ATM threads for them to govern.

C. The DCP

As a discovery in the domain of distributed programming, the DCP reveals that the code structures are steady whatever the heterogeneity is in any specific distributed computing environments. In other words, various heterogeneous distributed systems can be programmed with a limited number of homogeneous code-level design patterns. In particular, no matter how complicated a distributed system, it can be programmed in the homogeneous code structures using the DCP. In GreatFree, any distributed systems, distributed APIs, or distributed frameworks are programmed with the DCP in essence.

1) The Contributions of the DCP

The DCP is a rapid programming approach as it unveils the magic code structures for distributed systems development. It is made up with a limited number of code-level design patterns, which are the steady code structures to compose the DP, the DAA, and distributed frameworks. Each of the patterns plays the role of one particular member of those final systems only. As the DCP represents the fixed code structures, distributed programming with its support is simplified as the procedure to follow the limited number of predefined patterns to assembly distributed APIs. It is no doubt that the solution speeds up distributed systems development.

At first, the DCP discloses that GreatFree is a rapid distributed programming paradigm that provides sufficient and necessary building block. For the homogeneity of the DCP, the programming effort is lowed obviously. In other words, GreatFree is the craftily simplified solution to sustain the balance between the ease of distributed programming with the DCP and the coverage of distributed computing environments. GreatFree does not intend to conceal all the distributed techniques because of the complexity of the Internet-oriented computing environments.

In addition, the DCP reveals that a generic solution is achievable since the variety of heterogeneous distributed systems adhere to the common principle that they are homogeneous in terms of the distributed code structures. If the principle is luckily founded, the solution is certainly invented. GreatFree is no doubt such a generic programming paradigm. More important, the DCP demonstrates that it is practical to propose a generic and rapid distributed programming language which relies on GreatFree. For the sake of popularity, one choice is the object-oriented script although it is not the unique choice. Although it is impossible to conceal all the technical details of the Internet-oriented computing environment, it is feasible to abstract them in the same forms with the common code structures.

2) The Internal and External Patterns

Using GreatFree, after one distributed algorithm for one particular domain is investigated clearly, the approach to specify it is straightforward since the unique task left is to assemble the primitive distributed programming components. The procedure is equivalent to the one to construct a new DAA or a distributed framework, i.e., programming distributed algorithms with GreatFree results in high-level models. In other words, either a new DAA or a new distributed framework is created through aggregating the DP as well as existing DAAs upon the DCP with respect to the corresponding distributed algorithms. During the procedure, the DCP is the unique series of components to aggregate various distributed resources and mechanisms. In brief, the internals of any DAA and distributed frameworks in GreatFree are implemented through programming with the DCP.

In contrast, any newly created DAA has its own code-level design pattern, i.e., the idiom that encloses the API for rapid programming. Compared with those internal ones to form the new DAA, the pattern is called the external one since it is employed for the implementation that weaves itself outside with other distributed programming components to construct more complicated ones. Each DAA is similar as each DAA is implemented by low-level components in the DCP. Therefore, each DAA either keeps one of the DCPs as its external pattern or reconstructs a new pattern which is a straightforward aggregation of some of the DCP. No any new code-level design patterns are invented for any new DAAs no matter how complicated a DAA is. That proves that external patterns for DAAs conform to the principle of DCP as well. In other words, the DCP is a self-similar system. For a newly created distributed framework, it does not make sense to discuss about its patterns since it is a semi-constructed system in which no additional distributed programming efforts left except specifying applications and reusing the DCP before a distributed system is established. However, it is feasible to extract the core of one distributed framework to create a new DAA.

IV. The Cloud-Side Operating System

The cloud-side operating system is inspired by GreatFree. Since GreatFree is a generic distributed programming language which represents the common principles of any distributed systems over the Internet, it indicates that it is feasible to build a new cloud system that is a generic development and running environment for any distributed systems. On the other hand, as a new operating system, similar to traditional ones, it is necessary to have a proper language to support applications developments on it. Without doubt, GreatFree exhibits the proper choice to be competent to play the role.

A. The Relationships Between Programming Languages and Operating Systems

The relationships between programming languages and operating systems are concluded as follows. At first, a operating system needs to be implemented with a programming language. Additionally, after the operating system is constructed, the same programming language is required to be the technique to develop upper-level applications on it. In other words, without an appropriate programming language, any operating system can hardly be established, and the operating system is useless since it is only a development and running environment without any applications which end users can access.

A programming language is a series of common representations to describe and manage computing resources in one particular computing environment. It is highly recommended that the representations are written in the format that is as human-readable as possible such that developers can program with them conveniently. An operating system is a development and running environment that fits the computing circumstance exactly. Therefore, the system can be implemented rapidly with the language only. Any other low-level languages must bring heavy workloads for sure and any other high-level ones can never support the establishment of such a system.

Once if the operating system is constructed, it speeds up applications development in the same environment. Usually, many semi-constructed frameworks created by the language are preinstalled on the operating system such that they lower the efforts of application programmers extraordinarily. In most cases, developers focus on application level specifications only when working with those tools. However, it is possible that those tools cannot provide some complicated developments with sufficient supports. Then, the programming language is the last choice to overcome the potential barriers in those cases. Although the development efforts are higher than using those frameworks, it is still a feasible solution compared with those languages that are not focused specially on the particular computing environment. In practice, if the difficult cases are used frequently, new frameworks are created upon the programming language such that other programmers enjoy the convenience of the new frameworks.

The combination of UNIX/C is the most well-known example to present the relationships between a programming language and the operating system. Initially, C is a system programming language to specify algorithms that fit the standalone and sequential computing environment. Most code of UNIX is written in C, and it is tough to implement such a complicated system with earlier generation languages, such as assembly ones. After UNIX is constructed, it is a common sense that many function libraries are available over the platform for particular applications developments. Furthermore, during the procedure of UNIX’s popularization, a huge bunches of function libraries were implemented with C to ease applications developments over UNIX.

B. The Problems of Traditional Programming Languages and Operating Systems

With the development of Internet technology, most applications are required to run in the concurrent and distributed manner rather than the standalone and sequential one. Unfortunately, because no proper programming languages were available in the past days, the standalone and sequential languages played the major role to program various distributed systems with the support of networking and threading. The procedure is notoriously difficult because developers are forced to make every effort to convert the standalone and sequential programs to the distributed and concurrent ones.

Even though many distributed frameworks are created to lower the workload of distributed systems development, there is no way to modify them to adapt to new environments conveniently. To build a distributed system with low costs, instead of programming with a single language, a couple of third-party heterogeneous frameworks are put together roughly with inefficient protocols, such as HTTP/JSON, without knowing internals of each of them. If the system to be implemented is a large scale one, a lot of heterogeneous frameworks have to be pieced together. That is the nightmare of developers. In fact, because of the native drawbacks of traditional languages, it is a tough job for each developer to implement the simplest distributed system. Although frameworks help, because of the complexity of the Internet-oriented computing environment, it is impossible to program any distributed systems from scratch in most cases. However, piecing heterogeneous frameworks together always results in poor adaptability, low performance, high costs and maintenance difficulties.

The above problems also unveil that the current operating systems are not the proper development and running environment for distributed systems over the Internet-oriented computing environment. Since those operating systems are implemented with standalone and sequential languages, they do not provide distributed and concurrent systems with sufficient supports. That is the primary reason that almost each distributed application needs to be developed and run over frameworks rather than those operating systems directly. Because of that, those heterogeneous distributed frameworks are called the middleware layer between applications and the operating systems. The larger the scale of a distributed system, the more complicated the middle layer. It is not difficult to imagine the heavy overhead of computing resources consumption and the chaotic architectures.

In brief, at present, the fundamental software in terms of operating systems as well as programming languages is not well established for distributed systems' development and running.

C. The Concept of Cloud-Side Operating System and GreatFree

The cloud-side operating system is a generic development and running environment for distributed systems over the Internet-oriented computing circumstance. At this moment, such a system is still not available. When talking about the term of operating systems, it always represents the traditional ones, such as UNIX and Windows, which are viewed as the development and running platforms for standalone and sequential applications. Because of their native drawbacks, they are improper choices to support distributed systems development and running.

As a counterpart of traditional ones, the idea of the cloud-side operating system is originated from the generic programming language, GreatFree. At present, GreatFree is becoming more and more mature in the domain of distributed programming over the Internet. For that, it inspires the establishment of the cloud-side operating system. With its distinct characters for distributed programming, GreatFree is not only the correct technique to implement the cloud-side operating system but also the right choice to program upper level applications over the same platform.

V. CONCLUSIONS

By now, we have completed a complete delivery room procedure. This process is the core work of implementing the server using the distributed elements of GreatFree. You can see that all the programs involved are written according to the design patterns provided by GreatFree. As a beginner, there must be a process of adaptation to these patterns. But at least the process is straightforward. With traditional languages, accomplishing this task is uncertain and unwieldy, and even the most sophisticated programmers don't want to tread lightly. Lonely Chatter is just the simplest distributed system, but it gets harder in more complex distributed scenarios. In contrast, GreatFree has provided a set of design methods with consistent ideas, clear steps and stable forms. More importantly, it can be used for any distributed problem. No matter what system, these patterns are used repeatedly from simple to complex. This reflects the unique point of GreatFree technology.

Reference

[1] J. V. Guttag, “Introduction to Computation and Programming Using Python”, the MIT Press, ISBN: 978-0-262-52500-8, 2013.

[2] A. Gupta, “Java EE 7 Essentials”, O’Reilly Media, ISBN: 978-1-449-37017-6, 2013.

[3] A. Goncalves, “Beginning Java EE 7”, Apress, ISBN-10: 143024626X, ISBN-13: 978-1430246268, 2013.

[4] S. Newman, “Building Microservices - Designing Fine-Grained Systems”, O’Reilly, ISBN: 978-1491950357

[5] C. Richardson, “Microservice Patterns”, Manning Publications, ISBN-10: 1617294543, ISBN-13: 978-1617294549, 2018.

[6] Apache Whisk, https://openwhisk.apache.org

[7] AWS Lambda, https://aws.amazon.com/lambda

[8] IBM Cloud Functions, https://www.ibm.com/cloud/functions

[9] Google Cloud Functions, https://cloud.google.com/functions

[10] Microsoft Azure Functions, https://azure.microsoft.com/services/functions

[11] Oracle Fn Functions, https://fnproject.io

[12] Service-Oriented Architecture Standards – The Open Group, https://www.opengroup.org/forum/service-oriented-architecture-soa

[13] M. Bell, “Introduction to Service-Oriented Modeling, Service-Oriented Modeling: Service Analysis, Design, and Architecture”, Wiley & Sons, ISBN: 978-0-470-14111-3

[14] T. White, “Hadoop: The Definite Guide”, the Third Edition, O’Reilly, ISBN: 978-1-449-32891-7, 2012.

[15] S. Ghemawat, H. Gobioff, S. T. Leung, “The Google File System”, Proceedings of the 19th ACM SOSP, Pages: 29-43, 2003.

[16] V. Jason, “Pro Hadoop”, Apress, ISBN: 978-1-4302-1942-2, 2009.

[17] M. Zaharia, M. Chowdhury, M. Franklin, S. Shenker, I. Stoica, “Spark: Cluster Computing with Working Sets”, Technical Report UCB/EECS-2010-53, EECS Department, University of California, Berkeley, 2010.

[18] M. Zaharia, M. Chowdhury, T. Das, A. Dave, J. Ma, M. McCauley, M. Franklin, S. Shenker, I. Stoica, “Resilient Distributed Datasets: A Fault-Tolerant Abstractions for In-Memory Cluster Computing”, Technical Report UCB/EECS-2011-82, EECS Department, University of California, Berkeley, 2011.

[19] M. Luksa, “Kubernetes in Action”, Manning, ISBN-13: 978-1617293726, ISBN-10: 9781617293726, 2018.

[20] J. D. Moore, “Kubernetes: The Complete Guide To Master Kubernetes”, Independently Published, ISBN-10: 1096165775, ISBN-13: 978-1096165774, 2019 (not downloaded yet. 05/20/2019, LB).

[21] A. Shrivastwa, S. Sarat, K. Jackson, C. Bunch, E. Sigler, T. Campbell, “OpenStack: Building a Cloud Environment”, Packt Publishing, ISBN-10: 1787123189, ISBN-13: 978-1787123182, 2016.

[22] B. Silverman, M. Solberg, “OpenStack for Architectures: Design Production-Ready Private Could Infrastructure”, the Second Edition, Packt Publishing, ISBN-10: 1788624513, ISBN-13: 978-1788624510, 2018.

[23] D. R. Butenhof, “Programming with POSIX Threads”, Addison-Wesley, ISBN: 0-201-63392-2, 1997.

[24] B. Nichols, D. Buttlar, J. Farrell, “Pthreads Programming”, O’Reilly, ISBN: 1-5692-115-1, 1996.

[25] B. Goetz, T. Peierls, J. Bloch, J. Bowbeer, D. Holmes, D. Lea, “Java Concurrency In Practice”, Addison-Wesley Professional, ISBN-10: 0-321-34960-1, ISBN-13: 978-0-321-34960-6, 2006.

[26] D. Lea, “Concurrent Programming in Java, Design Principles and Patterns”, the Second Edition, Addison-Wesley, ISBN: 0-201-31009-0, 1999.

[27] S. Cleary, “Concurrency in C# Cookbook, Asynchronous, Parallel, and Multithreaded Programming”, O’Reilly, ISBN: 978-1-449-36756-5, 2014.

[28] C. Hughes, T. Hughes, “Parallel and Distributed Programming Using C++”, Addison-Wesley, ISBN: 0-13-101376-9, 2003.

[29] R. Terrell, “Concurrency in .NET, Modern Patterns of Concurrent and Parallel Programming”, Manning, ISBN: 978-1-617-29299-6, 2018.

[30] H. Okamura, M. Tokoro, “The Design and Implementation of ConcurrentSmalltalk”, Proceedings of the First ACM Conference on Object-Oriented Programming Systems, Languages, and Applications, Pages: 331-340, 1986.

[31] Y. Yasuhiko, “The Design and Implementation of ConcurrentSmalltalk”, Proceedings of Conferences on Object-Oriented Programming Systems, Languages and Applications, Pages: 331-340, 1986.

[32] H. Okamura, M. Tokoro, “ConcurrentSmalltalk-90”, Proceedings of TOOLS Pacific’90, 1990.

[33] I. Balbaert, “Rust Essentials”, Packt Publishing, ISBN: 978-1-78528-576-9, 2015

[34] G. Zaccone, “Python Parallel Programming Cookbook”, Packt Publishing, ISBN: 978-1-78528-958-3, 2015.

[35] A. Shrivastwa, S. Sarat, K. Jackson, C. Bunch, E. Sigler, T. Campbell, “OpenStack: Building a Cloud Environment”, Packt Publishing, ISBN-10: 1787123189, ISBN-13: 978-1787123182, 2016.

[36] B. Silverman, M. Solberg, “OpenStack for Architectures: Design Production-Ready Private Could Infrastructure”, the Second Edition, Packt Publishing, ISBN-10: 1788624513, ISBN-13: 978-1788624510, 2018.

[37] K. Jackson, C. Bunch, E. Sigler, J. Denton, “OpenStack Cloud Computing Cookbook”, Packt Publishing, ISBN-10: 1788398769, ISBN-13: 978-1788398763, 2018.

[38] J. Rutherglen, D. Wampler, E. Capriolo, “Programming Hive”, O’Reilly, ISBN: 978-1-449-31933-5, 2012.

[39] M. Armbrust, R. S. Xin, C. Lian, Y. Huai, D. Liu, J. K. Bradley, X. Meng, T. Kaftan, M. J. Franklin, A. Ghodsi, M. Zaharia, “Spark SQL: Relational Data Processing in Spark”, Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data, Pages: 1383-1394, 2015.

[40] A. Sarkar, “Learning Spark SQL: Architect Streaming Analytics and Machine Learning Solution”, Packt Publishing, ISBN-10: 1785888358, ISBN-13: 978-1785888359, 2017.

[41] F. Chang, et. al., “Bigtable: A Distributed Storage System for Structured Data”, Journal of ACM Transaction on Computer Systems (TOCS), Volume 26, Issue 2, Article No. 4, Pages: 4:2-4:26, 2008.

[42] N. Dimiduk, A. Khurana, “HBase In Action”, Manning Publications, ISBN: 978-1617290527, 2012.

[43] L. Georgo, “HBase: The Definitive Guide”, O’Reilly Media, ISBN: 978-1-449-39610-7, 2011.

[44] S. Akhtar, R. Magham, “Pro Apache Phoenix: An SQL Driver for HBase”, the First Edition, Apress, ISBN-10: 9781484223697, ISBN-13: 978-1484223697, 2016.

[45] M. Kornacker, et. al., “Impala: A Modern, Open-Source SQL Engine for Hadoop”, Proceedings of the 7th Biennial Conference on Innovative Data Systems Research (CIDR’15), 2015.

[46] J. Russell, “Getting Started with Impala”, ISBN-10: 1491905778, ISBN-13: 978-1491905777, O’Reilly Media, 2015.

[47] A. Katsifodimos, S. Schelter, “Apache Flink: Stream Analytics at Scale”, Proceedings of 2016 IEEE International Conference on Cloud Engineering Workshop (IC2EW), Pages: 193-193.

[48] P. Carbone, A. Katsifodimos, S. Ewen, V. Markl, S. Haridi, K. Tzoumas, “Apache Flink: Stream and Batch Processing in a Single Engine”, Bulletin of the IEEE Computer Society Technical Committee on Data Engineering, Volume 36, No. 4, Pages: 17-29, 2015.

[49] F. Hueske, V. Kalavri, “Stream Processing with Apache Flink”, O’Reilly Media, ISBN-10: 149197429X, ISBN-13: 978-1491974292, 2019.

[50] K. M. M. Thein, “Apache Kafka: Next Generation Distributed Messaging System”, International Journal of Scientific Engineering and Technology Research, ISSN: 2319-8885, Volume: 03, Issue: 47, Pages: 9478-9483, 2014.

[51] N. Garg, “Apache Kafka”, Packt Publishing, ISBN: 978-1-78216-793-8, 2013

[52] S. T. Allen, M. Jankowskl, P. Pathirana, “Storm Applied: Strategies for Real-Time Event Processing”, Manning Publications, ISBN-10: 1617291897, ISBN-13: 978-1617291890, 2015.

[53] S. Chintapalli, D. Dagit, B. Evans, R. Farivar, T. Graves, M. Holderbaugh, Z. Liu, K. Nusbaum, K. Patil, B. J. Peng, P. Poulosky, “Benchmarking Streaming Computation Engines: Storm, Flink and Spark Streaming”, Proceedings of 2016 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW), Pages: 1789-1792 (not downloaded yet, 06/24/2019, LB).

[54] C. Olston, B. Reed, U. Srivastava, R. Kumar, A. Tomkins, “Pig-Latin: A Not-So-Foreign Language for Data Processing”, Proceedings of ACM SIGMOD International Conference on Management of Data, Pages: 1099-1110, 2008.

[55] A. Gates, D. Dal, “Programming Pig: Dataflow Scripting with Hadoop”, O’Reilly Media, ISBN-10: 9781491937099, ISBN-13: 978-14919337099, 2016.

[56] M. Islam, A. K. Huang, M. Battisha, M. Chiang, S. Srinivasan, C. Peters, A. Neumann, A. Abdeinur, “Oozie: Towards a Scalable Workflow Management System for Hadoop”, Proceedings of the 1st ACM SIGMOD Workshop on Scalable Workflow Execution Engines and Technologies, Pages: 4-13, 2012.

[57] M. K. Islam, A. Srinivasan, “Apache Oozie: The Workflow Scheduler for Hadoop”, ISBN-10: 1449369928, ISBN-13: 978-1449369927, 2015.

[58] D. Smiley, E. Pugh, K. Parisa, M. Mitchell, “Apache Solr Enterprise Search Server”, the 3rd Edition, Packt Publishing, ISBN: 978-1-78216-136-3, 2015.

[59] A. Serafini, “Apache Solr: Beginner’s Guide”, Packt Publishing, ISBN: 978-1-78216-252-0, 2013.

[60] J. Brittain, I. F. Darwin, “Tomcat: The Definitive Guide”, the 2nd Edition, O’Reilly Media, ISBN-10: 0-596-10106-6, ISBN-13: 978-0596-10106-0, 2007.

[61] D. Thomas, “Programming Elixir >= 1.6: Functional |> Concurrent |> Pragmatic |> Fun”, Pragmatic Bookshelf, ISBN-10: 1680502999, ISBN-13: 978-1680502992, 2018.

[62] S. Juri, “Elixir In Action”, the 2nd Edition, Manning Publications, ISBN-10: 1617295027, ISBN-13: 978-1617295027, 2019.

[63] J. Armstrong, “A History of Erlang”, Proceedings of the Third ACM SIGPLAN Conferences on History of Programming Languages, Pages: 6-1 – 6-26, 2007.

[64] J. Armstrong, “The Development of Erlang”, Proceedings of the 2nd ACM SIGPLAN International Conference on Functional Programming, Pages: 196-203, 1997.

[65] J. Armstrong, “Making Reliable Distributed Systems in the Presence of Software Errors”, PhD Dissertation, Royal Institute of Technology, 2003.

[66] J. Armstrong, “Erlang”, Communications of the ACM, Volume: 53, No. 9, Pages: 68-75, 2010

[67] J. Armstrong, R. Virding, C. Wikstrom, M. Williams, “Concurrent Programming in Erlang”, the 2nd Edition, Prentice Hall, ISBN-10: 013508301X, ISBN-13: 978-0135083017, 1996.

[68] J. Armstrong, “Programming Erlang: Software for a Concurrent World”, the 2nd Edition, Pragmatic Bookshelf, ISBN-13: 978-1-937785-53-6, 2013.

[69] F. Cesarini, S. Thompson, “Erlang Programming: A Concurrent Approach to Software Development”, O’Reilly Media, ISBN-10: 0596518188, ISBN-13: 978-0596518189, 2009.

[70] V. A. Sarawart, K. Kahn, J. Levy, “Janus: A Step Towards Distributed Constraint Programming”, Proceedings of the 1990 North American Conference on Logic Programming, Pages: 431-446, 1990.

[71] V. A. Saraswat, M. Rinard, P. Panangaden, “The Semantic Foundations of Concurrent Constraint Programming”, Proceedings of Ninth ACM Symposium on Principles of Programming Languages, Pages: 333-352, 1991.

[72] D. Gudeman, S. K. Debray, K. DeBosschere, “jc: an Efficient and Portable Sequential Implementation of Janus”, Proceedings of the International Conference and Symposium on Logic Programming, Pages: 399-416, 1992.

[73] Red Programming Language, https://www.red-lang.org

[74] C. Varela, G. Agha, “Programming Dynamically Reconfigurable Open Systems with SALSA”, Proceedings of ACM Conference on Object-Oriented Programming, Systems, Languages, and Applications, Pages: 20-34, 2001

[75] B. Nobakht, F. S. de Boer, “Programming with Actors in Java 8”, Proceedings of Leveraging Applications of Formal Methods, Verification and Validation, Specialized Techniques and Applications, Pages: 37-53, 2014

[76] Akka, https://akka.io

[77] M. K. Gupta, “Akka Essentials”, Packt Publishing, ISBN-10: 1849518289, ISBN-13: 978-1849518284, 2012

[78] D. Wyatt, “Akka Concurrency”, Artima Inc., ISBN-10: 0981531660, ISBN-13: 978-0981531663, 2012

[79] R. Roestenburg, R. Bakker, R. Williams, “Akka in Action”, Manning Publications, ISBN-10: 1617291013, ISBN-13: 978-1617291012, 2016

[80] V. Vernon, “Reactive Messaging Patterns with the Actor Model: Applications and Integration in Scala and Akka”, Addison-Wesley Professional, ISBN-10: 0133846830, ISBN-13: 978-0133846836, 2015

[81] P. Haller, F. Sommers, “Actors in Scala”, Artima Inc., ISBN-10: 0981531652, ISBN-13: 978-0981531656, 2012

[82] N. Raychaudhuri, C. Fowler, “Scala in Action”, Manning Publications, ISBN-10: 1935182757, ISBN-13: 978-1935182757, 2013

Research on Improved Dual Channel Medical Short Text Intention Recognition Algorithm

Chao Wang
Xi'an Technological University
College of Computer Science and Technology
Xi'an, China
E-mail:1501873640@qq.com

Fei Xu
Xi'an Technological University
College of Computer Science and Technology
Xi'an, China
E-mail:xufei@xatu.edu

Yongyong Sun
Xi'an Technological University
College of Computer Science and Technology
Xi'an, China
E-mail:yongsunjd@126.com

Abstract—**The increasing application of medical robots in the healthcare sector underscores the critical importance of intent recognition in enhancing the interaction and assistance capabilities of these robots. Traditional intent recognition methods utilize convolutional neural networks (CNNs) for text analysis but often fall short in capturing global features, resulting in incomplete information. To address this challenge, this paper introduces an innovative approach by combining an enhanced CNN with bidirectional gated recurrent units (BiGRU) to construct a dual-channel short-text intent recognition model. This model effectively leverages both local and global features to more accurately comprehend user needs and intentions. Experimental results demonstrate that this model excels, achieving an accuracy rate of 96.68% and an F1 score of 96.67% on the THUCNews_Title dataset. In comparison to conventional intent recognition models, it exhibits significantly improved performance, thereby providing substantial support for medical robots in patient care and assisting healthcare professionals.**

Keywords-Intention Recognition; Albert; Bigru; Dual Channel

I. INTRODUCTION

Natural language understanding(NLU) plays a fundamental role in robot question-answering systems in the medical field. Exceptional intent recognition modules help simplify the complexity of NLU, allowing robots to more effectively process text by categorizing intricate questions into the relevant intents. Medical question-answering is a focal point in robotics research within the medical domain due to the highly specialized nature of medical knowledge. Accurately identifying the intent of questions enables robots to better integrate medical knowledge, thus enhancing search result performance. In comparison to systems without intent recognition or those with suboptimal performance, outstanding medical question-answering modules can significantly alleviate the workload of subsequent robot tasks. From both input and output perspectives, intent recognition tasks can be regarded as text classification tasks within the realm of natural language processing, providing foundational support for robot work in the medical domain.

With the continuous advancement of neural network technologies, researchers have dedicated substantial efforts to improve intent recognition, particularly in the context of medical question-answering. Short-text medical queries, characterized by their brevity and concise information, often pose challenges for traditional intent recognition methods. The accuracy of natural language understanding is paramount to the performance of the entire question-answering system. Inaccurate comprehension in the early

DOI: 10.2478/ijanmc-2023-0079

stages of information retrieval can lead to subsequent inaccurate responses. Therefore, the aim of this study is to provide more accurate natural language understanding tools, especially for handling interrogative sentences in the domain of medical robotics. In this paper, we introduce a dual-channel medical short-text intent recognition model, denoted as the AB-CNN-BGRU-att model. This model combines TextCNN with BiGRU-att and employs multiple pooling strategies. The BiGRU-att module employs a dual-channel approach to capture features at different levels, thereby capturing the global information within the text. Simultaneously, TextCNN leverages different-sized convolution kernels and pooling strategies to extract a broader array of local features. Experimental results demonstrate that the AB-CNN-BGRU-att model outperforms other popular intent recognition models, particularly in the context of medical robotics applications. This model significantly enhances a robot's ability to comprehend interrogative sentences.

II. Related works

The document [1] initially introduced Convolutional Neural Networks (CNN), originally used in computer vision, as the TextCNN model, a classic model in text classification. Later, WANG Haitao et al. [2] addressed TextCNN's shortcomings in handling short texts by employing non-linear sliding methods and N-gram models. Ma Sidan et al. [3] improved Word2vec by utilizing text similarity for classification. Subsequently, Sun Hong et al. [4] and Chi Haiyang et al. [5] used BERT as an embedding layer, incorporating BiGRU to capture global sentence features, and utilized attention mechanisms for classification, demonstrating improvements on their respective datasets. Due to the large size of the BERT model, Wen Chaodong et al. [6] and Zeng Cheng [7] proposed using the ALBERT model as an enhancement. Wen Chaodong's experiments with the ALBERT-BiGRU model exceeded the performance of Word2vec and GloVe. Zeng Cheng, using the ALBERT model, demonstrated improved F1 values compared to other models by connecting the CNN layer and feeding it into the BiGRU layer for classification.

Recent scholars [8-10] emphasize the significance of both local and global features in short text corpora. Their multi-channel approach processes inputs independently from the embedding layer and merges features for classification. Additionally, Wu Di et al. [11] proposed an enhanced embedding layer in a dual-channel model by combining static and dynamic word vectors from ELMo and GloVe, outperforming traditional models on datasets such as IMDB.

III. Model framework

The TextCNN model uses CNN for text feature extraction but overlooks the entire sentence context. To address this, we augment BERT TextCNN with a BiGRU network to capture global features. We replace the heavier BERT with the lighter ALBERT, which still generates rich word vectors but with fewer parameters. These vectors feed into TextCNN and BiGRU to respectively extract local and global features. TextCNN utilizes various kernel sizes and pooling strategies, while BiGRU captures global features. These features are merged, and using Dropout and softmax, probability values are computed for multi-classification. Refer to Figure 1 for the model architecture.

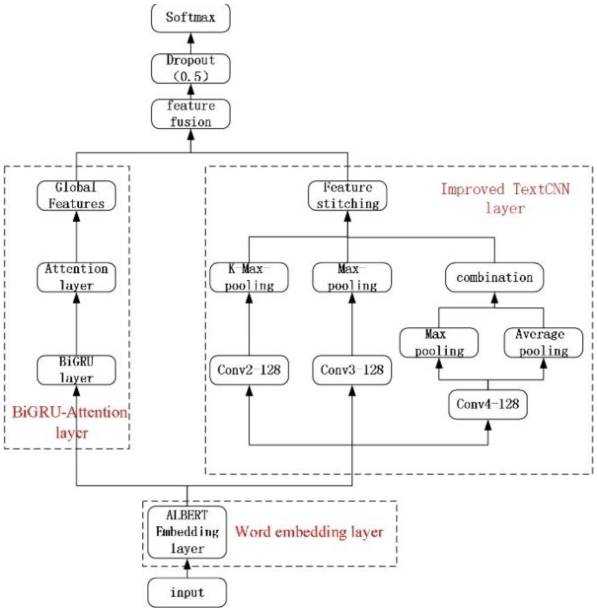


Figure 1. Architecture diagram of AB-CNN-BGRU-att model

A. Word embedding layer

Several Word embedding models are widely used, including Word2vec, GloVe, and BERT. Among these, BERT has gained recognition in numerous experiments within the NLP community, being considered one of the top models. ALBERT, a variation of BERT, simplifies the original BERT while maintaining similar performance. Official data from the ALBERT paper reveals that it achieves comparable performance to BERT base across several representative tasks, yet with significantly fewer parameters—six times fewer—and nearly three times faster processing time. Consequently, this paper adopts ALBERT for the Word embedding layer due to its efficiency, as illustrated in Figure 2.

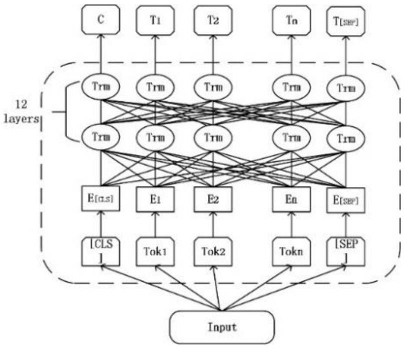


Figure 2. ALBERT model structure

When text enters the Word embedding layer, it's marked with [CLS] and [SEP] to show sentence boundaries. The resulting serialized text generates the En vector, which, processed by the Transformer encoder, produces Tn from its features. ALBERT and BERT both employ the Transformer's encoder section, composed of multiple identical network layers featuring residual connections between the "Multi Head Attention" and "FeedForward" layers. The "Multi Head Attention" layer functions on input vectors Q, K, and V, derived from text queries, keys, and values in the sequence, with equations (1) to (3) defining the specific computations.

$$head_t = Attention(QW_t^Q, KW_t^K, VW_t^V), t \in (1,2,...,h), \quad (1)$$

$$Attention(Q,K,V) = Softmax(\frac{QK^T}{\sqrt{(d_t)}})V \quad (2)$$

Merge the resulting matrices:

$$MultiHead(Q,K,V) = Concat(head_1, head_2, ..., head_h)W^0 \quad (3)$$

W^0 represents the weight matrix to ensure the final matrix's dimensions align with the sequence length, W_t^Q, W_t^K, W_t^V represents the weight matrices for individual Q, K, and V vectors, while d_t denotes their dimensional size.

B. BiGRU-att Module

To capture global text features and enhance the model's grasp of the text's core concept, this study integrates a BiGRU layer following ALBERT's word vector output. This layer extracts comprehensive feature details for the entire sentence. The network structure of the GRU is depicted in Figure 3.

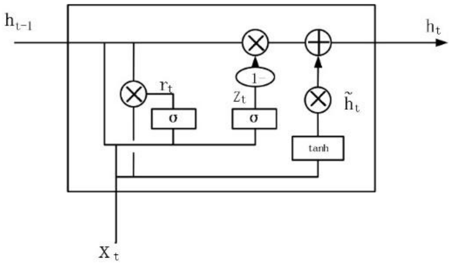


Figure 3. GRU network structure

The standard GRU's hidden state is unidirectional, focusing solely on the present input state without considering the impact of the text context on this state. This unidirectional nature fails to capture how subsequent information affects preceding states. To address this limitation, this paper employs BiGRU, a variant of GRU. BiGRU integrates two GRU layers with opposite directions, allowing output information to be influenced by both directional outcomes. Formula (4) demonstrates the final output result, while Figure 4 illustrates the BiGRU model structure.

$$h_t^{(i)} = [\vec{h}_t^{(i)}, \overleftarrow{h}_t^{(i)}] \quad (4)$$

In the above equation, $\vec{h}_t^{(i)}$ represents the information obtained by the i-th text passing through the forward GRU, and $\overleftarrow{h}_t^{(i)}$ represents the information obtained by the i-th text passing through the backward GRU. $h_t^{(i)}$ is the final result obtained from this text through BiGRU.

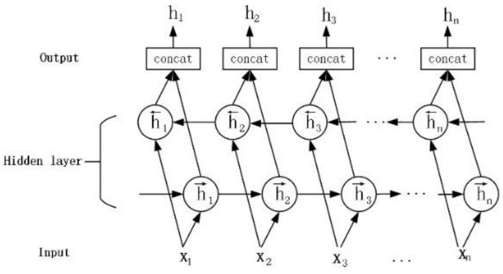


Figure 4. BiGRU network structure

The attention mechanism assigns weights to words in the text to prioritize crucial features, enabling the model to concentrate on words with higher weight scores and enhance classification accuracy. In this process, the Attention layer computes word weights for each BiGRU output vector, generating a final sentence representation by the weighted sum of these scores and corresponding position feature vectors. This BiGRU Attention layer enables the model to autonomously emphasize significant words with higher weight scores, thereby improving its ability to capture global features in the input text.

C. Improved TextCNN Module

The revised TextCNN model includes multiple convolutional layers with varied sizes, diverse pooling layers, and fully connected layers, an advancement from the original TextCNN model. The model architecture is depicted in Figure 5.

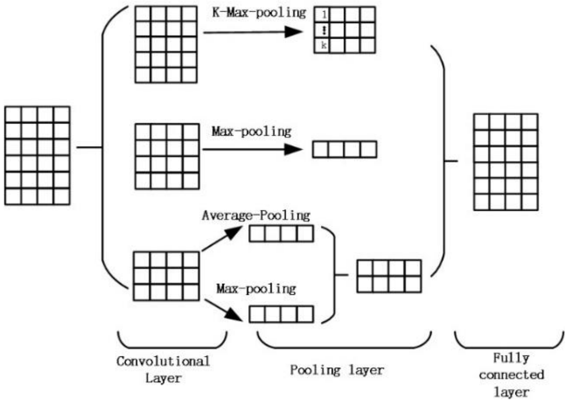


Figure 5. Improved TextCNN model structure

The enhanced TextCNN layer conducts convolution by examining text features using various convolution kernel sizes, aligned with the Token Embedding's dimension, and the row information of the vector matrix represents words. When the width of the convolution kernel is inconsistent with the dimension of Token Embedding, the convolution kernel cannot extract complete word information. This matrix acquires the characteristic mapping matrix. The enhanced TextCNN layer conducts convolutions on text features using diverse kernel sizes. The kernel width matches the Token Embedding dimension, with rows in the vector matrix representing words. If the kernel width mismatches the Token Embedding dimension, complete word information might not be extracted. Through a nonlinear Activation function, this matrix obtains the feature mapping matrix $c = [c_1, c_2, \cdots, c_n]$.The characteristic formula is presented in Formula (5).

$$c_i = f(w \cdot x_{i:i+h-1} + b) \tag{5}$$

Where, f is the Activation function, w is the weight matrix of the convolution kernel, and b is the offset term.

In short text questions, the TextCNN model focuses on extracting local features due to its dual-channel structure and limited sentence length. Unlike common choices such as kernel sizes 3, 4, and 5, this model selects sizes of 2, 3, and 4 for the convolutional kernels. For kernel size 2 convolution, the model uses K-Max pooling. Which selects the top K scores during pooling, capturing more abundant information compared to the typical maximum pooling method. The latter overlooks repeating features, seeing them only once, while K-Max pooling retains relative order information between some features by retaining K higher-scored features.

For kernel size 3, the model uses max pooling, focusing on essential text features by discarding weaker ones, minimizing noise, and emphasizing keywords.

Using a kernel size of 4, the model employs both maximum and average pooling strategies. Concatenating the resulting features is beneficial as maximum pooling focuses on the highest-scored feature, whereas average pooling considers each word's information.

After utilizing different kernel sizes and corresponding pooling operations, diverse local features are acquired. To prevent overfitting, a dropout layer follows the TextCNN pooling layer, enhancing the model's generalization. These features, combined with global features from the BiGRU-att module, form the final feature vector. Classification results are determined through the final fully connected layer, as outlined in Formula (6).

$$Z = soft\max(W_Z \cdot F + b) \tag{6}$$

Among these, Z stands for the predicted intention tag result, $soft\max$ represents the Activation function, W_Z indicates the weight of the fully connected layer, F is the final feature vector, and b represents the offset term.

D. AB-CNN-BGRU-att algorithm

The AB-CNN-BGRU-att (ALBERT-TextCNN-BiGRU-attention) algorithm determines intention labels' probabilities for intention recognition by analyzing the input text corpus. Its detailed process is depicted in Algorithm 1.

ALGORITHM I. AB-CNN-BGRU-ATT ALGORITHM FLOW

```
Algorithm: AB-CNN-BGRU-att algorithm
Input: $S = (s_1, s_2, s_3, \cdots s_n)$, $s$ is the input text sequence
Output: Intention identification label results
1. Data preprocessing, importing training sets, testing sets
2. Load the ALBERT model to obtain dynamic word vectors Token
3. $conv_{output_{1\sim n}} = Conv_{1\sim n}(T)$;
4. $pooling_{output_{1\sim n}} = Pooling(Conv_{output_{1\sim n}})$;
5. $cnn\_output = Concat(pooling_{output_{1\sim n}})$;
6. $forward = GRU(T)$;
7. $backward = GRU(T)$;
8. $bigru\_output = Concat(forward, backward)$;
9. $output = Concat(cnn\_output, bigru\_output)$;
10. $dropout = Dropout(output)$;
11. $dense = Dense(dropout)$;
12. $out = Softmax(dense)$;
13. $END$.
```

IV. EXPERIMENT AND RESULT ANALYSIS

A. Experimental data

The paper uses two datasets for experiments. The first one, THUCNews_Title, is drawn from THUCNews, containing 200,000 titles, each not exceeding 30 characters. It covers 10 categories. Table 1 illustrates the THUCNews_Title dataset.

The study focuses on common medical conditions. The KUAKE-QIC dataset, sourced from Alibaba Tianchi Laboratory, validates the model's performance. This dataset aims to improve search result relevance in medical queries, crucial in a field with specialized knowledge. It includes 11 categories. There are 6931 training, 1955 validation, and 1994 test samples. Approximately 96% of the data (6684 samples) contain less than 30 words, fitting the experimental criteria for short medical text datasets. Table 1 displays the KUAKE-QIC dataset.

TABLE I. EXPERIMENTAL DATASET

Name	Training Set	Test Set	Validation Set	Category	Total
KUAKE-QIC	6931	1994	1955	11	10880
THUCNews _Title	180000	10000	10000	10	200000

B. Parameter settings

The key parameters of the improved version of TextCNN in the AB-CNN-BGRU att model are as follows: word vector dimension is 384, activation function uses ReLu, learning rate is 1e-5, Dropout is 0.5, and batch size is 128. The key parameters of

BiGRU att in the AB-CNN-BGRU att model are as follows: hidden layer size is 256, word vector dimension is 384, activation function uses ReLu, Dropout is 0.2, and batch size is 128.

C. Experimental result

By tuning the model's hyperparameters and training it on the THUCNews_Title dataset, the model was tested on the THUCNews_Title test set. The obtained results for each category were compared. It's observed that the model achieves classification scores above 90% for each category. Notably, technology-related texts pose higher complexity and uncommon vocabulary, while stocks and social texts share similarities with other labels, leading to potential confusion and reduced accuracy. Overall, the model demonstrates its ability to accurately identify intentions in short texts, effectively interpreting text intentions despite limited sentence information and a concise corpus. Figure 6 displays the validation results of the model.

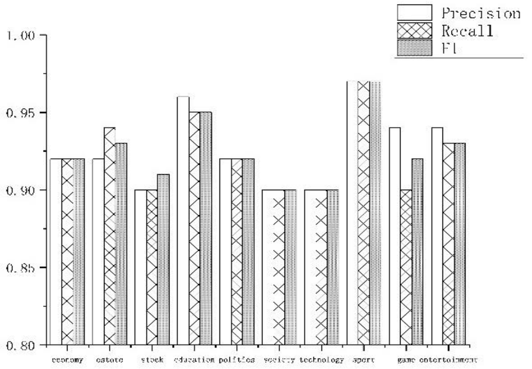


Figure 6. Model validation results

In the KUAKE-QIC dataset experiment, the model was trained for 20 epochs. The test set achieved an accuracy of 86.02, positioning it as the third-ranked model in the CBLUE3.0 ranking. The top-ranking model achieves an accuracy of 87.0117, followed by the second and third models with accuracies of 86.0589 and 85.9084, respectively.

D. Comparative experiment

In an experiment comparing the BiGRU att and BiLSTM att layers, this study maintained consistent parameters across two network layers. The comparison encompassed factors like average time per epoch, total training time, final accuracy, and F1 value. Figure 7 displays the contrast in epoch times, while Table 3 offers a comprehensive overview. Notably, the BiGRU att layer significantly outperformed the BiLSTM att layer in training efficiency. Although both layers achieved comparable accuracy and F1 scores, the study opts for the BiGRU-att layer due to its superior training efficiency and outcomes.

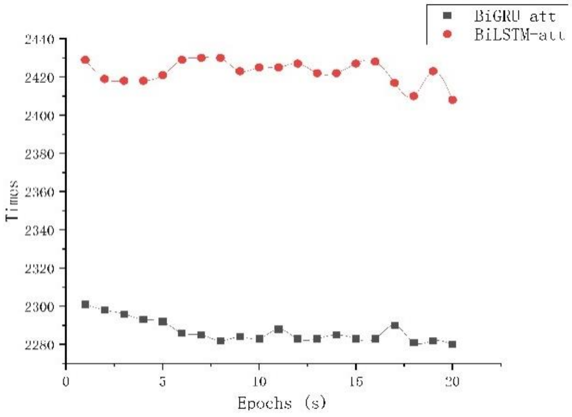


Figure 7. Comparison of Network Time

TABLE II. COMPARISON BETWEEN BIGRU-ATT AND BILSTM-ATT

Network Layer	Average Duration	Total Duration	Acc%	F1%
BiGRU-att	2286.9s	45738s	90.83	90.64
BiLSTM-att	2422.55s	48451s	90.45	90.41

To confirm our model's superiority under identical conditions to other models, we conducted comparative experiments using the THUCNews_Title dataset. Details of the models used in the experiments are outlined below:

SAttBiGRU: Utilizes BiGRU to capture global features and enhances text features by applying Self Attention, providing richer feature information for classification.

Self-Attention-CNN: Combines Self Attention with the fundamental TextCNN. It applies weighting using Self Attention to compact text information from the TextCNN's embedding layer. Following max pooling, the fully connected layer outputs classification results.

BiGRU-MCNN: Global features are extracted via BiGRU, while detailed local features are obtained through multi-channel CNN. The model then merges these two feature types and utilizes a fully connected layer to generate classification outcomes.

MC-AttCNN-AttBiGRU: Initially employs the attention mechanism to weigh multi-channel CNN and BiGRU. Subsequently, it concatenates the derived text feature vectors and conducts classification via a fully connected layer.

TABLE III. COMPARISON OF EXPERIMENTAL RESULTS

Model	Acc%	Pre%	Recall%	F1%
SAttBiGRU	96.16	96.20	96.16	96.17
Self-Attention-CNN	94.85	94.89	94.85	94.85
BiGRU-MCNN	95.43	95.45	95.43	95.43
MC-AttCNN-AttBiGRU	95.93	95.98	95.93	95.93

The results in Table 4 demonstrate the performance superiority of the AB-CNN-BGRU-att model over other models. Across various indicators using the THUCNews_Title dataset, this model exhibits a consistent improvement of one to two percentage points compared to the best-performing existing models. These comparative findings substantiate the advantageous performance of the AB-CNN-BGRU-att model proposed in this study.

E. Ablation experiment

Ablation experiments were performed to assess the efficiency of the proposed model for short text classification. Each local network element - TextCNN, enhanced TextCNN, BiGRU att, and AB-CNN-BGRU-att - underwent individual analysis in these.ALBERT was utilized as the Word embedding layer. The findings from the ablation experiment are summarized in Table 9:

TABLE IV. RESULTS OF ABLATION EXPERIMENT

Model	Acc%	Pre%	Recall%	F1%
TextCNN	89.96	89.90	89.96	89.90
Improved TextCNN	94.85	94.89	94.85	94.85
BiGRU-att	94.00	94.17	94.00	94.90
AB-CNN-BGRU-att	96.68	96.68	96.67	96.67

Table 5 demonstrates that the basic TextCNN model yielded unsatisfactory classification results with all indicators below 90%. This poor performance might stem from TextCNN's inefficiency in handling short texts. In contrast, the improved TextCNN model significantly enhanced all indicators. Employing various convolution kernel sizes and pooling strategies proved effective in obtaining richer local features, enhancing the model's performance. However, the BiGRU-att model exhibited slightly lower performance compared to the improved TextCNN model, highlighting the importance of global features in recognizing intentions, the AB-CNN-BGRU-att model, integrating local and global features, demonstrated an enhancement of almost two percentage points over the improved TextCNN and BiGRU-att models.

V. CONCLUSIONS

This paper introduces a dual-channel intent recognition model for medical short texts by combining Convolutional Neural Network (CNN) and Bidirectional Gated Recurrent Unit with Attention (BiGRU-Att). It also incorporates ALBERT, BiGRU attention, and an enhanced TextCNN model. The model processes vectors obtained from ALBERT separately, sending them to the BiGRU-Att network model for global feature extraction and the TextCNN model for local feature extraction using multiple pooling strategies and a hybrid pooling approach. After merging these two types of features, a classification result is obtained through a fully connected layer with softmax activation. This model's performance is evaluated against four other models using publicly accessible datasets.

Comparative experimental data clearly demonstrate the superior performance of the proposed model across various evaluation metrics. The experimental data also demonstrate the model's capacity to yield more precise intent recognition outcomes, which are crucial for the tasks performed by medical robots in the healthcare domain.

While the model in this study excels in short-text intent recognition for medical domain robots, it still heavily relies on extensive annotated datasets as the foundation. Subsequent work will explore semi supervision learning approaches to reduce manual annotation efforts and simultaneously enhance model performance, thus better supporting the applications of medical domain robots.

ACKNOWLEDGMENT

One of us (R.B.G.) thanks New Network and Detection Control National Joint Local Engineering Laboratory Fund Project.

REFERENCES

[1] Yoon Kim. Convolutional Neural Networks for Sentence Classification. [J]. CoRR, 2014 ,abs/1408. 5882(abs/ 14 08.5882).

[2] WANG Haitao, HE Jie, ZHANG Xiaohong, LIU Shufen. A Short Text Classification Method Based on N-Gram and CNN [J]. Chinese Journal of Electronics, 2020,(02): 248-254.

[3] MA Si-dan, LIU Dong-su. Text Classification Method Based on Weighted Word2vec [J]. Information Science, 2019, (11):38-42.

[4] SUN Hong, CHEN Qiang-yue. Chinese Text Classification Based on BE R T and Attention[J]. Journal of Chinese Computer Systems, 2022, (01):22-26.

[5] CHI Haiyang, YAN Xin, ZHOU Feng, XU Guangyi, ZHANG Lei. An online health community user intention identification method based on BERT-BiGRU-Attention [J]. Journal of Hebei University of Science and Technology, 2020, (03):225-232.

[6] WEN Chaodong, ZENG Cheng, REN Junwei , ZHANG Yan. Patent text classification based on ALBERT and bidirectional gated recurrent unit [J]. Journal of Computer Applications, 2021, (02):407-412.

[7] ZENG Cheng, WENE Chaodong, SUN Yumin, PAN Lie, HE Peng. Motional Analysis of Bullet Screen Text Based on ALBERT-CRNN [J]. Journal of Zhengzhou University(Natural Science Edition), 2021, 53(3): 1-8.

[8] LI Yang, DONG Hongbin. Text sentiment analysis based on feature fusion of convolution neural network and bidirectional long short-term memory network [J]. Journal of Computer Applications, 2018, (11):3075-3080.

[9] LI Qihang, LIAO Wei, MENG Jingwen. Dual-Channel DAC-RNN Text Classification Model Based on Attention Mechanism [J/OL]. Computer Engineering and Applications: 1-9(2021-04-21) [2022-01-25].

[10] SONG Zhongshan,NIU Yue,ZHENG Lu,TIE Jun, JIANG Hai. Multiscale double-layer convolution and global feature text classification model [J]. Computer Engineering and Applications:1-11.

[11] WU Di, WANG Ziyu, ZHAO Weichao. ELMo-CNN-BiGRU Dual-Channel Text Sentiment Classification Model [J]. Computer Engineering, 2022, (08):105-112.

Research and Implementation of Forest Fire Detection Algorithm Improvement

Xi Zhou
School of Computer Science and Engineering
Xi'an Technological University
Xi'an, China
E-mail: 2680620694@qq.com

Changyuan Wang
School of Computer Science and Engineering
Xi'an Technological University
Xi'an, China
E-mail: cyw@163.com

***Abstract*—To overcome low efficiency and accuracy of existing forest fire detection algorithms, this paper proposes a network model to enhance the real-time and robustness of detection. This structure is based on the YOLOv5 target detection algorithm and combines the backbone network with The feature extraction module combines the attention module dsCBAM improved by depth-separable convolution, and replaces the loss function CIoU of the original model with a VariFocal loss function that is more suitable for the imbalanced characteristics of positive and negative samples in the forest fire data set. Experiments were conducted on a self-made and public forest fire data set. The accuracy and recall rate of the model can reach 87.1% and 81.6%, which are 7.40% and 3.20% higher than the original model, and the number of images processed per second reaches 64 frames, a growth rate of 8.47%. At the same time, this model was compared horizontally with other improved methods. The accuracy, recall rate and processing speed were all improved in the range of 3% to 10%. The effectiveness of the improved method in this article was verified, and the external perception level of the forest fire scene was deeper.**

Keywords-Fire Target Detection; YOLOv5; CBAM; Depth Separable Convolution; VariFocal Loss

I. INTRODUCTION

Forests are one of the most important ecosystems on Earth. They provide species diversity and support the survival and reproduction of a wide variety of plants, animals and insects. It plays an irreplaceable special role in the balance of the earth's ecosystem, climate regulation, resource protection, and economic development. If humans cherish and protect forests and sustainably use forest resources, forests will give back to humans with long-term ecological and economic value. Fire occurs artificially or naturally in the forest. It will spread uncontrollably and gradually develop into a disaster. It will not only cause immeasurable permanent damage to various resources and properties in the forest but also cause immeasurable permanent damage to humans and other people living in the surrounding area. A huge threat to the life safety of living things.

The type of fire is inseparable from the firefighting strategy. The focus of urban building fires is on controlling the fire and rescuing trapped people. Due to the speed of its spread, the breadth of its scope, and the huge difference between firefighting resource supply and firefighting demand, firefighting in forest scenes focuses on early detection and prevention of fires. Rapid detection of flame signs will be an important measure to prevent forest fires and respond to existing fires. However, because there are many types of fire scenes and the internal conditions are complex, relying solely on fire rescue personnel to screen fire scenes with harsh conditions has many uncertain and limiting factors. Therefore, we can use the camera equipment carried by individual firefighters or obtain information about the fire scene through other channels. The situation is transmitted back to the fire command headquarters for processing, allowing for a further comprehensive and in-depth understanding of the fire situation. Fire scene information perception and interaction are the foundation and premise of firefighting and are directly related to the depth and breadth of digital applications in fire scenes. Once the knowledge and understanding of fire scene information is lost, the fire command department will lose the ability to coordinate and

DOI: 10.2478/ijanmc-2023-0080

plan firefighting operations from a high position when a large fire occurs. Therefore, this paper applies digitization to forest fire scenes, utilizing computers to analyze and process forest fire images, thereby reducing the manual analysis workload in firefighting activities. This approach enables timely and efficient detection of fires in the early stages for prompt alarm and response. It also fulfills the requirement of promptly locating and initiating targeted firefighting actions during the development of forest fires. This enhances the external perception of internal conditions at the fire scene. Furthermore, based on the situation at the fire scene, it facilitates task-driven scheduling assistance for individual soldier cooperation, thereby achieving the requirement for organized, efficient, and rational firefighting and disaster relief operations.

II. RELATED WORK

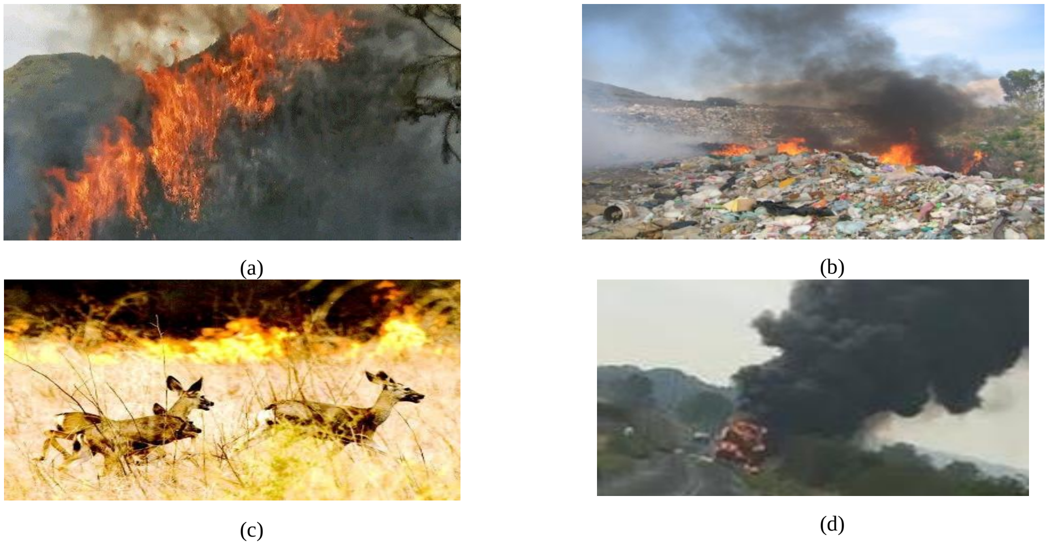
All Fire, generated and spread by humans or nature in forests, can become a disaster [1]. It not only causes incalculable permanent damage to various resources and properties in the forest, but also poses a huge threat to the safety of human and other living beings living in the surrounding areas. This article applies digitalization to forest fire scenes, using computers to analyze and process forest fire scene images, reducing the workload of manual analysis in firefighting behavior, enabling timely and efficient detection in the early stages of fires [2], and providing timely alarms and responses, strengthening the external perception ability of the fire scene for internal conditions, and achieving the requirements of organizing and commanding orderly, efficient, and reasonable firefighting and disaster relief work.

Traditional forest fire detection relies on image processing techniques of classical computer vision to analyze and process the features of flame targets in images, including extracting edge features [3], texture instability and similarity analysis [4], foreground features [5], background modeling [6], and flame color analysis [7]. These classic algorithms have good detection performance, but there are drawbacks such as poor generalization ability [8] and slow detection speed [9]. Based on the characteristics of different fire scenarios, people choose different classic network models and develop various improvement plans for them. Reference [10] improves the GMM algorithm by fusing texture and similarity feature information of different colors in the image, but its learning ability for nonlinear change features is limited. Using depthwise separable convolution and CBAM to form a depthwise separable attention module, a new semantic segmentation network is formed [11], and the fusion multi-scale improved FRCNN [12] method results in slower model processing speed. Building deep neural networks to learn data representation and feature extraction has gradually become a trend in researching fire detection. YOLO has become one of the most optimal object detection algorithms at present. There are improvement options to choose YOLOv3, combined with the CAM [13], or in the detection output module, the improved K-means algorithm optimizes the prior box [14], or adds a variable convolution module [15]. The stability of model accuracy is greatly affected by the environment. On the basis of YOLOv4 network, there are methods such as color enhancement [16], introduction of attention mechanism and residual structure [17], which cannot reduce false positives in certain situations. In YOLOv5, the Neck module was introduced into the weighted bidirectional feature pyramid network [18] to replace the original path aggregation network, transfer learning [19] was adopted to train the model, and the Focal loss function [20] was introduced. The SPP structure was changed by a better performing SPPF structure [21], but the processing speed of the model still cannot meet the timeliness requirements of forest fire detection.

In response to the above issues, this article puts forward an improved fire detection model based on YOLOv5. This model will introduce the attention module dsCBAM, which replaces ordinary convolutions with depthwise separable convolutions, into the backbone network responsible for feature extraction in the YOLOv5 algorithm. This will improve the inference speed of the model and significantly improve its convergence speed. At the same time, the model has advantages in both representation ability in regions of interest and detection robustness in diverse environments.

III. DESIGN OF FIRE DETECTION MODEL

*A. D*atasets



(a) (b)

(c) (d)

Figure 1. Part of the image data used for training: (a) Fire with poor resolution. (b) Fire in a small area. (c) Fire with flame obstruction. (d) Fire disturbed by smoke.

Since forest fire images cannot be collected and reproduced through experiments, the scene image data are public forest fire scene image data crawled on the Internet, and are collected and simulated by some enterprises and related research institutes to create publicly available data sets. Fire images are used to assist. Taking into account the diversity of real fire situations, images will contain different scenes, weather conditions, and fire intensity. Due to different image data acquisition channels and shooting conditions, image data may encounter several different recognition difficulties as shown in Figure1, such as poor image resolution, small fire range, flame obstruction, smoke interference, etc. The dataset applied to the model in this article is manually selected to filter out images that are not suitable for training in individual extreme cases. Then use labelimg software to label the flame area in VOC data set format using suitable image data.

B. YOLOv5s

YOLO is a classic target detection algorithm known for its efficient real-time detection. In fire detection tasks with high time-efficiency requirements, the YOLO algorithm can meet the need for rapid identification. Currently, the more mature version of YOLO is YOLOv5, and YOLOv5 is divided into s, m, l, and x, according to the complexity of the model. This article will conduct research and discussion based on the lightest YOLOv 5s 6.0 version.

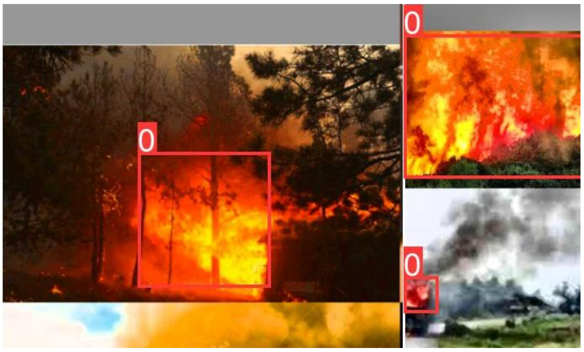


Figure 2. YOLOv 5 input data enhancement method

On the data preprocessing part, YOLOv5s follows the Mosaic method. As shown in Figure 2, it will apply random scaling, random cropping, and random arrangement to the input images for splicing. The processed data is more accurate in detection. Sexuality and discernment abilities are enhanced. Secondly, the difference from 4 version is that the former backbone network apply the single CSP. Two tiny altering structure of CSP are applied in the 5th version. The backbone part adopts CSP1_X and neck part adopts CSP2_X. Normally, the backbone part of YOLOv5s implements by CBL, C3, and SPPF modules to stack in the Neck part. This can help YOLOv5 better handle targets of different sizes, improve network feature fusion capability, and improve detection performance. Finally, YOLOv5's head output network calculates box and class probability of the target. The head network consists of multiple convolutional layers. It uses threshold filtering and NMS to obtain the final detection result and output the target prediction result. Different from the YOLO algorithm proposed in the previous sequence, YOLOv5 adopts a more lightweight convolution structure, which reduces the amount of calculation and maintains good accuracy.

C. VariFocal Loss

The computing method measures the disparity between the label predicted by the neural network and the expected true label to a certain extent. A good loss function will have a positive impact on the training process and final results of the neural network. YOLOv5 uses a loss function called CIoU (Complete Intersection over Union) to optimize the target detection task. The CIoU loss function takes into account the degree of overlap between the predicted boundary and the real bounding box and optimizes the positioning of the target more accurately. Generally speaking, in practice, the target to be detected in the training image data, that is, the positive sample, only accounts for a small part of the image, especially image data such as a forest fire scene that contains many small flame targets, and most of the area is the background. , constitute the negative samples during training; this will lead to a large number of negative samples in the training data, while the positive samples will account for a relatively small proportion, and the training effect of the model will become worse. Normally, background class negative samples are generally easy-to-separate samples, while target class positive samples are difficult-to-separate samples. As shown in (1), in order to solve the problem of uneven distribution of the two samples, Focal Loss adds weight factors to the samples that α are difficult to separate and those that are easy to separate, increasing the weight of the difficult-to-separate samples and reducing the weight of the easy-to-separate samples, thereby controlling the positive The problem of too large gap between negative samples. Among them, in (1) α Represents balanced weight, $\left(1-\mathrm{p}\right)^{\gamma}$ is a regulatory factor, γ is an adjustable focusing parameter. Therefore, Focal Loss is suitable for detecting image data of dense targets, and has good effects on data sets with characteristics such as small size, crowding, and occlusion.

$$FL\left(p,y\right)=\begin{cases}-\alpha\left(1-\mathrm{p}\right)^{\gamma}\log(p), y=1\\ -\left(1-\mathrm{p}\right)^{\gamma}\log(1-p), others\end{cases} \quad (1)$$

VariFocal Loss is proposed on the basis of Focal Loss, because Focal Loss processes positive and negative samples in a balanced manner, while VariFocal loss only reduces the loss contribution of negative samples without reducing the weight of positive samples in the same way. As shown in (2), the main improvement of VariFocal Loss lies in the introduction of parameter controlled weights for target classification loss, where p is the predicted value of IoU aware classification score (IACS), α and γ is an adjustable scaling factor, and q is a positive sample growth parameter. When it is a negative background sample, $q=0$; When it is the target positive sample, q is equal to the IoU between the generated bbox and the annotation box at that point. In the traditional CIoU loss function, the weights for target classification loss and target localization loss are fixed. VariFocal Loss introduces α and γ parameters to adaptively adjust the loss weights based on the difficulty level of different samples [22]. When the sample is more challenging, larger values of α and γ increase the weight of target classification loss, emphasizing classification accuracy. When the sample is less challenging, smaller values of α and γ decrease the weight of target classification loss, prioritizing localization accuracy and dynamically adjusting the weight of target classification loss. When the sample difficulty is greater, α the value of sum is larger, the weight of the target classification loss increases, and more attention is paid to the accuracy of the classification; when the sample difficulty is low, the value of sum is small α , γ . The γ weight of the target classification loss decreases, and more attention is paid to the accuracy of the classification Positioning accuracy, dynamically adjust the weight of the target classification loss. By introducing VariFocal Loss, the target detection model can better balance the trade-off between target classification and target positioning, thereby improving the performance of target detection. VariFocal Loss has been applied in some target detection algorithms.

$$VFL\left(p,y\right)=\begin{cases}-q\left(\mathrm{q}\log(p)+(1-\mathrm{q})\log(1-p)\right),q>0\\ -\alpha p^{\gamma}\log(1-p),q=0\end{cases}\quad(2)$$

D. CBAM

In the process of development, the attention mechanism derives various types of attention. According to different classifications of attention, such as multi-scale attention, contextual attention, parallel branch attention, channel attention, spatial attention, etc. Depending on the size of the attention scale, there are several more outstanding models, such as Transformer, SE, CBAM [23], and so on.

CBAM is one of the new lightweight attention modules used to enhance convolutional neural networks. Quantitative attention model convolutional block model. It does not directly calculate the attention map, but separates it, learns attention from channel and spatial respectively, adaptively learns the channel correlation and spatial importance of the input feature map, and simultaneously takes advantage of both to improve the accuracy of feature extraction. Accuracy. Considering the difficulty of fire scene transmission and the portability requirements of individual firefighters, the pixels of fire scene images are generally very low. When it is difficult to apply computer-related algorithms for identification and detection, false alarms and missed detections will occur. The CBAM module, as is shown in Figure 3, inputs the feature map $\mathrm{F},\mathrm{F}\in\mathrm{R}^{\mathrm{C}\times\mathrm{H}\times\mathrm{W}}$, processes it into a one-bit feature map through the channel attention module $\mathrm{F}',\mathrm{F}'\in\mathrm{R}^{\mathrm{C}\times1\times1}$, and then uses the spatial attention module to generate a two-dimensional spatial attention map $\mathrm{F}'',\mathrm{F}''\in\mathrm{R}^{1\times\mathrm{H}\times\mathrm{W}}$.

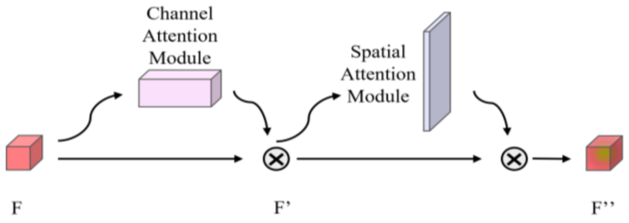


Figure 3. CBAM overall structure

The CAM part , in Figure 4, performs global average pooling and global maximum pooling operations on the input feature layer, and connects the two pooling results using a shared multi-layer perceptron. By performing a weighting operation on the original input feature layer channel-by-channel multiplication, the feature information of different levels of the upper-level output feature map can be extracted. The CAM adopts the global average pooling and global maximum pooling serial structures and combines the results of the two pooling methods to achieve compressed spatial dimensions of the input feature map, with stronger representational power.

The SAM structure, in Figure 5, focuses on which part of the input image information is more significant and is a complement to the CAM in the previous part. To calculate spatial attention, first apply average pooling and maximum pooling along the channel direction of each feature point, stack and aggregate them to generate the channel information of a feature map, and generate two two-bit feature maps. Spatial attention obtains the global maximum feature in the spatial dimension by performing global maximum pooling in the channel dimension and learns the weight of each spatial position through two fully connected layers. In this way, the model can automatically learn the importance of each spatial location, that is, which spatial locations are more important for target positioning.

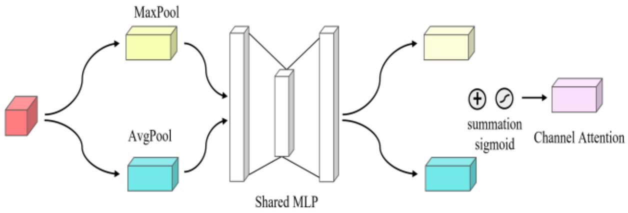


Figure 4. CAM structure

In the forest fire detection task, there may be different key channels for different fire types and backgrounds. Channel attention can help the model adaptively select channel information suitable for the current task, thereby reducing the interference of irrelevant information and improving feature representation. effectiveness. Fires usually appear at specific locations in images, and spatial attention can help the model focus on these important spatial locations and improve target positioning accuracy. By combining channel attention and spatial attention, the CBAM module enables the model to pay more attention to important channel information and spatial position information in the feature extraction stage, thereby enhancing the model's perceptual ability. In the forest fire detection task, CBAM can help the model better understand the correlation and importance of the input feature map, improve the model's detection and positioning capabilities of forest fire targets, and thereby improve the performance and robustness of the forest fire detection system.

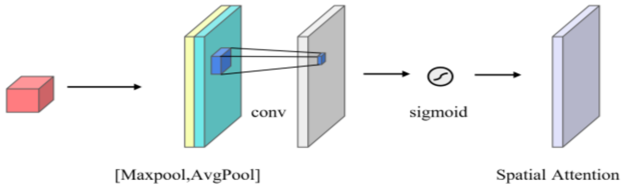


Figure 5. SAM structure

E. Lightweight Convolution

Ordinary convolution is shown in Figure 6. Assume that the number of input channels is M, the size is $D_F \times D_F$, the number of output channels is N, the convolution kernel size is $D_K \times D_K$, and the bias term is ignored b. Then, the amount of calculation required for this convolution operation is

$$Q_c = D_K \times D_K \times M \times N \times D_F \times D_F \qquad (3)$$

, the required parameters are shown in (4).

$$P_c = D_K \times D_K \times M \times N \qquad (4)$$

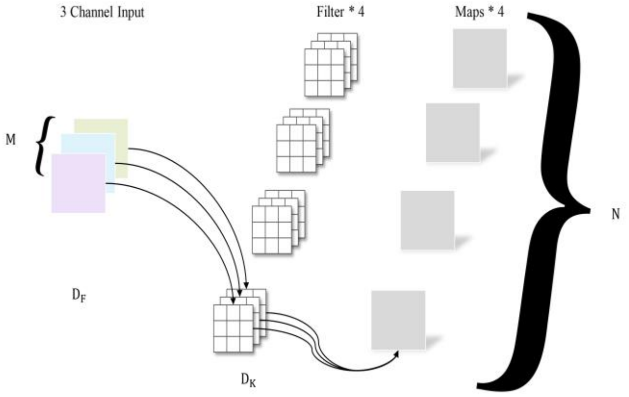
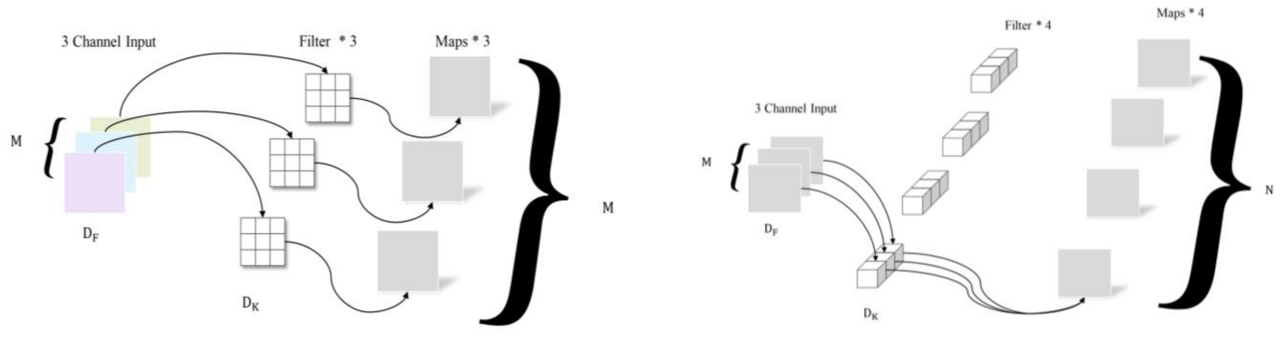


Figure 6. Ordinary convolution

The input feature map of the convolution is divided into g groups, each convolution kernel is also divided into groups accordingly, and the convolution operation is performed in the corresponding group. Each set of convolutions generates one feature map, and a total of g feature maps are generated. The number of groups g is like a control knob. The minimum value is 1, and $g=1$ the convolution at this time is ordinary convolution; the maximum value is the number of channels of the input feature map C, and $g=C$ the convolution at this time is depth separation convolution, also called channel-by-channel convolution.



(a)

(b)

Figure 7. Depth separation convolution: (a) Depth convolution. (b) Pointwise convolution

In other words, depthwise separative convolution is a special form of grouped convolution, where the number of groups is the number of channels of the feature map. That is, each feature map is divided into a group, and convolution is performed within the group. A convolution kernel in the group generates a feature map. This convolutional form is the most efficient form of convolution. Compared with ordinary convolution, multiple feature maps can be generated with the same amount of parameters and calculations, while ordinary convolution can only generate one feature map. Pointwise convolution is just 1×1an ordinary convolution. Because depth convolution does not integrate inter-channel information, it needs to be used in conjunction with point-by-point convolution. The operation of point-wise convolution is very similar to the conventional convolution operation. The size of its convolution kernel is $1\times 1\times \mathrm{M}$, M which is the number of channels of the previous layer. Therefore, the convolution operation here will weightedly combine the feature maps of the previous step in the depth direction to generate a new feature map.

Depth separable convolution is equivalent to Figure 7(a). Assuming that this convolution and the ordinary convolution above face the same feature weighting task, the corresponding calculation amount of the depth convolution is

$$\mathrm{Q_{dw}} = \mathrm{D_K} \times \mathrm{D_K} \times \mathrm{M} \times \mathrm{D_F} \times \mathrm{D_F} \tag{5}$$

The parameter quantity is

$$\mathrm{P_{dw}} = \mathrm{M} \times \mathrm{D_K} \times \mathrm{D_K}\,. \tag{6}$$

The corresponding calculation amount of pointwise convolution, as is shown in figure 7(b), is

$$\mathrm{Q_{pw}} = 1 \times \mathrm{M} \times \mathrm{N} \times \mathrm{D_F} \times \mathrm{D_F} \tag{7}$$

The parameter quantity is

$$\mathrm{P_{pw}} = \mathrm{M} \times \mathrm{N} \times 1 \tag{8}$$

Then the total calculation amount and parameter amount of the combined depth-separable convolution are the sum of the two, and the calculation amount is

$$\begin{aligned} \mathrm{Q_{ds}} &= \mathrm{Q_{dw}} + \mathrm{Q_{pw}} \\ &= \mathrm{D_K} \times \mathrm{D_K} \times \mathrm{M} \times \mathrm{D_F} \times \mathrm{D_F} + \mathrm{M} \times \mathrm{N} \times \mathrm{D_F} \times \mathrm{D_F}, \end{aligned} \tag{9}$$

The parameter quantity is

$$\begin{aligned} \mathrm{P_{ds}} &= \mathrm{P_{dw}} + \mathrm{P_{pw}} \\ &= \mathrm{D_K} \times \mathrm{D_K} \times \mathrm{M} + \mathrm{M} \times \mathrm{N} \times 1. \end{aligned} \tag{10}$$

Compute the calculation amount and calculation parameters of depthwise separable convolution and ordinary convolution, that is

$$\mathrm{Q_{ds}} / \mathrm{Q_c} = 1/\mathrm{N} + 1/\mathrm{D_k^2} \tag{11}$$

$$P_{ds} / P_c = 1/N + 1/D_k^2 \qquad (12)$$

From (11) (12), it can be seen that the calculation amount and parameter amount of the former are $1/N + 1/D_k^2$ times that of the latter. It shows that modified convolution reduces the required parameters and has reference significance in lightweight advanced models. In order to further lightweight the model, the convolution operation in the CBAM module can substitute modified convolution for the initial convolution method, which is referred to as dsCBAM in this article.

F. Improved Forest Fire Detection Algorithm

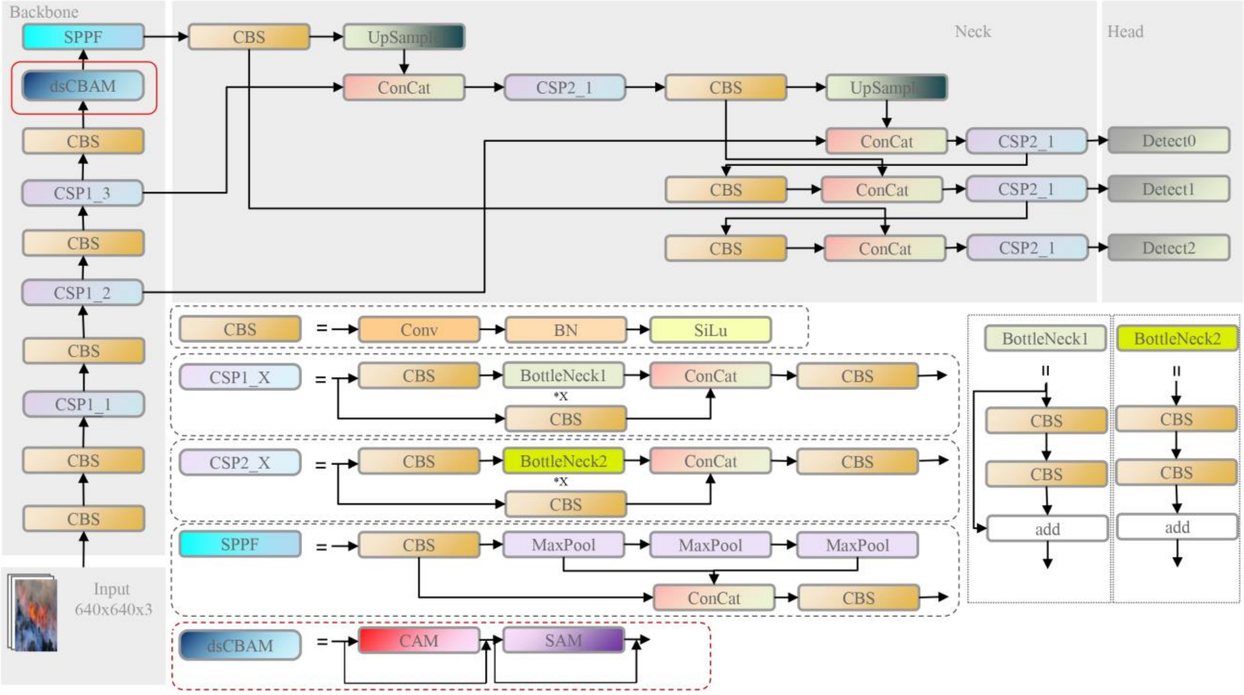


Figure 8. Improved model framework

The forest fire detection framework of this article is shown in Figure 8, which is an improvement based on the YOLOv5s model. In response to the real-time requirements of forest fires, the first step is to improve the lightweight attention model and replace the convolutions in the CBAM module with depth-separable convolutions. Targets usually occupy a small proportion of the screen, which may cause sample imbalance. Replacing the initial loss function with the mentioned earlier function can improve performance for this problem. The third step is to add the lightweight CBAM model to YOLOv5s to enhance the robustness of the forest fire detection system.

The dsCBAM module can adaptively adjust the channel and spatial information through the CAM and SAM, which helps the YOLO network better understand the target structure and contextual relationships in the image and enhance its ability to perceive fire fields. As shown in Figure 9, this article adds the CBAM module to YOLO to replace the last CSP 1_1 module in the original model. Replacing the CSP1_1 module with the CBAM module will enhance the model's ability to perceive targets at different scales, directions, and angles, thereby improving detection accuracy. The introduction of the CBAM module may help reduce noise or redundant information inside the prediction frame, make the target boundary clearer, enhance feature extraction and representation capabilities, and thus help improve detection quality. The CBAM module makes the structure of the backbone network richer and more diverse, making the network more robust to changes and disturbances in the input image, thereby increasing the generalization performance of the model.

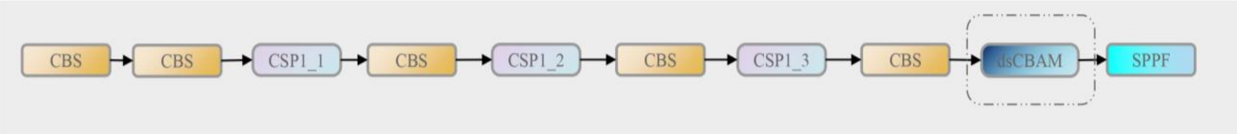


Figure 9. The position of CBAM in YOLOv5s 6.0 version

IV. RESULTS AND DISCUSSION

A. Training

TABLE I. DATASET SETTINGS

Dataset	Training	Test	Validation	Total
Homemade forest fire data set	1442	617	617	2676
Other institutes data set	600	200	200	1000

TABLE II. EXPERIMENTAL SETTINGS

Lab Environment	Detail
programming language	Python3.8.5
operating system	Windows 10
deep learning framework	Pytorch 1.8.0
GPU	4x NVIDIA TITIAN V

The data set needs to be manually screened one by one and the flame targets in it need to be labeled. As shown in Table 1. According to the needs of the experiment, the collected forest fire image data set was divided into a training set, a test set, and a verification set in a ratio of 6:2:2 to carry out experiments on the model.

TABLE III. TRAINING SETTINGS PARAMETERS

Training parameters	Detail
Epochs	100
Batch-size	16
Image-size	6 40 × 640
Initial learning rate	0.01
Optimization algorithm	SGD

The experimental settings of this experiment are shown in Table 2. It shows some parameter settings during the training process of the experiment. In this experiment, the training optimization algorithm uses the default stochastic gradient descent method (SGD). During training, the adaptive moment estimation (Adam) optimization algorithm can be selected according to the actual situation.

B. Model Evaluation

The evaluation index of the public data set Microsoft COCO is recognized as effective and state-of-the-art in the field of object detection. It is used in this article to evaluate the performance of the proposed improved forest fire detection algorithm. The five indicators of P, R, AP, mAP and FPS will be expanded below.

$$P = TP / (TP+FP) \tag{13}$$

P (Precision) refers to the ratio of correctly detected targets to all detection results in (15). Among them, TP (true positive) represents the predicted correct box. The boxes predicted by the model are calculated one by one with the labeled boxes of the image. R (Recall) refers to the proportion of truly detected targets to all real targets in (14).

$$R = TP/(TP+FN) \tag{14}$$

$$AP = \int_0^1 P\left(r\right) dr \tag{15}$$

AP (Average Precision) essentially describes the performance of the model on a single category. In the multi-category target detection task, each category has an AP value. The metric provide specific numerical values to measure the algorithm's prediction accuracy and target detection capabilities.

$$FPS = 1000 / (pre_{process} + inf + NMS) \tag{16}$$

$pre_{process}$ refers to the preprocessing time for converting the input image into the format required by the algorithm, including image aspect ratio scaling, padding, normalization and other operation times. Value inf refers to the inference time, that is, the forward pass calculation time from inputting the image into the model to the model output result after preprocessing. NMS It can be understood that post-processing time is mainly the time spent on converting the model output results and other operations. The sum of the three is the total time of image processing. After calculation by formula (16), FPS (Frame Per Second) is obtained. If tested and compared in the same hardware environment, the lightweight effect of the algorithm can be expressed to a certain extent.

C. Ablation Experiment

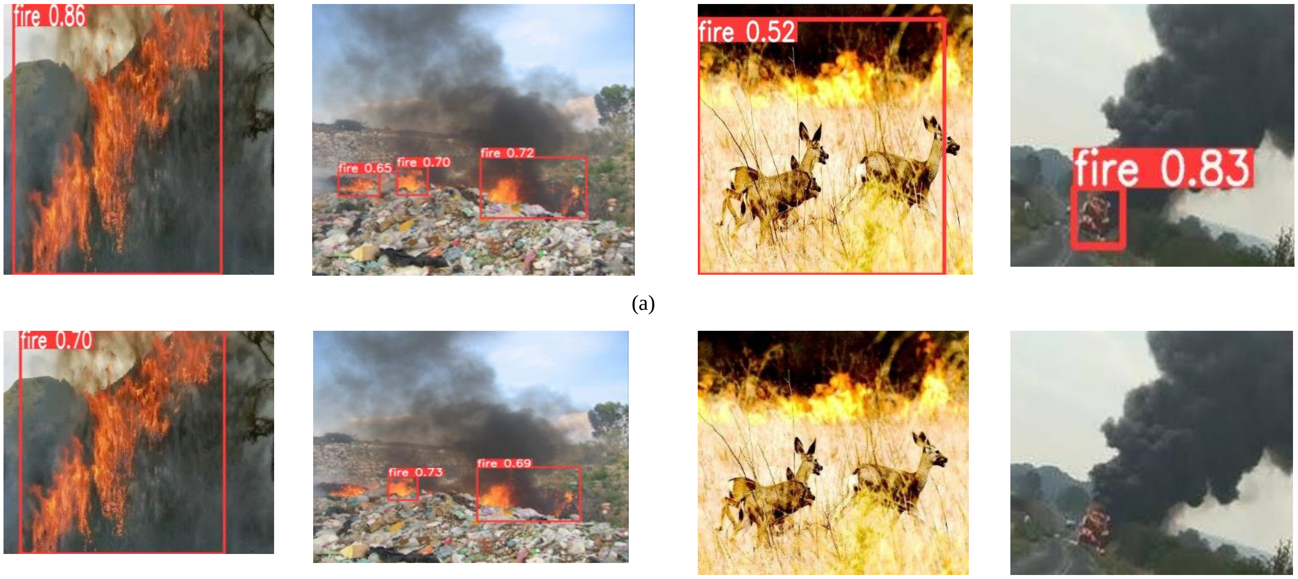
TableTABLE IV. presents the results of this experiment. The experiments were evaluated separately on the same data set, and eight solutions were compared horizontally, namely (1) original YOLOv5s model; (2) combination of CBAM and YOLOv5s model; (3) combination of SE and YOLOv5s model; (4) Combining ECA with YOLOv5s model; (5) Improving the model combining CBAM with YOLOv5s; (6) Improving the model combining CBAM with Alpha-IoU and YOLOv5s; (7) Improving the model combining CBAM with SIOU and YOLOv5s; (8) Improving CBAM with VariFocal Loss Combined model with YOLOv5s. They show that improvement methods are effective from the four indicators of accuracy P, recall rate R and frames per second (FPS). After adding the CBAM model, the original model's recognition accuracy of flame targets in forest fires has been slightly improved, and the model's ability to perceive flame targets has been further enhanced. By introducing depthwise separable convolution, the speed of data processing of the model is improved, and the degree of lightweight and portability of the model is deepened. Finally, through ablation experiments to compare the three loss functions of Alpha-IoU, SioU, and VariFocal, the loss function proposed in this article was selected as the loss function with the best performance, which verified the importance of suppressing negative samples in improving the performance of the target recognition algorithm. All in all, compared with the traditional YOLO algorithm, the improved model has achieved significant performance improvements in both the difficult detection task of small target detection and the speed of detection.

TABLE IV. COMPARATIVE TEST RESULTS OF THE MODEL

Model	P	R	FPS
YOLOv5s	0.811	0.786	59
YOLOv5s + CBAM	0.814	0.790	60
YOLOv5s + SE	0.810	0.787	5 8
YOLOv5s + ECA	0.812	0.791	5 9
YOLOv5s + dsCBAM	0.812	0.787	62
YOLOv5s + dsCBAM + Alpha-IoU	0.821	0.813	61
YOLOv5s + dsCBAM + SIoU	0.860	0.834	60
YOLOv5s + dsCBAM+ VariFocal （Ours）	0.871	0.816	64

D. Comparision

Figure 10 shows the cpmparative results of the original and the improved. By comparing the initial net and the proposed net to detect four groups of images, the detection results can more intuitively and objectively show that the improved model has better performance. The fire targets in the first set of images can be accurately detected, but the confidence of the improved model is significantly improved. There are three flame targets in the second set of images. The original model can only detect the two larger targets, while the improved model can detect all targets. The flames in the third group of images were blocked to a certain extent by foreground objects and could not be identified by the original model. The improved model accurately identified its location. The proportion of flames in the last set of images is relatively small, and the improved model solves the problem that the original model cannot detect. It can be observed that the improved model does not miss small fire targets and can detect fires more accurately even when the image quality and size are not very high Figures 11 present the robustness experiment, the initial net misclassified forest night lights as flames, while the improved model's attention mechanism enhanced the feature learning of the detection targets, thereby improving the occurrence of false detections.



(a)

(b)

Figure 10. Experimental identification results: (a) Improved model. (b) Original model.

By analyzing and comparing the experimental performance index data in Table 4, we can intuitively observe the robustness of all aspects of the model. Through the comparison of Experiments 1 to 4, we can see that after adding three different attention mechanisms: CBAM, SE, and ECA to the original YOLOv5s model, the results are affected to varying degrees. Among them, As to YOLOv5s combined with SE, the accuracy and FPS of the model have declined, while the recall rate has slightly improved. After the introduction of YOLOv5s in the ECA attention mechanism, the three indicators of P, R have slightly improved, and the model processing speed has almost no change. After the assistance of CBAM, P, R improved more significantly than ECA, and the growth of R was even better, indicating that it can detect more targets and reduce the missed detection problem. The processing speed is also not very high. Significant improvement. From the comparison, CBAM has the best improvement in focusing on small targets and detecting speed and is more in line with the requirements of diverse detection environments. The comparison between Experiment 2 and Experiment 5 directly shows the performance of applying the modified convolution. The experimental accuracy and recall rate result data show that the replacement strategy can shorten the processing time of each image based on ensuring the accuracy of the model, making the lightweight features of the model more prominent. Experiments 5-8 respectively completed the training, verification, and detection tasks of forest fire images by applying the improved CBAM and four different loss functions of the original loss function, Alpha-IoU, SIoU, and VariFocal. Comparing the experimental results of these four different loss functions on the training task, we can observe that the first replacement loss function has a small range of growth in the three indicators measuring model training, but the addition also affects the processing time of the model. Compared with the experimental performance of the SIOU loss and the loss used in Experiments 5 and 6, the training accuracy on the data set has been significantly improved, but this also makes the model pay the price of processing speed. Experiment 8 is the experimental data based on the improvement points proposed in this article. It has good adaptability to changes in input images, lighting conditions, occlusions, etc., and also takes into account the processing speed of the model, so that performance and efficiency are balanced to a certain extent.

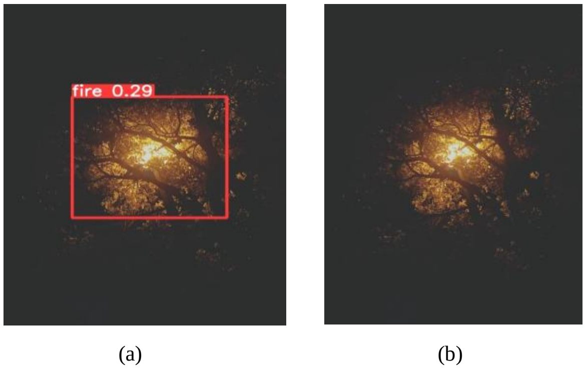


Figure 11. Experimental Experimental results of misdetection of forest street lights at night. (a) Original model. (b) Improved model.

E. Discussion

In this study, we apply CBAM, VariFocal, depthwise separable convolution, and YOLOv5s to the forest fire detection task. By comparing experimental results, we observe that the model achieves significant performance improvements on multiple metrics. First, the CBAM module helps improve the model's attention to key areas, allowing for better detection of features. Secondly, the VariFocal loss function introduces dynamic weight allocation. Furthermore, depthwise separable convolution reduces the computational effort and maintains model performance. Through experiments, we observed that the improved models of CBAM, VariFocal, depthwise separable convolution, and YOLOv5 are highly robust when processing forest fire images in different scenarios, and the model can accurately detect various scales, poses, and density of fire targets, and also has certain adaptability to images under different lighting conditions. Compared with other deep learning-based methods, our method achieves faster inference speed while maintaining high accuracy. Although the improved models of CBAM, VariFocal, depthwise separable convolution, and YOLOv5 achieved good results in the forest fire detection task, there are still some limitations. For example, a model may perform poorly when dealing with low-resolution or blurry images. In addition, the robustness of the model in complex scenarios still needs to be further improved. Future work may need to consider combining multi-modal data, introducing a target tracking module.

V. CONCLUSIONS

This study conducted an in-depth study on the forest fire detection task by applying improved methods of CBAM, VariFocal, depthwise separable convolution, and YOLOv5s, introduced the working principle and working method of the original model, and deeply an alyzed the principles and possible improvements of various improvements. Achievability. For forest fire detection tasks, the assistance of the CBAM structure helps to improve detection model's focus on key areas and the detection accuracy of fire targets. Its mechanism based on channel attention and spatial attention can effectively extract fire features such as flames and smoke in images. The assistance of the advanced loss function is able to overcome imbalance of sample category and present better results. This loss function uses dynamic weight allocation to make the model pay more attention to minority class samples, thereby improving detection accuracy. The application of depthwise separable convolution reduces the computational load of the model while maintaining model performance. This lightweight convolution operation helps improve the running efficiency of the model, making it more suitable for practical fire detection applications. Improvements in YOLOv5 show good performance in forest fire detection. Its fast and accurate target detection capabilities enable the model to monitor forest areas in real-time and detect the occurrence of fires promptly, thus providing the opportunity for rapid response and processing. Experiments are conducted to demonstrate the actual performance of various improvement ideas. Experiments on forest fire detection tasks have proven that the improved method in this article effectively enhances the perception ability of the original YOLO model and achieves good results. The accuracy rate is improved by 0.06 based on the original model, and the number of frames processed per second is 3 frames has been added, which greatly improves the accuracy and efficiency of forest fire detection. The results of this study provide an important reference and foundation for further research and development in the field of forest fire detection. By improving existing models and technologies, we can improve our monitoring and early warning capabilities for forest fires, thereby reducing the harm of fires to the environment and humans. Future work can explore more deep learning methods and technologies, integrate multi-source data, and enhance the robustness and real-time performance of the algorithm to further advance the development of forest fire detection technology. In summary, the results of this study provide useful exploration for research and practical applications in the field of forest fire detection and demonstrate the potential of CBAM, VariFocal, depthwise separable convolution, and YOLOv5 improved models in fire detection. It is hoped that this article can provide guidance for the prevention and control of forest fires and reduce the harm of fires to the natural environment and human society.

References

[1] Liao Shujiang. A preliminary study on the trend of fire spread [J]. Fire Science and Technology, 2012, 31(7): 670-673.

[2] Wang M. Risk Information, Risk Perception and Fire Prevention Behavior [D]. University of Science and Technology of China, 2017.

[3] Lv P T, Li J, Wu L Y et al. Research on automatic edge detection of fire video images [J]. Applied Science. 2003.

[4] YAN Yunyang, GAO Shangbing, GUO Zhibo, et al. Automatic fire detection based on video images [J]. Computer Application Research, 2008, 25(4): 1075-1078.

[5] TAN Yong, XIE Linbai, FENG Hongwei, et al. Image-based flame detection algorithm [J]. Laser & Optoelectronics Progress, 2019, 56(16): 161012.

[6] CUI Bingcheng, CHENG Naiwei, ZHAO Peng. Exploration of smoke image detection method based on matlab [J]. Science and Technology Innovation. 2019, (28).

[7] Gong F, Li C, Gong W, et al. A real-time fire detection method from video with multifeature fusion [J]. Computational intelligence and neuroscience, 2019, 2019.

[8] Li P, Zhao W. Image fire detection algorithms based on convolutional neural networks [J]. Case Studies in Thermal Engineering, 2020, 19: 100625.

[9] Saeed F, Paul A, Karthigaikumar P, et al. Convolutional neural network based early fire detection [J]. Multimedia Tools and Applications, 2020, 79: 9083-9099.

[10] ZHANG Chi, MENG Qinghao, WELL Tao. Video flame detection algorithm based on improved GMM and multi-feature fusion [J]. Laser & Optoelectronics Progress, 2021, 58(4): 0410006.

[11] Jing K, Jia Y, Zhang C, et al. MobileAttentionNet: An Efficient Network for Semantic Segmentation of Forest Fire Images [C]//2021 6th International Symposium on Computer and Information Processing Technology (ISCIPT). IEEE, 2021: 377-380.

[12] Zhang L, Wang M, Ding Y, et al. MS-FRCNN: A Multi-Scale Faster RCNN Model for Small Target Forest Fire Detection [J]. Forests, 2023, 14(3): 616.

[13] Zhang X, Qian K, Jing K, et al. Fire detection based on convolutional neural networks with channel attention [C]//2020 Chinese Automation Congress (CAC). IEEE, 2020: 3080-3085.

[14] ZHAO Yuanyuan, ZHU Jun, XIE Yakun, et al. Improved Yolo-v3 algorithm for real-time flame detection in video images [J]. Journal of Wuhan University (Information Science Edition), 2021, 46(3): 326-334.

[15] DING Hao, WANG Huiqin, WANG Ke. Improved YOLOv3 flame detection algorithm based on dynamic shape feature extraction and enhancement [J]. Laser & Optoelectronics Progress, 2022, 59(24):2410003-2410003-9.

[16] Avazov K, Mukhiddinov M, Makhmudov F, et al. Fire detection method in smart city environments using a deep-learning-based approach [J]. Electronics, 2021, 11(1): 73.

[17] Sun J, Ge H, Zhang Z. AS-YOLO: An improved YOLOv4 based on attention mechanism and SqueezeNet for person detection [C]//2021 IEEE 5th Advanced Information Technology, Electronic and Automation Control Conference (IAEAC). IEEE, 2021, 5: 1451-1456.

[18] Xue Q, Lin H, Wang F. Fcdm: an improved forest fire classification and detection model based on yolov5 [J]. Forests, 2022, 13(12): 2129.

[19] Xue Z, Lin H, Wang F. A small target forest fire detection model based on YOLOv5 improvement[J]. Forests, 2022, 13(8): 1332.

[20] Yang T, Xu S, Li W, et al. A smoke and flame detection method using an improved yolov5 algorithm [C]//2022 IEEE International Conference on Real-time Computing and Robotics (RCAR). IEEE, 2022: 366-371.

[21] Yang X, Wang Z, He Y, et al. Research on open flame recognition algorithm in construction site based on attention mechanism [C]//2023 15th International Conference on Advanced Computational Intelligence (ICACI). IEEE, 2023: 1-6.

[22] Zhang H, Wang Y, Dayoub F, et al. VariFocalnet: An iou-aware dense object detector [C]//Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. 2021: 8514-8523.

[23] Woo S, Park J, Lee J Y, et al. Cbam: Convolutional block attention module [C]//Proceedings of the European conference on computer vision (ECCV). 2018: 3-19.