Informed Object Detection for Computer Games

Anton Dubovskoi[1][0009−0003−4912−2877], Maksim Sokolov[1], and Ildar Baimuratov[2,3][0000−0002−6573−131X]

[1] ITMO University, Saint-Petersburg, Russia
netfeel95142@gmail.com, naymoll@yandex.ru
[2] Leibniz University Hannover, Hannover, Germany
[3] TIB - Leibniz Information Centre for Science and Technology, Hannover, Germany
baimuratov.i@gmail.com

Abstract. One of the traditional areas of artificial intelligence application is games. Today, solutions for this industry are developed mostly by companies that cooperate with developers and have access to game APIs. Not all researchers have the ability to gain such access, which can limit the progress of AI. One way around this limitation could be to apply computer vision to an image on the screen reflecting what is happening in the game. However, most current computer vision techniques use labeled data. Annotation requires a large amount of labor, time, and has a high cost. The problem of training data is even more acute for computer games, because modern games periodically receive updates and add-ons. To reduce the need for labeled data, we find it promising to use a hybrid approach that combines machine learning with knowledge representation. Thus, the goal of this research is to develop a hybrid method for object recognition in computer games, which reduces the need for labeled data but has accuracy comparable to end-to-end methods. This paper presents an overview of AI applications in computer games and hybrid computer vision methods. A model of object recognition using informed machine learning was developed. The game Hearthstone was chosen as the application and the task of detecting and classifying the cards present on the table was considered. Experiments have shown that the model provides higher accuracy than the models of the YOLO family while using a simpler annotation during training.

Keywords: Hybrid AI · Informed Machine Learning · Computer Vision · Object Recognition · Computer Games.

1 Introduction

One of the traditional and relevant areas of artificial intelligence application is games. Well-known examples are Deep Blue [2] from IBM, OpenAI Five[4] from OpenAI, AlphaGo [19] and AlphaStar [21] from Google DeepMind, etc. As you can see, today's existing solutions are mainly developed by large companies that collaborate with developers and have access to game APIs. Not all researchers

[4] https://openai.com/research/openai-five

have the ability to gain such access, which can limit progress in the field of AI. One way around this limitation could be to apply computer vision to an image on the screen reflecting what is happening in the game, and then recognize game objects on it.

However, most modern computer vision methods for classifying objects in images use supervised learning, which requires a large amount of training data. The preparation of such training data, in turn, requires a large amount of labor, time, and has a high cost. Thus, there is a problem of creating specialized datasets for each task. This problem is partially solved by transfer learning, but the models still need to be fine-tuned on task-specific data to obtain high-accuracy predictions. When applied to computer games, the problem of training data is even more acute as modern computer games periodically receive various updates and add-ons. Consequently, after each such change, new sets of training data would have to be produced in order to improve the efficiency of object recognition.

To reduce the need for labeled data, we find it promising to use a hybrid approach that combines machine learning with knowledge representation. One such method is Informed Machine Learning (IML) – the process of incorporating prior knowledge from the subject domain into machine learning algorithms and models [17]. Thus, the goal of this study is to develop a method for object recognition in computer games based on informed machine learning, which reduces the need for labeled data, but has accuracy comparable to classical methods.

In this study, the computer game of Blizzard Ent., Hearthstone[5], was chosen as the use case as it is an eSports discipline with the largest number of users in its genre, which also has a sufficient number of open resources and documentation necessary for the use of IML. We consider the problem of detecting and classifying the cards present on the playing field. We use models from the YOLO [11] family to recognize objects. As prior knowledge, we consider a set of individual cards from a site that provides complete and up-to-date information on Hearthstone HSReplay.net[6]. We have collected a dataset based on screenshots of the game and annotated it in two ways to train and evaluate the developed model. The first method involves annotating object boundaries only and is used to train the hybrid model, while the second method also involves annotating object classes. We compare the accuracy of the developed hybrid algorithm with the results of the end-to-end model trained on the created dataset with the second annotating method. The contribution of this study is an IML-based object recognition algorithm that uses only object boundary annotation, but has an accuracy that exceeds the accuracy of the end-to-end model.

2 Related Work

2.1 AI in Games

Perhaps one of the most famous examples of artificial intelligence in games is the chess supercomputer Deep Blue [2], developed by IBM, which beat world chess

[5] https://hearthstone.blizzard.com/
[6] https://hsreplay.net/

champion Garry Kasparov in 1997. The AlphaGo program, developed by Google DeepMind in 2015 [19] was able to beat a professional go player. In addition, a prime example of using artificial intelligence in games is OpenAI Five by OpenAI[7], which plays Dota 2[8]. This program was able to defeat a professional player in 2017. Google DeepMind partnered with Blizzard Entertainment[9] to create an artificial intelligence to play StarCraft 2[10] called AlphaStar which took the highest in-game rank in 2019 [21].

There are also examples of applications of *computer vision* in games. For example, [4] describes the development of computer opponents (bots) using computer vision in a racing game. To train the bots, an open-source computer vision library, OpenCV, is used. Paper [13] describes the concept of a framework and scripting programming language for writing "profiles" to games. Profiles describe the rules by which various game events should be read and responded to. The system includes functions that respond to, for example, the pressing of a key or the appearance of a new frame, as well as a visualization and machine learning (OpenCV) module. Article [1] describes an algorithm for extracting key situations and predicting match results in the game StarCraft 2 based on convolutional neural networks. Match replays are used as input to the model. Paper [14] describes a game state classification model. Sets of coordinates of game entities transformed into a set of sequential images are used as inputs. The RCNN model was used for the classification problem. Paper [18] describes the use of video games to train computer vision models. In particular, to recognize and annotate road objects and urban infrastructure. Article [6] studies the behavior of non-player characters (NPCs) and other objects in games based on observing only graphical output using computer vision techniques. Article [16] describes a plug-in for the Unreal Engine 4 (UE4) game engine. This plugin allows you to visualize the information needed for computer vision right during the game. Paper [9] describes using Double Deep Q-Network (DDQN) to play Super Mario Bros. DDQN includes three convolutional layers for image processing. Article [12] describes the creation of an agent for first-person shooter games. The input data used is frames from the game DOOM. The authors have modified the ViZDoom engine so that you can get information about the location of certain objects in the current frame.

Obviously, the most convenient approach for the application of AI in games is the use of API, which can provide accurate and complete information that is not available in the analysis of frames from the game. However, it is not always possible to interact directly with the game.

2.2 Hybrid AI

Hybrid intelligent systems are systems that use both machine learning and knowledge representation methods in parallel. The knowledge representation

[7] https://openai.com/
[8] https://www.dota2.com/
[9] https://www.blizzard.com/
[10] https://starcraft2.com/

methods include, for example, expert systems, logics, association rules, etc. Combining different methods of artificial intelligence in some cases allows to provide the best solution, which cannot be obtained by separate methods.

One approach to hybrid AI is Informed Machine Learning (IML). A comprehensive overview of the field of IML is given in [17]. It also provides a definition of the term and a variety of uses for the method. IML is the process of incorporating prior domain knowledge into machine learning algorithms and models to guide algorithm decisions and improve overall performance. Prior knowledge can be either a regular unstructured data set or knowledge graphs. They can be integrated in a variety of ways depending on the specific problem and the information available. Figure 1 from [17] shows a generalized IML application diagram. It demonstrates that in addition to basic information (the "Data" block), prior knowledge (the "Prior Knowledge" block) is integrated into the machine learning process.

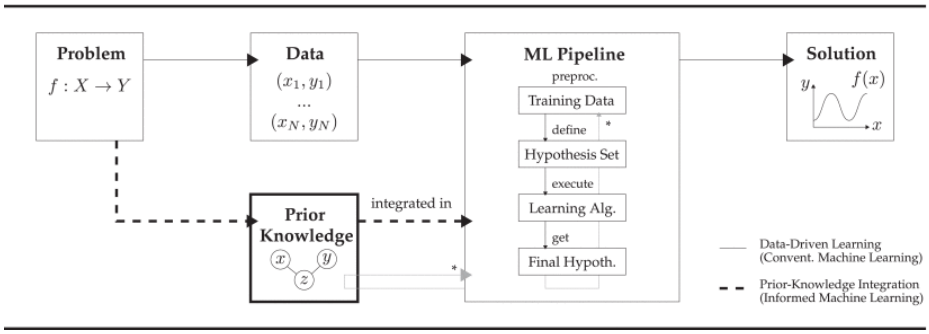

Fig. 1. Generalized IML application chart [17]

There are works that use a hybrid approach for computer vision, but they do not consider video games. In [22], the authors propose an image knowledge graph model that incorporates semantic relations between objects in the image and scene relations. This model combines several popular convolutional neural network architectures to classify objects in images followed by ontology generation. Subsequently, an algorithm is used to improve the classification of objects in images using the relations that have emerged in the extracted knowledge graph.

In [10], a new framework for Knowledge Graph Representation Fusion (KGRF) is proposed to introduce prior knowledge in the image classification problem. Specifically, it uses graph attention network (GAT) to extract knowledge representations from the constructed knowledge graph, CB-CNN [8] to extract objects from images, and Multimodal Compact Bilinear (MCB) module [7] to combine information from the knowledge graph and the extracted objects.

Article [5] proposes an approach to improve the classification of images using ontologies. The system described in the paper uses the HMAX model to extract

objects from images, WordNet to create an ontology, and the OWL API[11] to complete it.

Zhang el al. [23] investigate the effect of ontologies on image classification. A two-level ontology (semantic ontology and visual ontology) is constructed to hierarchically organize a large number of image classes. Semantic ontology is built according to semantic similarity between classes using WordNet[12], and visual ontology is built according to cross-class visual similarity using deep features. Deep features are extracted by the Inception V3 [20] model.

Monka et al. [15] provide a broad overview of knowledge graph embedding methods and describe several learning objectives suitable for combining them with visual embeddings. The paper provides answers to questions such as: How can a knowledge graph be combined with a deep learning pipeline? What are the properties of the respective combinations? What knowledge graphs already exist that can be used as auxiliary knowledge? What datasets exist that can be used in combination with auxiliary knowledge to evaluate visual transfer learning?

The article by Ding et al. [3] considers the application of ontologies in image object recognition. It is found that the combination of ontology and traditional image recognition technology can improve the recognition accuracy, enhance the ability of high-level semantic recognition, reduce the need for a large number of training samples, and improve the scalability of image recognition system.

3 Method

The developed method can be divided into two consecutive steps: 1) object boundary recognition and 2) class detection. Figure 2 shows the general scheme of the algorithm.

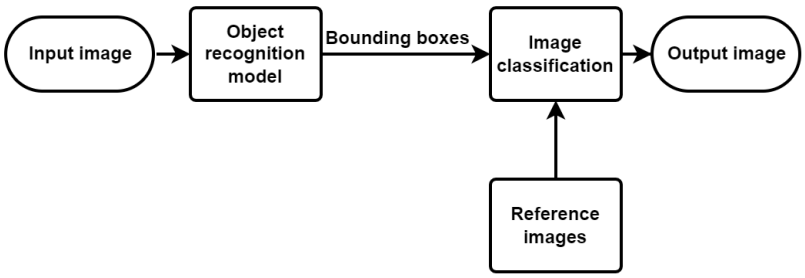

Fig. 2. General algorithm scheme

In the first step, object boundary detection is performed using a machine learning model. The model is trained on the dataset to recognize object bound-

[11] https://github.com/owlcs/owlapi
[12] https://wordnet.princeton.edu/

aries without specifying a particular class. An input image is fed to the model. The result of this step is a modified original image along with the object boundary data that the model was able to detect.

The second step involves the image classification module. The module receives the boundaries of the objects detected at the previous stage. In addition to boundaries, the module uses a set of reference images of objects, each of which corresponds to one of the classes by which classification is performed. For each passed boundary, the most similar image from the reference set is determined along with its corresponding class, using certain algorithms, such as the nearest neighbor method and feature matching.

Key points are characteristic areas of the image that can be matched with other images that contain similar elements. The search for key points is performed using the Scale-Invariant Feature Transform (SIFT) algorithm. The search method is as follows:

1. Scale-space Extrema Detection. Scale-space filtering is used to find potential key points.
2. Keypoint Localization. Any low-contrast keypoints and edge keypoints are removed, leaving only points of strong interest.
3. Orientation Assignment. Each keypoint is assigned an orientation to achieve rotation invariance.
4. Keypoint Descriptor. A keypoint descriptor is created.
5. Keypoint Matching. Keypoints between two images are matched by determining their nearest neighbors.

The whole classification process is described in Algorithm 1, where R – set of reference images, L – set of class labels, $M \subset R \times L$ – mapping from reference images to labels, b – detected box, i – detection image.

Algorithm 1 Classification algorithm

Require: R, M, b, i
  $c \leftarrow crop(i, b)$ ▷ Cropping the image by the detected box
  for $r \in R$ **do**
    if r in c **then** ▷ Image matching with CV
      $l \leftarrow M(r)$ ▷ Getting a class by the reference image
      $i \leftarrow l$ ▷ Assigning the class to the image
    end if
  end for

Dividing the algorithm into these two stages allows a modular approach to object recognition and computer vision tasks. This provides an efficient execution of the complex process of object detection and classification, combining the capabilities of machine learning with the versatility of computer vision methods. It is worth noting that the speed of the proposed method depends linearly on the size of the reference collection of images for comparison. The larger this collection is, the more comparisons need to be made.

4 Evaluation

4.1 Data

The computer game of Blizzard Ent., Hearthstone was chosen as the use case. The task of detecting and classifying the cards present on the playing field is considered. Given the huge number of cards in the game, it was decided to set certain constraints in order to simplify the data collection process. Therefore, only cards from the classic and standard collections were used. This limited selection provides sufficient variety of card types and visual features to develop a reliable model capable of accurately classifying the cards on the playing field.

We created and annotated three sets of data. Roboflow[13] was used as an annotation tool. The first set includes 274 images, 836 annotations, and 142 unique card classes. Figure 3 shows an example from the first set. The images are taken directly from the game, with a resolution of 1920x1080 pixels, which is the most popular among players. It ensures that the resulting images contain enough detail and maintains the integrity of the visual information.

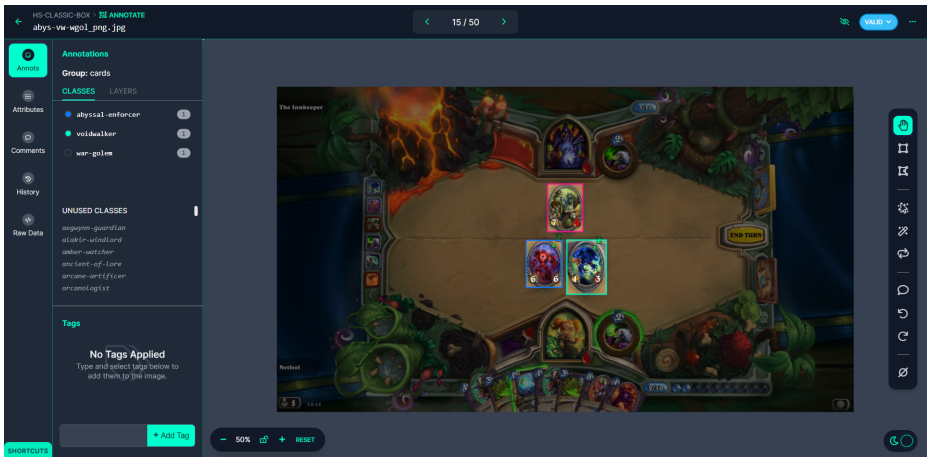

Fig. 3. An image from the first dataset

The second dataset includes the same 274 images as the first, but the annotation contains only 1 instead of 142 classes. This single class is an indication of the card's presence on the playing field, while excluding any specific information about the name of the card itself. This format requires significantly less intellectual effort from an annotator, speeds up the annotation process, and reduces the likelihood of errors when determining the class. Figure 4 shows an example of image from the second dataset.

[13] https://app.roboflow.com/

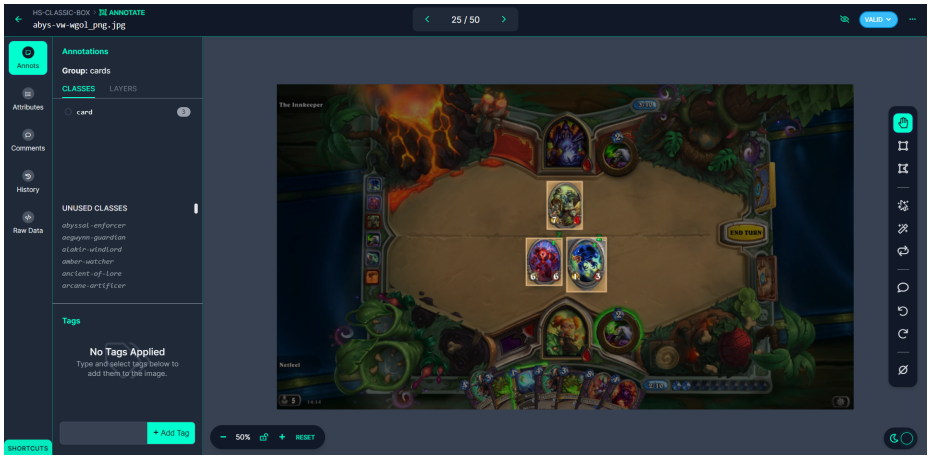

Fig. 4. An image from the second dataset

The third dataset is a reference dataset of 142 images representing unique cards. These images do not require annotations, but the file name must include either the name or a specific label that references the name of the class represented. The images were taken from the Hearthstone website HSReplay.net. Figure 5 shows an example of image from the reference set.

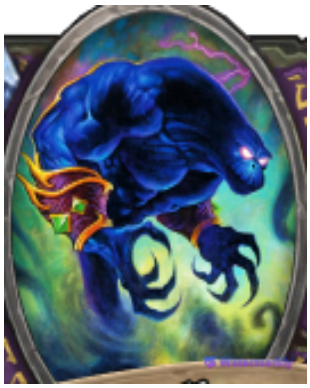

Fig. 5. An image from the reference dataset

4.2 Prototype

As an object recognition model we chose models of the YOLO family. OpenCV[14] was chosen as the computer vision library providing the algorithms necessary for utilizing the reference images. We also used the Python 3.10.9 to implement the prototype.

The *FlannBasedMatcher* algorithm from OpenCV was used to find and match key points between the reference and input images using the nearest-neighbor method more efficiently. The matching process involves calculating several metrics, such as Euclidean distance or Hamming distance, and selecting the most similar matches.

Listing 1.1 implements the classification algorithm, where *ref* is the reference image, *crop* – region of the detected image cropped by the rectangle obtained from YOLO, *kp1*, *kp2* – key points for *ref* and *crop* respectively, *des1 des2* are descriptors of key points *kp1* and *kp2*, *MIN_MATCH_COUNT* is the minimum number of matching points between images, at which the class is recognized (in our work it was taken as 10).

Listing 1.1. Python implementation of the classification algorithm

```
# Setting up the points
sift = cv.SIFT_create()
kp1, des1 = sift.detectAndCompute(ref, None)
kp2, des2 = sift.detectAndCompute(crop, None)
# Finding matches
index_params = dict(algorithm=FLANN_INDEX_KDTREE,
                    trees=5)
search_params = dict(checks=50)
flann = cv.FlannBasedMatcher(index_params,
                             search_params)
matches = flann.knnMatch(des1, des2, k=2)
# Count the number of matching vectors
count = 0
for m, n in matches:
    if m.distance < 0.7 * n.distance:
        count += 1
# Compare with the threshold and maximum values
if count > MIN_MATCH_COUNT and best_match[1] < count:
    best_match = name, count
# Add the best class to the list of recognized classes
    class_name = best_match[0]
    classes.append(class_name)
```

[14] https://opencv.org/

4.3 Results

For evaluation, both datasets (first and second) were divided into training, test, and validation parts. The distribution was 194, 56, and 28 images, respectively.

The configuration of the computer on which the training was performed is shown in Table 1.

Table 1. Computer Configuration

CPU	AMD Ryzen 5600X
GPU	NVIDIA GeForce RTX 2060 SUPER
RAM	Patriot Viper Steel 2x8GB DDR4 3200MHz
ROM	KINGSTON SNVS1000G 1TB

The Table 2 represents the settings of the YOLO models.

Table 2. The parameters of YOLO models

Parameter	**Value**
epochs	150
batch	16
optimizer	SGD

The metrics of the models performance are presented in Table 3, where YOLOvX + CV is the proposed hybrid model using YOLOvX trained on the second data set together with the classification Algorithm 1, and YOLOvX – the benchmark model YOLOvX trained on the first data set. An example image with recognition results is shown in Figure 6. Here, the recognized card boundaries are indicated by colored frames, and the caption above the frame corresponds to the recognized class.

Table 3. Metrics

Model	**Precision**	**Recall**	**F1**	**Dataset**
YOLOv5 + CV	**0,967**	**0,976**	**0,971**	2+3
YOLOv8 + CV	**0,967**	**0,976**	**0,971**	2+3
YOLOv5	0,823	0,819	0,821	1
YOLOv8	0,778	0,834	0,805	1

The second dataset is not intended to recognize different classes, but only to detect the presence of an object in the image. Therefore, using the YOLOvX model without CV on it would not make sense and we can compare only the

Fig. 6. Example image with recognition results

results of the models on identical images but with different annotations. The proposed method achieved an F1 score of 0.971 when trained on just 274 images and one labeled class, which is 0.15 more than YOLOvX trained on the same images but with 142 class labels. This allows us to conclude that with the same amount of training data the developed model shows better results than YOLOv5 and YOLOv8, using simpler markup.

5 Conclusion

This paper presents an overview of the state-of-the-art AI applications in computer games and of the hybrid approach to computer vision, demonstrating that hybrid computer vision has not been applied yet to video games. An informed machine learning-based model of object recognition in games was developed, combining a machine learning model that determines object boundaries and an algorithm that classifies detected objects based on comparison with images from a set of reference images. To evaluate the proposed method, a prototype was implemented, a dataset of images from the Hearthstone game was collected and annotated in two different ways. The experiments showed that the developed model provides higher accuracy compared to the end-to-end application of machine learning models of the YOLO family. The presented approach is applicable to other games that have sufficient documentation to create a dataset of reference images. A significant advantage of our method is the qualitative improvement of the classification generalizability. If new classes appear as a result of a game update, our method will not require new labeled data and retraining

of the recognition model unlike classical supervised learning. Our method only requires adding single images of new classes to the set of reference images.

One limitation of the proposed method is its time complexity which depends linearly on the number of reference images. The speed is important for real-time applications such as computer games. To eliminate this limitation, further research and optimization of the proposed method are needed. For example, real-time performance can be achieved through alternative architectures or the use of hardware acceleration techniques.

References

1. Baek, I., Kim, S.B.: 3-dimensional convolutional neural networks for predicting StarCraft 2 results and extracting key game situations. PLOS ONE **17**(3), e0264550 (Mar 2022), https://doi.org/10.1371/journal.pone.0264550
2. Campbell, M., Hoane, A., hsiung Hsu, F.: Deep blue. Artificial Intelligence **134**(1), 57–83 (2002). https://doi.org/https://doi.org/10.1016/S0004-3702(01)00129-1
3. Ding, Z., Yao, L., Liu, B., Wu, J.: Review of the Application of Ontology in the Field of Image Object Recognition. In: Proceedings of the 11th International Conference on Computer Modeling and Simulation. pp. 142–146. ACM, North Rockhampton QLD Australia (Jan 2019). https://doi.org/10.1145/3307363.3307387
4. Erdelyi, C.: Using computer vision techniques to play an existing video game (March 2019), https://csusm-dspace.calstate.edu/bitstream/handle/10211.3/209944/ErdelyiChristopher_Spring2019.pdf
5. Filali, J., Zghal, H., Martinet, J.: Ontology and HMAX Features-based Image Classification using Merged Classifiers:. In: Proceedings of the 14th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications. pp. 124–134. SCITEPRESS - Science and Technology Publications, Prague, Czech Republic (2019). https://doi.org/10.5220/0007444101240134
6. Fink, A., Denzinger, J., Aycock, J.: Extracting NPC behavior from computer games using computer vision and machine learning techniques. In: 2007 IEEE Symposium on Computational Intelligence and Games. IEEE (2007), https://doi.org/10.1109/cig.2007.368075
7. Fukui, A., Park, D.H., Yang, D., Rohrbach, A., Darrell, T., Rohrbach, M.: Multimodal Compact Bilinear Pooling for Visual Question Answering and Visual Grounding (Sep 2016)
8. Gao, Y., Beijbom, O., Zhang, N., Darrell, T.: Compact Bilinear Pooling (Apr 2016)
9. Grebenisan, A.: Play super mario bros with a double deep q-network (2020), https://blog.paperspace.com/building-double-deep-q-network-super-mario-bros/
10. He, Y., Tian, L., Zhang, L., Zeng, X.: Knowledge Graph Representation Fusion Framework for Fine-Grained Object Recognition in Smart Cities. Complexity **2021**, 1–9 (Jul 2021). https://doi.org/10.1155/2021/8041029
11. Jocher, G., Ayush Chaurasia, Stoken, A., Borovec, J., NanoCode012, Yonghye Kwon, Kalen Michael, TaoXie, Jiacong Fang, Imyhxy, Lorna, Zeng Yifu, Wong, C., Abhiram V, Montes, D., Zhiqiang Wang, Fati, C., Jebastin Nadar, Laughing, UnglvKitDe, Sonck, V., Tkianai, YxNONG, Skalski, P., Hogan, A., Dhruv Nair, Strobel, M., Jain, M.: ultralytics/yolov5: v7.0 - YOLOv5 SOTA Realtime Instance Segmentation (Nov 2022). https://doi.org/10.5281/ZENODO.7347926
12. Lample, G., Chaplot, D.S.: Playing fps games with deep reinforcement learning (January 2018), https://arxiv.org/abs/1609.05521

13. Lipka, P.: Gamescripter - a vision based tool for playing games (June 2011), https://www.doc.ic.ac.uk/teaching/distinguished-projects/2011/p.lipka.pdf
14. Lommaert, K.: Deep learning for pattern recognition in movements of game entities (June 2019), https://www.gamedeveloper.com/design/deep-learning-for-pattern-recognition-in-movements-of-game-entities
15. Monka, S., Halilaj, L., Rettinger, A.: A survey on visual transfer learning using knowledge graphs. Semantic Web **13**(3), 477–510 (Apr 2022). https://doi.org/10.3233/SW-212959
16. Qiu, W., Zhong, F., Zhang, Y., Qiao, S., Xiao, Z., Kim, T.S., Wang, Y.: UnrealCV. In: Proceedings of the 25th ACM international conference on Multimedia. ACM (Oct 2017), https://doi.org/10.1145/3123266.3129396
17. von Rueden, L., Mayer, S., Beckh, K., Georgiev, B., Giesselbach, S., Heese, R., Kirsch, B., Pfrommer, J., Pick, A., Ramamurthy, R., Walczak, M., Garcke, J., Bauckhage, C., Schuecker, J.: Informed Machine Learning – A Taxonomy and Survey of Integrating Knowledge into Learning Systems. IEEE Transactions on Knowledge and Data Engineering pp. 1–1 (2021). https://doi.org/10.1109/TKDE.2021.3079836
18. Shafaei, A., Little, J.J., Schmidt, M.: Play and learn: Using video games to train computer vision models (2016), https://arxiv.org/abs/1608.01745
19. Silver, D., Huang, A., Maddison, C.J., Guez, A., Sifre, L., Van Den Driessche, G., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., Lanctot, M., Dieleman, S., Grewe, D., Nham, J., Kalchbrenner, N., Sutskever, I., Lillicrap, T., Leach, M., Kavukcuoglu, K., Graepel, T., Hassabis, D.: Mastering the game of Go with deep neural networks and tree search. Nature **529**(7587), 484–489 (Jan 2016). https://doi.org/10.1038/nature16961, https://www.nature.com/articles/nature16961
20. Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., Wojna, Z.: Rethinking the Inception Architecture for Computer Vision. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 2818–2826. IEEE, Las Vegas, NV, USA (Jun 2016). https://doi.org/10.1109/CVPR.2016.308
21. Vinyals, O., Babuschkin, I., Czarnecki, W.M., Mathieu, M., Dudzik, A., Chung, J., Choi, D.H., Powell, R., Ewalds, T., Georgiev, P., Oh, J., Horgan, D., Kroiss, M., Danihelka, I., Huang, A., Sifre, L., Cai, T., Agapiou, J.P., Jaderberg, M., Vezhnevets, A.S., Leblond, R., Pohlen, T., Dalibard, V., Budden, D., Sulsky, Y., Molloy, J., Paine, T.L., Gulcehre, C., Wang, Z., Pfaff, T., Wu, Y., Ring, R., Yogatama, D., Wünsch, D., McKinney, K., Smith, O., Schaul, T., Lillicrap, T.P., Kavukcuoglu, K., Hassabis, D., Apps, C., Silver, D.: Grandmaster level in starcraft ii using multi-agent reinforcement learning. Nature pp. 1–5 (2019)
22. Zhang, D., Cui, M., Yang, Y., Yang, P., Xie, C., Liu, D., Yu, B., Chen, Z.: Knowledge Graph-Based Image Classification Refinement. IEEE Access **7**, 57678–57690 (2019). https://doi.org/10.1109/ACCESS.2019.2912627
23. Zhang, Y., Qu, Y., Li, C., Lei, Y., Fan, J.: Ontology-driven hierarchical sparse coding for large-scale image classification. Neurocomputing **360**, 209–219 (Sep 2019). https://doi.org/10.1016/j.neucom.2019.05.059

List of reviewers' comments

---------- REVIEW 1 ----------

SUBMISSION: 27

TITLE: Informed Object Detection for Computer Games

AUTHORS: Anton Dubovskoi, Maksim Sokolov and Ildar Baimuratov

---------- Overall evaluation ----------

SCORE: 1 (weak accept)

----- TEXT:

The paper is focused on the acute problem of training data for computer games research using AI approaches. To reduce the need for labeled data, authors try to find a hybrid approach that combines machine learning with knowledge representation by object recognition in computer games.

In the paper an overview of the state-of-the-art AI applications in computer games and a hybrid approach to computer vision are presented first. Then follows presentation of an informed machine learning-based model of object recognition in games, combining a machine learning model that determines object boundaries and an algorithm that classifies detected objects based on comparison with images from a sample set, developed by authors. For evaluation the proposed method, a dataset based on Heathstone game images with two annotation variants was collected and a prototype application was implemented.

It seems reasonable to make the following remarks:

1) One cannot find explicit explanation in the paper for a reason why the computer game Hearthstone was chosen as the application? Will the proposed method be valid for other computer games? What will happend to the results of evaluation for a dataset based on other computer games?

2) The results section (4.3) of the paper is too short. I would recommend to extend it by adding more details about the experiment and evaluation procedure and presented results discussion as well.

3) Some textual comments are necessary to figure 6. - What are recognition results?

Response:

1) In this study, the computer game of Blizzard Ent., Hearthstone, was chosen as the use case as it is an eSports discipline with the largest number of users in its genre, which also has a sufficient number of open resources and documentation necessary for the use of IML. *(1 Introduction)*

The presented approach is applicable to other games that have sufficient documentation to create a dataset of reference images. *(5 Conclusion)*

2) A significant advantage of our method is the qualitative improvement of the classification generalizability. If new classes appear as a result of a game update, our method will not require new labeled data and retraining of the recognition model unlike classical supervised learning. Our method only requires adding single images of new classes to the set of reference images. *(5 Conclusion)*

3) Here, the recognized card boundaries are indicated by colored frames, and the caption above the frame corresponds to the recognized class. *(4.3 Results)*

---------- REVIEW 2 ----------

SUBMISSION: 27

TITLE: Informed Object Detection for Computer Games

AUTHORS: Anton Dubovskoi, Maksim Sokolov and Ildar Baimuratov

---------- Overall evaluation ----------

SCORE: 2 (accept)

----- TEXT:

Сильные стороны работы

- Рассматривается интересная задача из области компьютерного зрения. Авторы используют автоматические методы для распознания элементов компьютерной игры прямо по экранной картинке. Таким образом, становится возможным создавать автоматических игроков-роботов, которые не являются частью игровой программы, но взаимодействуют с ней посредством обычного пользовательского интерфейса. Думаю, что результаты данной работы могут применяться в дальнейшем и для автоматизации процессов в других областях (см. RPA).

- Предложено решение базирующееся на авторском алгоритме, основанном на применении информированного машинного обучения.

- Работа написана ясным языком и легко читается.

Слабые стороны работы

- "Based on the above metrics, we can conclude that the developed model shows better results than YOLOv5 and YOLOv8, while using a simpler annotation." - на самом деле, из текста не вполне очевидно, почему аннотирование в предлагаемом авторами подходе проще. Стоит сказать про это подробнее, как-то подтвердить это утверждение фактами, сравнениями.

- Также стоило бы подробнее рассмотреть тот факт, что сравнение разных моделей происходит на различных набор данных. Надо как-то объяснить, что такое сравнение действительно состоятельно.

Предложения по улучшению

- Хорошая идея - добавить в раздел 2 параграф, который бы пояснил место предлагаемого авторами метода среди рассмотренных.

- Таблицы 1 и 2 вполне могли бы и не быть таблицами. В данном случае такой формат скорее съедает дополнительное место.

- Было бы интересно посмотреть на результаты в сравнении, когда используется только первый или второй набор данных.

Резюме

Считаю, что работа довольно хорошо подготовлена, и также интересна для обсуждения в рамках конференции.

Response:

1) The second dataset includes the same 274 images as the first, but the annotation contains only 1 instead of 142 classes. This format requires significantly less intellectual effort from an annotator, speeds up the annotation process, and reduces the likelihood of errors when determining the class. *(4.1 Data)*

The third dataset is a reference dataset of 142 images representing unique cards. These images do not require annotations, but the file name must include either the name or a specific label that references the name of the class represented. *(4.1 Data)*

2) The second dataset is not intended to recognize different classes, but only to detect the presence of an object in the image. Therefore, using the YOLOvX model without CV on it would not make sense and we can compare only the results of the models on identical images but with different annotations. *(4.3 Results)*

---------- REVIEW 3 ----------

SUBMISSION: 27

TITLE: Informed Object Detection for Computer Games

AUTHORS: Anton Dubovskoi, Maksim Sokolov and Ildar Baimuratov

---------- Overall evaluation ----------

SCORE: 1 (weak accept)

----- TEXT:

The paper describes a method to identify objects in computer games without labels.

Strong points of the paper:

The bibliography seems complete.

Potential use in different areas.

The paper is novel in that it uses methods that have been used in other fields to computer games.

The explanations are quite clear, in general terms.

Points that are to be discussed:

The experiments need to be explained with more detail.

The amount of pictures that are used for testing is not relevant. From a statistical point of view, Section 4.3 is weak and need more explanation and detail.

Point to be clarified: why did the authors chose this particular game? It would have been much more interesting to chose different games to text.

Improvements:

Place python code in an appendix.

The English is somehow convoluted in some paragraphs. I suggest to check the language carefully.

Response:

1) The second dataset is not intended to recognize different classes, but only to detect the presence of an object in the image. Therefore, using the YOLOvX model without CV on it would not make sense and we can compare only the results of the models on identical images but with different annotations. The proposed method achieved an F1 score of 0.971 when trained on just 274 images and one labeled class, which is 0.15 more than YOLOvX trained on the same images but with 142 class labels. This allows us to conclude that with the same amount of training data the developed model shows better results than YOLOv5 and YOLOv8, using simpler markup. *(4.3 Results)*