YOLO-based Mobile Legends Match Result Parsing

Muhammad Rizqi Nur
Information System
Institut Teknologi Sepuluh Nopember
Surabaya, Indonesia
rizqinur2010@gmail.com

Rarasmaya Indraswari
Information System
Institut Teknologi Sepuluh Nopember
Surabaya, Indonesia
raras@its.ac.id

***Abstract*—MOBA competitive gaming can benefit from AI advancement. However, data availability is a major issue for Mobile Legends, as opposed to more mature MOBA. In order to obtain high amount of data, it has to be crowdsourced, but it is only viable to collect screenshots. In this paper we propose a framework to automatically parse mobile legends match result screenshots based on YOLO. YOLO is used to locate and classify objects. Text objects are then parsed with OCR. The results are evaluated and compared with older approach using CNN classifiers. The new approach is 25 times faster while achieving the same perfect performance as the old CNN classifier approach.**

Keywords—Mobile Legends, image parsing, object recognition, optical character recognition

I. Introduction

Mobile Legends, released in 2017 by Moonton, is one of the most popular Multiplayer Online Battle Arena (MOBA) game ins southeast Asia. By April 2023, Mobile Legends had reached 78 millions monthly players and 5 millions daily players [1]. Mobile Legends has also became an e-sport by having national and international tournaments held officially by Moonton, which are ML Professional League (MPL) and M World Championship (M-Series), with prize pool that keeps increasing. The 11th season of MPL Indonesia had IDR 4.8 billion prize pool [2], while M4 had 12 billions prize pool [3].

Victory in Mobile Legends matches doesn't only depend on the player skills, but also by the team composition and teamwork. Mobile Legends has 120 heroes and each team has to choose 5. Each player in a team has to fill a certain position and role, while each hero can only fill certain position and roles. The line up of a team has to fill every position and role properly [4]. Furthermore, each hero has strengths and weaknesses. Strengths of a hero are to be combined with another, while the weaknesses are to be complemented by another. On the other hand, the weakness of enemies are to be exploited [5]. Therefore, a hero can have synergic or counter relationship with another [6]–[8]. The line up of a team should maximize synergy within the team while also countering the enemy team. The strategy relies heavily on the line up, thus the team line up heavily affects victory. In addition to that, a bad line up can cause poor gaming experience [7].

The making of team line up is done in drafting phase, where each team alternately ban and choose heroes [4], [7]. The subsequent hero picks and the strategy of a team can be guessed by the heroes they ban and pick early [9]. Therefore the fight already begun in drafting phase where each team tries to guess and counter enemy strategy before the actual game even begin. In normal matches, drafting is done by the players. In professional matches, however, drafting is often helped by coaches or analysts [4].

Each player and coach should have deep understanding of the heroes and their relationships. However, the huge hero pool and the time limit can still be overwhelming [5], [7], [8]. A lot of research has been done to develop drafting support system for MOBA [5] [4] [10], [11] [7], [8], but very few has been done for Mobile Legends and they don't use modern approaches [12]–[15]. This is primarily caused by the lack of data availability. DOTA 2 and LoL are very mature and have websites dedicated to providing game data and statistics, so it is easy to obtain data to train models. Such sites do not exist for Mobile Legends yet, and Moonton does not provide any way to obtain game data either. Therefore, there needs to be a way to collect these data.

In our preliminary study, we have tried to collect match result data from screenshots and train a model ourselves. We used LD Player, OpenCV, CNN classifiers, and OCR tools to automate the process. However, data collection and parsing still takes a lot of time and only 5056 data were collected within one week. That amount of data was not enough as the model keeps overfitting. To collect more data faster, it has to be crowd sourced and the parsing has to be faster. If screenshots are crowd sourced, the cropping strategy we used will not be applicable because every device has different screen resolution and aspect ratio. Furthermore, running CNN classifiers multiple times, as done previously in our preliminary study, is slow. Therefore, in this paper we propose a Mobile Legends match result screenshot parser based on YOLO. The YOLO model is used to both do classification tasks, as well as detect the bounding box of texts to crop and parse with OCR [16]. The performance is compared to the previous CNN approach.

II. Yolo

You Only Look Once (YOLO) [17] is a model that revolutionized the object recognition topic. Object recognition used to be done by performing object detection and classification separately; passing the cropped result of object detection to the classifier. This method is slow; too slow to be used for real time object detection. YOLO fixed this by performing object detection and classification at once; hence the name. The YOLO model has three objectives: bounding box regression, object detection (whether there is an object in the box), and object classification (what the object is). The class confidence score is then calculated as conditional probability by multiplying the object probability and the highest class probability. Training a model to do this from scratch would be very hard. YOLO achieved it by doing transfer learning from a pretrained CNN classifier as a backbone.

YOLO keeps being improved, with YOLOv8 being the latest version [18]. YOLOv2 added predefined anchor boxes which helped make better predictions of various aspect ratios.

It also used batch normalization and retrained the CNN with higher resolution images. YOLOv3 makes predictions at three levels of granularity, which fixed YOLOv2 issue of small object detection. It also added residual connections. The original author discontinued YOLO at v3, so the further YOLO versions are not developed by the original author. YOLOv4 uses path aggregation which uses concatenation to combine features from different levels of layer. It also added a lot of optimizations. Similar to YOLOv4, YOLOv5 improves by adding optimizations with state-of-the-art techniques at the time. In addition to that, it is the first version to be written in PyTorch, which makes further development and research easier. Furthermore, it added automated anchor box learning from the dataset. YOLOv6 does anchor-free approach which makes it 51% faster. YOLOv6 also optimizes the network to a range of hardwares and decouples the prediction head to make prediction at each level of granularity. YOLOv6 also introduced new loss functions and uses knowledge distillation (teacher-student) to train the smaller models. YOLOv7 reparametrized the architecture to be able to be scaled flexibly for tradeoff between performance and speed. It also proposed an auxiliary head to learn to predict at the early layers, so better features are learned. YOLOv8 improved to target commercial usage, with improved speed and accuracy while also having high compatibility and being easy to train and use.

III. Method

A. Data

The ground truth of the parsing result is the data we have previously collected from patch 1.7.68. This time non-ranked matches, invalid matches, and matches with AFK players are re-included, resulting in 5329 images. Although the data was collected and parsed automatically, it has been validated manually to ensure correctness. To train the YOLO model, each screenshot was labeled using the ground truth data (Figure 1). The bounding boxes are all in the same position for every image because they were taken from the same LD Player instance, so it can be done automatically by script. This will not be an issue for the YOLO training because augmentations will be applied to alter the bounding boxes. The data were split into training and test set with 80:20 ratio using pylabel to ensure that the classes are balanced. This results in 4229 training set and 1100 test set. The data are available on Kaggle [19].

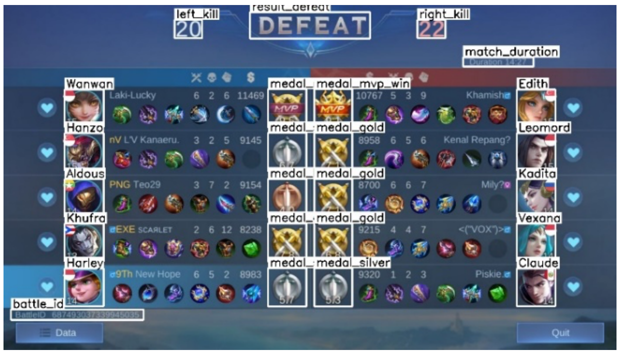

Figure 1. Labeled screenshot sample

B. Framework

The parsing framework is illustrated in Figure 2. YOLOv8 is used to detect the objects, that is to determine the bounding boxes and assign a class. Classification tasks, such as determining the hero, are done directly as the object detection task. Objects that belong to left or right team are assigned by their normalized position. Text objects such as battle ID are further processed with OCR using PaddleOCR. The score below the medals is also processed with OCR after cropping it. The match duration is converted to total minutes.

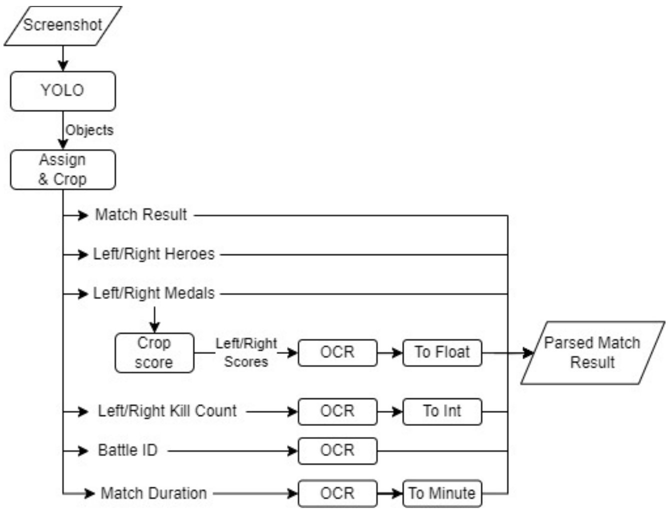

Figure 2. The parsing framework

C. Evaluation

The model is trained for 3x50 epochs. The model was trained for 50 epochs, then the best model is loaded. This is repeated 3 times. The training was done using Nvidia P100 GPU on Kaggle. For classification tasks, the approach is evaluated with accuracy and F1-score. For numbers, the approach is evaluated with root mean squared error (RMSE) and mean absolute error (MAE). For battle ID though, the approach is evaluated with character error rate (CER) and Levenshtein distance. The inference time per image is reported. However, the OCR time is excluded because both approaches use the same OCR method.

The new approach is compared to a CNN classifier approach used in our preliminary study (Figure 3) [20]. In the old approach, pictures of interest are cropped out of a match result screenshot and processed with CNN classifiers and OCR to extract information. The CNN classifiers were trained by transfer learning from MobileNet V3 Large [21]. The OCR used is PaddleOCR. Both were found to be the fastest and most accurate ready-to-use options.

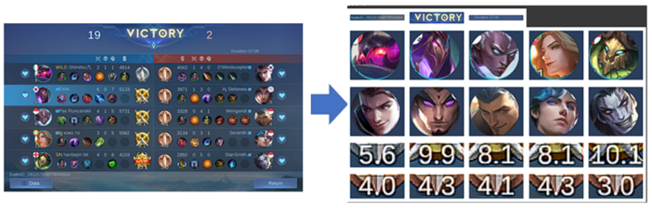

Figure 3. The old approach using cropping, MobileNet v3 CNN classifiers, and PaddleOCR

IV. Results and Discussion

A. Object Recognition

An example of object recognition result is shown on Figure 4. Objects are predicted with very high confidence. Table 1 shows the object recognition metrics on validation set. There are 131 classes to recognize with 119 of them being hero icons. It appears that 150 epochs were enough for the model to converge.

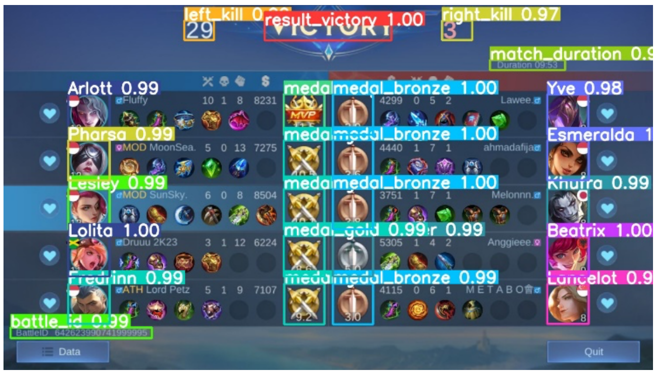
Figure 4. Example of object recognition result

TABLE 1. OBJECT RECOGNITION METRICS

Metric	Value
Precision	1.0
Recall	1.0
mAP50	0.995
mAP50-95	0.995
Fitness	0.995

B. *Classification Tasks*

The new approach performed perfectly, reaching perfect accuracy and F1-score for match result, hero, and medal parsing, just like the old approach. However, it did not happen out of the box. Previously, the YOLO model predicted 6 heroes for a team in some cases, although there should have been 5. A closer look on one of the cases shows that a hero icon is predicted twice, but the wrong one has lower confidence (Figure 5). This was easily fixed by raising the confidence threshold.

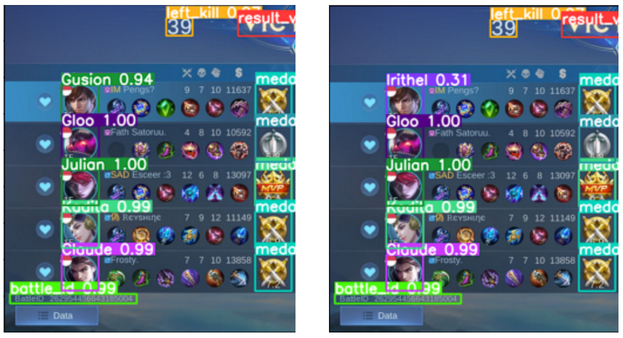
(a) (b)

Figure 5. Example of multiple overlapping detection on one object: (a) correct prediction on top, (b) wrong prediction on top

C. *Number Recognition Tasks*

The new approach performed perfectly, reaching zero RMSE and MAE for team kill count, match duration, and medal scores, just like the old approach. However, it did not happen out of the box. Most of them were caused by misreading 5 as 9 (Figure 6). This was caused by the swords of the gold medal, which are dominantly white, blending with the white score text. This was fixed by doing a better cropping of the score text. It might be better to try to fine tune the OCR model for this task, but it is outside the scope of this study.

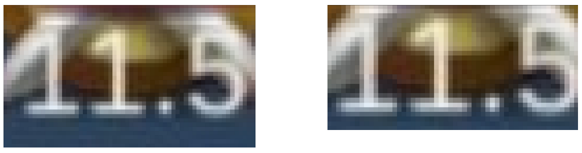
(a) (b)

Figure 6. Example of score text blended with the medal swords: (a) bad cropping, (b) better cropping

D. *Pure OCR Task*

The new approach performed perfectly, reaching zero Levenshtein distance and character error rate (CER), just like the old approach.

E. *Inference Time*

Table 2 shows the inference time comparison. The new approach is 25 times faster than the old approach. This is because YOLO does everything in one forward pass.

TABLE 2. INFERENCE TIME IN SECONDS

Type	Old	New
Average Per Image	0.807	0.032
Total	888.788	36.162

V. CONCLUSION

This study has developed a Mobile Legends match result parser based on YOLO. The new approach is 25 times faster while achieving the same perfect performance as the previous approach. However, it still has some shortcomings. Mobile Legends keep being updated. New heroes are added, old heroes are revamped, and user interface may change. The CNN classifiers used in the old approach are easy to train; the datasets are easy to prepare. When new hero is added or old heroes are revamped, the change to the dataset is minimal. Object recognition dataset, however, is hard to prepare. It was easy in this study only because the old data and parsing result were available. Future research can be done to extend YOLOv8 to support template matching in addition to the trained classes, so it can be easily adapted to visual changes. Alternatively, future research can be done to find a way to fine tune YOLO with classification dataset.

REFERENCES

[1] activeplayer.io, 'Mobile Legends: Bang Bang Live Player Count and Statistics', *activeplayer.io*, 2023. https://activeplayer.io/mlbb/ (accessed May 30, 2023).

[2] S. D. Maarif, 'Jadwal MPL S11 2023: Daftar Tim, Format, Prize Pool, Cara Nonton', *tirto.id*, 2023. https://tirto.id/jadwal-mpl-s11-2023-daftar-tim-format-prize-pool-cara-nonton-gCl5 (accessed May 30, 2023).

[3] C. W. Surbakti, 'Distribusi prize pool M4 World Championship', *oneesports.id*, 2023. https://www.oneesports.id/mobile-legends/prize-pool-m4-world-championship/ (accessed May 30, 2023).

[4] D. Gourdeau and L. Archambault, 'Discriminative Neural Network for Hero Selection in Professional Heroes of the Storm and DOTA 2', *IEEE Trans. Games*, vol. 13, no. 4, pp. 380–387, 2021, doi: 10.1109/TG.2020.2972463.

[5] H. Lee, D. Hwang, H. Kim, B. Lee, and J. Choo, *DraftRec: Personalized Draft Recommendation for Winning in Multi-Player Online Battle Arena Games*, vol. 1, no. 1. Association for Computing Machinery, 2022. doi: 10.1145/3485447.3512278.

[6] S. J. Hong, S. K. Lee, and S. Il Yang, 'Champion Recommendation System of League of Legends', *Int. Conf. ICT Converg.*, vol. 2020-Octob, pp. 1252–1254, 2020, doi: 10.1109/ICTC49870.2020.9289546.

[7] Z. Chen *et al.*, 'The Art of Drafting: A Team-Oriented Hero Recommendation System for Multiplayer Online Battle Arena Games', 2018, [Online]. Available: http://arxiv.org/abs/1806.10130

[8] S. Chen, M. Zhu, D. Ye, W. Zhang, Q. Fu, and W. Yang, 'Which Heroes to Pick? Learning to Draft in MOBA Games with Neural Networks and Tree Search', *IEEE Trans. Games*, vol. 13, no. 4, pp. 410–421, 2021, doi: 10.1109/TG.2021.3095264.

[9] A. Summerville, M. Cook, and B. Steenhuisen, 'Draft-Analysis of the ancients: Predicting draft picks in DotA 2 using machine learning', *AAAI Work. - Tech. Rep.*, vol. WS-16-21-, no. Godec, pp. 100–106, 2016, doi: 10.1609/aiide.v12i2.12899.

[10] Y. Shen, J. Zhou, W. Lin, and Z. Feng, 'A Deep Learning Supported Sequential Recommendation Mechanism for Ban-Pick in MOBA Games', *2022 2nd IEEE Int. Conf. Softw. Eng. Artif. Intell. SEAI 2022*, pp. 259–265, 2022, doi: 10.1109/SEAI55746.2022.9832346.

[11] T. Inzitari, B. Lyons, and N. Al, *Predicting pick-ban sequence in League of Legends games*, vol. 5, no. CSCW1. Association for Computing Machinery, 2021.

[12] J. L. Putra and S. Seimahuira, 'Memprediksi Pola Ban Hero Pada Game Mobile Legends Menggunakan Algoritma Apriori', *Comput. Sci.*, vol. 1, no. 2, pp. 155–160, 2021, doi: 10.31294/coscience.v1i2.512.

[13] CA Tanjung, 'Sistem Penentuan Hero Counter pada Game Mobile Legends dengan Logika Fuzzy Metode Mamdani', 2018.

[14] A. T. Susilo, H. Setiawan, R. A. Saputro, T. Purwadi, and A. Saifudin, 'Penggunaan Metode Naïve Bayes untuk Memprediksi Tingkat Kemenangan pada Game Mobile Legends', *J. Teknol. Sist. Inf. dan Apl.*, vol. 4, no. 1, p. 46, 2021, doi: 10.32493/jtsi.v4i1.7807.

[15] A. Dharmawan, J. Gondohanindijo, Y. Prihati, and T. Hafidzin, 'Optimization of Players and Game Win Prediction Using Naïve Bayes Algorithm', *J. Elektro Luceat*, vol. 8, no. 1, 2022.

[16] A. Chazhoor and V. R. Sarobin, 'Intelligent automation of invoice parsing using computer vision techniques', *Multimed. Tools Appl.*, vol. 81, no. 20, pp. 29383–29403, 2022, doi: 10.1007/s11042-022-12916-x.

[17] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, 'You only look once: Unified, real-time object detection', *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, vol. 2016-Decem, pp. 779–788, 2016, doi: 10.1109/CVPR.2016.91.

[18] M. Hussain, 'YOLO-v1 to YOLO-v8, the Rise of YOLO and Its Complementary Nature toward Digital Manufacturing and Industrial Defect Detection', *Machines*, vol. 11, no. 7, 2023, doi: 10.3390/machines11070677.

[19] M. R. Nur, 'Mobile Legends Match Result Object Detection Dataset', *Kaggle*, 2023. https://www.kaggle.com/datasets/rizqinur/mobile-legends-match-result-object-detection (accessed Dec. 24, 2023).

[20] M. R. Nur, 'R-N/ml_draftpick_dss: Mobile legends draft pick decision support system', *Github*, 2023. https://github.com/R-N/ml_draftpick_dss (accessed May 08, 2024).

[21] A. Howard, W. Wang, G. Chu, L. Chen, B. Chen, and M. Tan, 'Searching for MobileNetV3', in *International Conference on Computer Vision*, 2019, pp. 1314–1324.