A Dataset to Investigate First-Person Shooter Players

David Halbhuber
david.halbhuber@ur.de
University of Regensburg
Regensburg, Germany

Julian Höpfinger
julian.höpfinger@stud.ur.de
University of Regensburg
Regensburg, Germany

Valentin Schwind
valentin.schwind@acm.org
Frankfurt University of Applied Sciences
Frankfurt, Germany

Niels Henze
niels.henze@ur.de
University of Regensburg
Regensburg, Germany

ABSTRACT

Datasets are multi-purpose research tools, enabling researchers to design, develop, and test solutions to classical computer sciences problems and novel research questions. In the gaming domain, however, there are few high-quality datasets providing both: (1) visual gameplay data and (2) additional information about the gameplay, such as user input. As a result, game researchers most of the time have to collect, process, and annotate gameplay data in time-consuming data collection studies themselves. We start closing this gap, by presenting a novel Counter-Strike: Global Offensive dataset. The contributed dataset is a collection of 12 high-skilled players playing Counter-Strike: Global Offensive. We showcase two deep learning-based examples using the presented dataset, demonstrating its versatility.

KEYWORDS

datasets, artificial neural networks, counter-strike: global offensive

ACM Reference Format:
David Halbhuber, Julian Höpfinger, Valentin Schwind, and Niels Henze. 2022. A Dataset to Investigate First-Person Shooter Players . In *Extended Abstracts of the Annual Symposium on Computer-Human Interaction in Play (CHI PLAY '22 EA), November 2–5, 2022, Bremen, Germany.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3505270.3558331

1 INTRODUCTION

Datasets allow researchers to investigate research questions, conduct experiments, and test novel methods without requiring time-consuming data collection. These datasets are usually highly versatile and allow multi-faceted research. The *Modified National Institute of Standards and Technology* (MNIST) dataset [22], for example, is a collection of 280 000 handwritten digits and was initially designed to be used as a benchmark in a computer vision (CV) challenge. Multiple teams attempted, and still attempt, to find the best computer-aided method for optical character recognition (OCR) [4, 9] in the MNIST challenge. However, MNIST is now used for a broad range of research projects due to its versatility. Shmelkov et al. [37], for example, used a generative adversarial network (GAN) trained on the MNIST dataset to generate artificial images of handwritten digits. In other work, Netzer et al. [31] used the dataset to train an unsupervised feature learning algorithm to read digits in natural images.

Publicly available datasets, such as MNIST, accelerated the development of novel approaches to classic computer sciences problems. Moreover, numerous research fields, such as medicine [16, 18, 30], deep learning [13, 17], and natural language processing [3, 43] benefited from datasets and their versatility. However, there are few high-quality datasets available in the gaming domain. As a result, gaming researchers need to gather data, annotate it and preprocess it themselves [15]. Evidently, there are only few datasets of Counter-Strike:Global Offensive (CS:GO) - a game played by almost 600 000 unique players daily and which is highly relevant to the e-sports scene [39, 40]. While some large datasets for CS:GO are available [20, 21, 34], those datasets do not provide visual game play data. These datasets do, however, provide valuable meta-information such as the outcome of matches, statistics about used weapons [21, 34], or behavioural data such as the eye movement and gaze pattern about the players [20, 32]. A public datasets of actual CS:GO game play, however, may accelerate research in and with the game.

In this work, we start closing the gap between gaming research and other data-richer research fields by providing a novel, yet still small, high-quality gaming dataset. We present our ongoing endeavor to create a Counter-Strike: Global Offensive dataset. So far, our dataset is a collection of twelve highly-skilled players (mean playtime: 1864.08 ± 1042.69 hours) playing *Counter-Strike: Global Offensive* (CS:GO). We recorded about 2 764 800 unique frames at 64 fps. We provide three versions of the material: (1) a pre-wrapped Python *Pickle* in 60×34 pixel, (2) a further down-scaled Python *Pickle* in 30×17 pixel, and finally (3) the original raw material using CS:GO's replay file format. Researchers using our dataset can either use one of the provided formats or generate the desired output using the replay data and our presented approach. Besides the visual material, our dataset is also highly annotated. For each recorded frame, we provide information about the current game state, such as the player's health, the equipped weapon, or whether an enemy spotted the player. Furthermore, we provide detailed data about the inputs to the game, such as which keys are pressed and how the mouse position changes between two consecutive frames.

CHI PLAY '22 EA, November 2–5, 2022, Bremen, Germany

ACM ISBN 978-1-4503-9211-2/22/11.
https://doi.org/10.1145/3505270.3558331

To conclude this work, we showcase two examples of using our dataset. The first example uses the input part of the dataset to predict mouse movement in the game using an artificial neural network (ANN). The second use case operates on the visual component of our dataset to train an ANN to predict a visual frame in the future. We provide all data via GitHub[1]. The repository contains all data, annotations, our pre-processing pipeline as Python scripts, and the source code to both ANNs showcased in this work. We encourage researchers to build on our work and either expand the presented dataset or use it in novel approaches to game research.

2 BACKGROUND

For some time, datasets were mostly published as a by-product of a publication. However, nowadays, datasets are becoming increasingly important and even can be considered the primary intellectual output of research [6]. Dekker [6] reasons that the key rationale for the relevance of datasets can be summarized in four points: (1) replication and verification of publications, (2) longitudinal research, (3) interdisciplinary use of data, and (4) value enhancement through cooperation and follow-up work. All of Dekker's [6] proposed rationals are crucial to the advance of gaming research. One game particularly relevant to the gaming research and especially to the e-sport scene is Counter-Strike: Global Offensive.

Counter-Strike: Global Offensive is a team-based, fast-paced first-person shooter (FPS) game published by *Valve* in 2012 [39]. In CS:GO, two teams of five players compete against each other. One team takes on the role of the so-called *Terrorists*, and the other team is called *Counter-Terrorists*. The terrorists win a round by either placing a bomb and thus detonating an objective or eliminating all counter-terrorists. The counter-terrorists goal is to prevent the terrorists from placing the bomb, either by guarding the objectives or eliminating the opposing team. Despite this seemingly simple premise, CS:GO is highly challenging and tactical, as it requires extensive know-how about the game world, game mechanics, and when to use which strategy to reach the objective. CS:GO is the research object in numerous publications. Lux et al. [26], for example, investigate how recorded CS:GO matches can be automatically summarized. This allows for the easy creation of highlight videos, which attract a larger audience than full-length videos. Other work, such as the work of Makarov and Ignatow [28] researches probabilistic methods to predict the winning team in CS:GO. Using their probabilistic model, the authors found that the individual player skill is useful in predicting the winning team. In similar work, Xenopoulos, Harish, and Claudio [41] propose a novel graph-based framework for CS:GO to value player actions in the game. Using their framework, the authors identify high-impact play and estimate the game's outcome uncertainty. Further, Park et al. [32] and Korotin et al. [20] investigated and assessed the gaze behaviour of amateur and professional CS:GO players to determine differences in both player groups. Other work using CS:GO as a research object investigates the effects of latency on player performance and game experience. Liu et al. [24, 25], for example, found that CS:GO is highly sensitive to latency. Previous work aids in understanding CS:GO players, their interaction with the game, and the game itself. However, previous work is highly specialized to their respective research question. None of it provides a annotated dataset of actual game play, which would allow researchers to investigate research questions in a data-driven manner.

[1]https://github.com/julian1198/csgo_dataset

3 DATA COLLECTION AND DATASET GENERATION

In this section, we describe how we collected and processed the data to generate our dataset. We conclude with a descriptive summary of the data gathered.

3.1 Data Collection Study

CS:GO, out of the box, offers a way to record gameplay sessions: *Point of View* (POV) recordings. POV replays are recorded from the player's perspective and during the gameplay. The recording has to be started manually via the in-game console. POV replays are superior to generic video material, such as Youtube videos, as they offer additional information about the game state for each frame recorded. The replays can be played back with the game itself. However, since a POV replay needs to be recorded while playing the game, there is no rich source of replay files on the Internet. Thus, to still be able to utilize the data richness of POV replays, we conducted a data collection study to gather said replay files.

3.1.1 Apparatus. We conducted a remote study since players of FPS games, especially high-skilled ones, heavily rely on their equipment and a familiar environment to perform at maximum capacity. Thus, in our study, the participants played CS:GO on their own devices using their mouse, keyboard, and headset.

3.1.2 Procedure and Tasks. Participants received a detailed e-mail about the study's procedure and tasks. After giving informed consent to the data collection, the participants started the study at any time they liked. The study was conducted online without the need for supervision by the experimenter. Conducting the data collection in a laboratory may have increased the experiment's internal validity but would have decreased the ecological validity. Since our goal was to create a dataset of real CS:GO players playing in an authentic and natural environment, we conducted the data collection in the wild. In the study, the participants had to complete a demographic questionnaire stating age, gender, occupation, and experience with CS:GO. After completing the questionnaire, the participants were instructed to play one match of CS:GO on the game map *De_Dust2* (one of the most played maps in the game). While playing, the participants manually recorded their gaming session via the in-game console. After the match, participants had to upload their POV replay to a Dropbox folder provided by the initial e-mail.

3.1.3 Participants. We collected data from 12 participants (all male and all right-handed). For the recruitment, we used our institution's mailing list. Participants were selected independently of age and gender but were screened for their skill and experience in CS:GO. CS:GO has an ELO-based ranking system with 19 internal ranks - 19 being the lowest and 1 being the highest possible. To participate, the participants had to be at least ranked *Gold Nova 1* (rank 12) in CS:GO. *Gold Nova 1* corresponds to about a middle skill [23] in CS:GO's internal ranking system. Participants were ranked *Gold*

Nova 3 (rank 10) (1 participant), *Master Guardian* (rank 6) (5 participants), *Distinguished Master Guardian* (rank 5) (4 participants) and *Legendary Eagle Master* (rank 2) (2 participants). Participants played CS:GO from 600 hours to 4.000 hours, with an average time played of 1864.08 hours (SD = 1042.69 hours). The participants' age ranged from 18 years to 27 years, with a mean age of 22.42 years (SD = 2.23 years). All participants were students at our institution and received one credit for their course of study for their participation in the data collection.

3.2 Dataset Generation

After gathering the POV replay files, we parsed all files to obtain (1) dedicated images of the gameplay for each player and gaming session, (2) additional game state information such as how much health the player has, whether the player is wearing armor, positional information about the player's avatar, which equipment the player is currently playing with, and if the player has spotted an enemy or was spotted by an enemy. Additionally, we also gathered data about the (3) player's mouse and keyboard input for each frame parsed.

3.2.1 Image Generation. To obtain individual images from the recorded replay files, we used *Half-Life AdvancedEffects* [8] (HLAE). HLAE is a creator toolkit for CS:GO, which allows content-creators to extract video material from replay files in multiple video formats. Additionally, it also provides the possibility to save images from a replay by parsing them frame-by-frame. We utilized the frame-by-frame function to generate individual images for each frame of the gaming session. Researchers using our dataset can export images in any dimension desired. Generating images for 30 minutes of CS:GO gameplay, which accounts for about 115 200 frames, took approximately 12 hours. This means, overall, the image generation of the gathered data took roughly 288 hours. In this time, we generated 2 764 800 unique images of CS:GO gameplay. Figure 1 shows three example images of gameplay generated from the gathered replay files. For easy handling and fast prototyping, we generated two down- and grey-scaled versions of the extracted images. Using Python and the Python library NumPy, we created two Pickles: One saving all images in 60×34 pixel and the other containing the images in 30×17 pixel.

3.2.2 Game State Data Generation. We used *demofile* [33] to extract textual game state information from the the replays. The tool works similarly to HLAE in processing replay files frame-by-frame. Using a listing-framework *demofile* reacts to predefined events, which then can be logged in a text file. Utilizing the tool, we extracted the following game state information of the gathered replay files: (1) *tickCount* - this corresponds to the frame count of the gathered images. Via *tickCount* it is possible to match textual game state information with the accompanying image of the dataset. (2) *teamNumber*, which shows the team affiliation (unassigned, *terrorist* or *counter-terrorist*). (3) *armor* states a player's current amount of armor. Players start with 0 *armor* which can be increased through special in-game items to up to 100. (4) *health* states how much health the player has left (0 - 100). Players die if their health is reduced to 0. (5) *placeName* indicates where in the game world the player currently is. (6) *hasC4* states if the player possesses the bomb needed to detonate the objective when playing as *terrorist*. (7) *hasDefuser* states if the player possesses a special game item that accelerates defusing a bomb. (8) *isScoped* indicates if the player currently uses a weapon with a scope. (9) *isSpotted* states if the player was spotted by an enemy, analogously (10) *hasSpotted* states whether the player has spotted an enemy. (11) *weapon* contains the currently used weapon.

3.2.3 Input Data Generation. To obtain the keyboard and mouse input for each frame, we, again, used *demofile*. Using the same event-listing approach, we extracted the following information about the users' input: (1) *tickCount* - this states to which frame this input data belongs. Using *tickCount* images and input data can be synchronized. Additionally, we recorded (2)*buttons* which contains a string of pressed buttons at this frame. And, lastly, we extracted how the mouse's pixel position changed in regards to the previous frame in X- and Y-coordinates(3) *mouseDeltaX* and (4) *mouseDeltaY*.

4 EXAMPLE USE CASES OF THE PRESENTED DATASET

In the following, we showcase two use cases of our dataset. The first example uses a similar approach to latency compensation as previous work [15, 35] and aims to predict the users' mouse movement using an ANN. The second example demonstrates a novel approach to frame-based prediction. This approach predicts the next frame based on a series of frames instead of predicting the users' input (as in the former example). Both approaches could be used to reduce processing time, for example, in cloud gaming. In both use cases we used *TensorFlow* [1] for developing the ANNs and *Optuna* [2] for hyperparameter optimization. Both developed ANNs were trained and evaluated using Google's *Colab Pro +* [14] which offers 52 GB RAM and 2 Nvida P100 each with 16 GB vRAM.

4.1 Mouse Movement Prediction

The first use case, the mouse movement prediction, is trained on the input part of our dataset. The goal in the use case is to predict the mouse delta of a future frame based on a fixed amount of previous consecutive frames. For this goal, we used a classical feed-forward ANN. Since it was initially unknown how many frames are optimal as prediction baseline, we defined it as a parameter for *Optuna* to tune. The optimization by *Optuna* revealed that a prediction baseline of 6 frames yields the best possible results. In conclusion, we defined that the ANN uses six frames as input to predict the mouse delta of every seventh frame as output. Further optimizing with *Optuna* showed that four hidden layers, with 48 units (1), 118 units (2), 196 units (3), and respectively 1 unit (4), are optimal. The optimization framework also suggested the use of *Adam* [19] as optimizer, a batch size of 355, the mean squared error (MSE) between the actual value and predicted output as loss function, and the use of *ReLu* [12] as activation function. The learning rate was adaptively adjusted using callback functions after each epoch. We used a 80/10/10 train/test/validate split to train the ANN on the textual data of our dataset. One epoch took about six seconds. Overall we trained the ANN for 3 000 epochs.

After training, we tested the prediction on the previously unseen validation set. The validation showed that the ANN achieved an

Figure 1: Shows three examples of images extracted from the gathered replay files of *Counter-Strike: Global Offensive* gameplay. The left shows a player scooping with a weapon to secure a passage, the middle shows a player running, and the right shows a player placing a bomb to detonate the objective.

error rate (MSE) of 7.88 pixel for *mouseDeltaX* and 1.93 pixel for *mouseDeltaY*. Thus, the ANN predicted the next frame's mouse position based on the previous six consecutive frames within a small error range.

4.2 Next-Frame Prediction

In the second use case, an ANN is trained to predict a frame in the future. Due to RAM limitations, we used the down-scaled 60 × 34 pixel *Pickle* in this example. Additionally, we reduced the amount of data feed in training and only used four games (instead of all 12 available) to limit RAM usage. Overall, this example was trained on 316 757 gameplay images. After optimization *Optuna* suggested a prediction baseline of 2 frames, a network structure with six hidden layers (layer 1: 568 units, layer 2: 731 units, layer 3: 329 units, layer 4: 1703 units, layer 5: 2550 units, and layer 6: 2040 units), and a batch size of 87. As activation function *Optuna* recommended *ReLu* for all hidden layers and a linear activation for the output layer. We used the MSE between the frame's actual pixel value and the pixel value of the predicted frame as loss function. The learning rate was adaptively adjusted using callback functions after each epoch. To prevent overfitting, we implemented a dropout [38] function with a dropout value of 0.2. The ANN was trained on 100 % of the previously defined data for 3 500 epochs, with each epoch taking about 24 s to train. As test set, we used the 60 × 34 pixel data of two gaming sessions not included in the training set (game 6 and game 7).

After training, we tested our model on the previously unseen test set. The test showed that the ANN achieved a MSE accuracy of 0.095. Pixel values in the down-scaled 60 × 34 pixel version of our dataset range from 0 to 1. Thus, a MSE of 0.095 corresponds to less than 1 % of error. Furthermore, our example shows that using frame data allowed the ANN to predict future frames with an error less than 1 %. Figure 2 shows two examples of predicted frames generated by the model (right) alongside the original image (left).

5 DISCUSSION

Our two use cases exemplify the use of our dataset for different scenarios. However, both examples offer a large number of directions for improvement. Thus, this section first discusses the dataset itself and how it could be further refined, expanded, and improved. Next, we discuss the two presented examples and possible improvements. Built on the use cases, we discuss implications and further uses of our dataset.

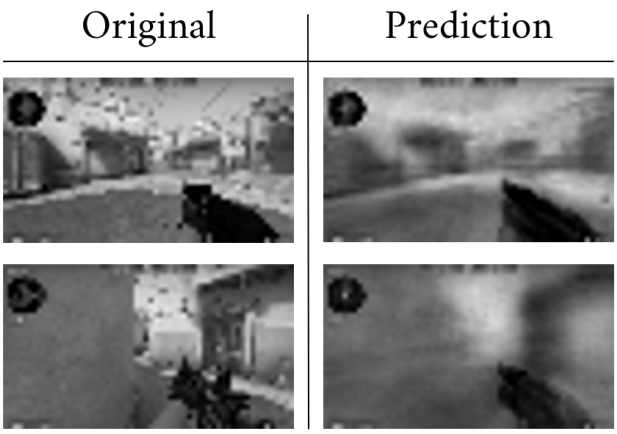

Figure 2: Shows two example images predicted by the next-frame prediction neural network (right), alongside the original image (left). The ANN generated the image based on two previous consecutive frames. The left side shows the actual image, which should be rendered after the two consecutive images used for prediction. The right shows the prediction. Although the prediction is blurry, architectural features such as walls, doors, and crucial game elements, such as the weapon and the mini-map, are recognizable.

5.1 Dataset

The dataset contains data from 12 high-skilled players playing CS:GO. However, we only recorded one particular game map. While some game elements are transferable, such as general player movement and weapon handling, some are map-specific. Thus, for example, some gameplay tactics are only applicable on one particular map and not on another map. However, considering that we provide our data via GitHub[2] and choose to use only open-source and easy-to-handle tools to generate the dataset, the presented dataset can be quickly and practical infinitely expanded, not only by us but by everyone interested. Our work may represent the starting point of a community-wide, large-scale data collection that could accelerate data-driven gaming research.

[2]https://github.com/julian1198/csgo_dataset

5.2 Use Cases

The mouse prediction example could be improved using recurrent neural networks (RNN). RNNs are a class of ANNs that can encode temporal information. Contrary to the classical feed-forward network used in the first use case, data is not only processed in one direction (input to output) but can be processed in both directions. This allows the RNN to model temporal dynamic behavior. RNNs have been successfully used in speech recognition [29], stock price prediction [36], and to detect emotions in conversations [27]. The presented feed-forward network does not utilize temporal information in its current form. However, it bases the prediction solely on the order in which the data is fed to the network. Using RNNs for the mouse prediction and thus, processing exact temporal information may improve the model's accuracy. Furthermore, the prediction could be improved by further fine-tuning hyperparameters. We used *Optuna* for hyperparameter tuning, which uses a sophisticated grid-search approach for optimization. Nevertheless, as in any deep learning-based approach, it is possible that our presented model did not optimize for a global but a local minimum. Using a more advanced optimizer instead of *Adam*, for example, *NAdam*, which employs Nesterov momentum in the training [7], could also enhance the mouse movement prediction. Similarly, the ANN for predicting the next frame could also be improved. We use a basic network architecture based on a Dense Neural Network (DNN) in the presented model. One way to improve the prediction is to use GANs. In a GAN architecture, two ANNs compete against each other. The gain of one network is coupled with the loss of the other network (zero-sum game) [13]. GANs generate new data from a given dataset through this interplay and have already been successfully used in creating artificial handwritten numbers [37] and in creating new game worlds in the computer game *Doom* [10, 11]. Using a GAN architecture could also increase the accuracy of the predicted frames in our presented use case.

5.3 Implication and Future Work

By using the presented dataset, researchers can now utilize deep learning in the context of CS:GO. We showcased two preliminary examples of deep learning-based latency compensation techniques. Researchers can build on our work to either refine our work or assemble novel approaches to reduce the adverse effects of latency in video games.

However, the possible applications of the presented dataset are far beyond latency compensation. Researchers, for example, can use it to investigate how high-skilled players of CS:GO behave in certain situations. Particularly high-pressure situations frequently occur in competitive multiplayer games such as CS:GO [5]. Often the outcome of a game round is determined by one player who has to stand their ground against a superior force, for example, more than one opponent or even the whole opposing team (1vs5 in CS:GO). Winning single-handed against such a superior force is called *Clutch*. By reviewing the demo files provided with our data, researchers can determine *Clutch* situation, analyze them accordingly, and potentially investigate what circumstances led to it and how players behaved. Ultimately, this could deepen our knowledge about games, gamers, and their interaction with the game.

Similarly, as we did in our examples, researchers can use the presented dataset to prototype novel deep learning methods for games. These methods do not have to be focused on latency compensation as ours did but can be used for a broad range of applications. Researchers, for example, could utilize the visual material for a classical task in computer vision, such as object detection or object recognition. One may train, for instance, an ANN to detect or anticipate enemies. Training a model on a large enough dataset may allow the model to anticipate where enemies are about to appear based on previously learned behaviors. In combination with a good software toolkit, this ANN could, for example, be used to train novice gamers - helping them to increase game awareness and to improve their gaming skills overall [42]. While the presented dataset is still small, the potential future applications of it are diverse and limited only by the creativity and endeavors of the researchers.

5.4 Conclusion

In this work, we present a novel Counter-Strike:Global Offensive dataset. We provide three different versions of the visual material. Additionally, the presented dataset is highly annotated, providing game and input information. We also presented two preliminary examples of the use of our dataset. Finally, in our discussion, we explore our dataset's possible future expansion and application.

REFERENCES

[1] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. 2015. TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems. https://doi.org/10.5555/3026877.3026899 Software available from tensorflow.org.

[2] Takuya Akiba, Shotaro Sano, Toshihiko Yanase, Takeru Ohta, and Masanori Koyama. 2019. Optuna: A Next-generation Hyperparameter Optimization Framework. In *Proceedings of the 25rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*.

[3] Javier Artiles and Satoshi Sekine. 2022. Tagged and Cleaned Wikipedia. https://nlp.cs.nyu.edu/wikipedia-data/. Accessed on 2021-01-01.

[4] Alejandro Baldominos, Yago Saez, and Pedro Isasi. 2019. A survey of handwritten character recognition with mnist and emnist. *Applied Sciences* 9, 15 (2019), 3169.

[5] Nicole A Beres, Madison Klarkowski, and Regan L Mandryk. 2021. Under Pressure: Exploring Choke and Clutch in Competitive Video Games. *Proceedings of the ACM on Human-Computer Interaction* 5, CHI PLAY (2021), 1–22.

[6] Ronald Dekker. 2006. The importance of having data-sets. (2006).

[7] Timothy Dozat. 2016. Incorporating nesterov momentum into adam. (2016).

[8] Half-Life Advanced Effects. 2021. *Half-Life Advanced Effects.* https://www.advancedfx.org/

[9] Adhesh Garg, Diwanshi Gupta, Sanjay Saxena, and Parimi Praveen Sahadev. 2019. Validation of random dataset using an efficient CNN model trained on MNIST handwritten dataset. In *2019 6th International Conference on Signal Processing and Integrated Networks (SPIN)*. IEEE, 602–606.

[10] Edoardo Giacomello, Pier Luca Lanzi, and Daniele Loiacono. 2018. Doom level generation using generative adversarial networks. In *2018 IEEE Games, Entertainment, Media Conference (GEM)*. IEEE, 316–323.

[11] Edoardo Giacomello, Pier Luca Lanzi, and Daniele Loiacono. 2019. Searching the latent space of a generative adversarial network to generate Doom levels. In *2019 IEEE Conference on Games (CoG)*. IEEE, 1–8.

[12] Xavier Glorot, Antoine Bordes, and Yoshua Bengio. 2011. Deep sparse rectifier neural networks. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*. JMLR Workshop and Conference Proceedings, 315–323.

[13] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. 2016. *Deep learning*. MIT press.

[14] Google. 2017. *Google Colab.* https://colab.research.google.com/

[15] David Halbhuber, Niels Henze, and Valentin Schwind. 2021. Increasing Player Performance and Game Experience in High Latency Systems. *Proceedings of the ACM on Human-Computer Interaction* 5, CHI PLAY (2021), 1–20.

[16] Md Kamrul Hasan, Md Toufick E Elahi, Md Ashraful Alam, Md Tasnim Jawad, and Robert Martí. 2022. DermoExpert: Skin lesion classification using a hybrid convolutional neural network through segmentation, transfer learning, and augmentation. *Informatics in Medicine Unlocked* (2022), 100819.

[17] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 770–778.

[18] John Jumper, Richard Evans, Alexander Pritzel, Tim Green, Michael Figurnov, Olaf Ronneberger, Kathryn Tunyasuvunakool, Russ Bates, Augustin Žídek, Anna Potapenko, et al. 2021. Highly accurate protein structure prediction with AlphaFold. *Nature* 596, 7873 (2021), 583–589.

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

[20] Alexander Korotin, Nikita Khromov, Anton Stepanov, Andrey Lange, Evgeny Burnaev, and Andrey Somov. 2019. Towards understanding of esports athletes' potentialities: The sensing system for data collection and analysis. In *2019 IEEE SmartWorld, Ubiquitous Intelligence & Computing, Advanced & Trusted Computing, Scalable Computing & Communications, Cloud & Big Data Computing, Internet of People and Smart City Innovation (SmartWorld/SCALCOM/UIC/ATC/CBDCom/IOP/SCI)*. IEEE, 1804–1810.

[21] KP. 2021. *CS:GO Competitive Matchmaking Data.* https://www.kaggle.com/skihikingkevin/csgo-matchmaking-damage

[22] Yann LeCun. 1998. The MNIST database of handwritten digits. *http://yann. lecun. com/exdb/mnist/* (1998).

[23] Leetify. 2021. *CSGO Rank Distribution for 2021.* https://blog.leetify.com/csgo-rank-distribution/

[24] Shengmei Liu, Mark Claypool, Atsuo Kuwahara, Jamie Sherman, and James J Scovell. 2021. Lower is better? The effects of local latencies on competitive first-person shooter game players. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*. 1–12.

[25] Shengmei Liu, Atsuo Kuwahara, James Scovell, Jamie Sherman, and Mark Claypool. 2021. Comparing the Effects of Network Latency versus Local Latency on Competitive First Person Shooter Game Players. (2021).

[26] Mathias Lux, Pål Halvorsen, Duc-Tien Dang-Nguyen, Håkon Stensland, Manoj Kesavulu, Martin Potthast, and Michael Riegler. 2019. Summarizing E-sports matches and tournaments: The example of counter-strike: Global offensive. In *Proceedings of the 11th ACM Workshop on Immersive Mixed and Virtual Environment Systems*. 13–18.

[27] Navonil Majumder, Soujanya Poria, Devamanyu Hazarika, Rada Mihalcea, Alexander Gelbukh, and Erik Cambria. 2019. Dialoguernn: An attentive rnn for emotion detection in conversations. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 33. 6818–6825.

[28] Ilya Makarov, Dmitry Savostyanov, Boris Litvyakov, and Dmitry I Ignatov. 2017. Predicting winning team and probabilistic ratings in "Dota 2" and "Counter-Strike: Global Offensive" video games. In *International Conference on Analysis of Images, Social Networks and Texts*. Springer, 183–196.

[29] Yajie Miao, Mohammad Gowayyed, and Florian Metze. 2015. EESEN: End-to-end speech recognition using deep RNN models and WFST-based decoding. In *2015 IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*. IEEE, 167–174.

[30] Hardik Nahata and Satya P Singh. 2020. Deep learning solutions for skin cancer detection and diagnosis. In *Machine Learning with Health Care Perspective*. Springer, 159–182.

[31] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y Ng. 2011. Reading digits in natural images with unsupervised feature learning. (2011).

[32] Eunji Park, Sangyoon Lee, Auejin Ham, Minyeop Choi, Sunjun Kim, and Byungjoo Lee. 2021. Secrets of Gosu: Understanding Physical Combat Skills of Professional Players in First-Person Shooters. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*. 1–14.

[33] Saul Rennison. 2021. *demofile.* https://saul.github.io/demofile/

[34] Gabriel Salles. 2021. *CS:GO Professional Matches.* https://www.kaggle.com/gabrieltardochi/counter-strike-global-offensive-matches

[35] Valentin Schwind, David Halbhuber, Jakob Fehle, Jonathan Sasse, Andreas Pfaffelhuber, Christoph Tögel, Julian Dietz, and Niels Henze. 2020. The Effects of Full-Body Avatar Movement Predictions in Virtual Reality using Neural Networks. In *26th ACM Symposium on Virtual Reality Software and Technology*. 1–11.

[36] Sreelekshmy Selvin, R Vinayakumar, EA Gopalakrishnan, Vijay Krishna Menon, and KP Soman. 2017. Stock price prediction using LSTM, RNN and CNN-sliding window model. In *2017 international conference on advances in computing, communications and informatics (icacci)*. IEEE, 1643–1647.

[37] Konstantin Shmelkov, Cordelia Schmid, and Karteek Alahari. 2018. How good is my GAN?. In *Proceedings of the European Conference on Computer Vision (ECCV)*. 213–229.

[38] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research* 15, 1 (2014), 1929–1958.

[39] Steam. 2022. Counter-Strike: Global Offensive. https://store.steampowered.com/app/730/CounterStrike_Global_Offensive/. Accessed on 2021-01-01.

[40] Steamcharts. 2022. An ongoing analysis of Steam's concurrent players. https://steamcharts.com/app/730. Accessed on 2021-01-01.

[41] Peter Xenopoulos, Harish Doraiswamy, and Claudio Silva. 2020. Valuing player actions in counter-strike: Global offensive. In *2020 IEEE International Conference on Big Data (Big Data)*. IEEE, 1283–1292.

[42] Nina Zhou, Matt Doerner, Lisa Ryan, and Christopher Pierse. 2021. AI Coach for Battle Royale Games. In *Extended Abstracts of the 2021 Annual Symposium on Computer-Human Interaction in Play* (Virtual Event, Austria) *(CHI PLAY '21)*. Association for Computing Machinery, New York, NY, USA, 280–286. https://doi.org/10.1145/3450337.3483456

[43] Yiwei Zhou, Alexandra Cristea, and Zachary Roberts. 2015. Is wikipedia really neutral? A sentiment perspective study of war-related wikipedia articles since 1945. In *Proceedings of the 29th Pacific Asia Conference on Language, Information and Computation*. 160–168.