Deep Reinforcement Learning in Serious Games: Analysis and Design of Deep Neural Network Architectures

Aline Dobrovsky(✉), Cezary W. Wilczak, Paul Hahn, Marko Hofmann, and Uwe M. Borghoff

Fakultät für Informatik, Universität der Bundeswehr München, 85577 Neubiberg, Germany
{aline.dobrovsky,cezary.wilczak,paul.hahn,marko.hofmann, uwe.borghoff}@unibw.de

Abstract. Serious games present a noteworthy research area for artificial intelligence, where automated adaptation and reasonable NPC behaviour present essential challenges. Deep reinforcement learning has already been successfully applied to game-playing. We aim to expand and improve the application of deep learning methods in SGs through investigating their architectural properties and respective application scenarios. In this paper, we examine promising architectures and conduct first experiments concerning CNN design and analysis for game-playing. Although precise statements about the applicability of different architectures are not yet possible, our findings allow for concluding some general recommendations for the choice of DL architectures in different scenarios. Furthermore, we point out promising prospects for further research.

Keywords: Deep learning · Serious games
Convolutional neural networks · Neural network visualization

1 Introduction

Serious Games (SGs) rank among the most important future e-learning trends; they attain enhanced public acceptance and importance [3]. Although more frequently used in recruitment and training, their production is still effortful and expensive. The generation of human behaviour for non-player characters (NPCs), player identification, adaptivity to the player, content generation and general game playing remain prevalent challenges [1,10]. SGs can profit from the application of machine learning methods to create diverse behaviour and from automated adaptation to increase learning effectiveness. Deep learning (DL) methods and deep reinforcement learning (DRL) in particular demonstrated to be an opportunity for application. Considerable results have already been achieved by this general method. DL means machine learning methods using multiple processing layers, usually deep neural networks (DNN). DRL means the combination of reinforcement learning and DL. A famous example is deep Q-learning

R. Moreno-Díaz et al. (Eds.): EUROCAST 2017, Part II, LNCS 10672, pp. 314–321, 2018.
https://doi.org/10.1007/978-3-319-74727-9_37

for learning to play Atari games [13], where a convolutional neural network was trained with a variant of Q-learning on different games and partially outperformed human game players.

A general requirement for the application of DL in games is the demand for a cost-effective development process due to limited time and capacities. Furthermore, AI behaviour and its development have to be accessible to, comprehensible for and influenceable by different domain experts. Game AI has to face very different, prevalent challenges, depending on game and application scenario, including action-state space representation (from symbolic to raw data) and size, observability, non-determinism and credit assignment. Exemplary application areas are NPC control an the generation of human behaviour, human player replacement in multi-player games, automated adaptation of game content and progress, game-testing, etc.

We aim to expand and improve the application of DL in games through offering a novel framework including interactive learning. An overview of the original conceptual framework and its components is described in [2], although it has been fundamentally extended by adding the aspect of automated game adaptation by now. In this paper, we attempt to examine the usability of DL methods in SGs through literature research and first experiments. We investigate DNN architectures and show important factors considering their application.

2 Deep Learning Architectures

The term "architecture" summarizes all characteristics that define the structure of the deep neural network; including general structure, types of layers and connections, topology of different layers, weights and activation functions. This problem is also strongly related to hyperparameter optimization. In this work, however, we mean to rather explore general properties and aim to find general recommendations for using DL in SGs. Our methodical approach comprised literature analysis of different architectures, identification of further research prospects and experimental investigation of specific aspects. This section presents selected architectures with regard to existing promising practical applications of DL in different domains. The approaches listed here should serve as inspirational examples for related application possibilities in SGs.

2.1 Investigation of Exemplary Architectures

Feedforward architectures are already extensively used for classification and clustering tasks. Feedforward NNs are actually function approximators, and, according to the universal approximation theorem, a network of appropriate size can achieve a desired degree of accuracy [4]. Especially convolutional neural networks (CNNs) have recently attracted attention due to their successes in practical applications. CNNs consist of several layers of convolutions and in each layer, filters are applied to data and their respective values are trained. The structure is hierarchical: every layer can detect distinctive features of the input data

on another abstraction level. Besides, spare connections and parameter sharing improve the training time. Because of their structure, CNNs are principally designed for 2D data, but can be adapted to other dimensions as well. Applications include computer vision and acoustic modelling for automated speech recognition; examples for 1D, 2D and 3D image and audio data can be found in ([4], p. 349). A commendable example for supervised learning is given by the GoogLeNet Inception architecture [14] that won the ILSVRC (image classification challenge) 2014. The handcrafted architecture comprises 27 layers, whereby inception modules count as one layer. A main goal was to increase depth and width of the network while trying to keep an acceptable computational effort.

CNNs can be used in the area of unsupervised learning as well, e.g. for unsupervised visual representation learning in videos [16]. The key issue of this work is that the otherwise necessary supervisional feedback can be substituted by using visual tracking, assuming that connected patches in tracks should have similar visual representations. As architecture, a siamese-triplet network with a ranking loss function is used. Every triplet uses the same architecture as Alexnet [8], an image classification architecture that won the ILSVRC 2012. CNN architectures have already been successfully used in games in combination with reinforcement learning (RL). In 2015, Google presented an approach of using a combination of RL and DNN for playing Atari video games. Their input state representation was a simplified version of the raw pixels of the game screen and their output controller moves. They tested their approach on 49 Atari games. The experiments showed that their variant of deep-Q-Learning even reached human level performance on over more than half of the games [13]. Investigating this success, the architecture exploits the CNN 2D image recognition potential. Their method performed best in quick-moving games but poorer than human players in long-horizon games where planning and a specific sequence of actions is important.

In contrast to feedforward techniques, recurrent NN architectures possess an internal memory through directed, cyclic connections. Their structure is better suited for processing sequences of arbitrary length, context modelling and time dependencies. A RNN with hidden-hidden recurrent connections actually corresponds to a universal turing machine. As architecture, the long short-term memory (LSTM) model is commonly used, because it avoids the vanishing gradient problem. LSTM proved good at handwriting recognition, speech recognition, machine translation, image captioning and parsing [4]. A third, emerging, type of architecture is provided by deep generative models. In contrast to the former mentioned discriminative models, a generative architecture can be used for probability distribution representation and sample generation. Particularly generative adversarial networks (GANs) have recently shown impressive capabilities. A GAN architecture consists of a generator and discriminator component, whereby the discriminator evaluates content by estimating if its original data or created by the generator. GANs have been used, for example, to generate contextually matching image samples with mandated properties or text to image synthesis (cf. [5]).

Altogether, we found very few concrete game applications of DL; the major activities remain in academic research. However, the mentioned approach of Atari game-playing could also be used in SGs, in restricted scenarios, as NPC control or human replacement, and also for game testing purposes. An effective DRL application to a more complex game could be achieved through exploiting additional symbolic game information for training, as has been shown in a 3D shooter [9]. Another possible DL application area is demonstrated by [12], where stacked denoising autoencoders are used for player goal recognition, which is a player-modeling task. Moreover, in [11], CNNs are used to recognise emotions through face images for player affect analysis.

2.2 Conclusions and Further Research Prospects

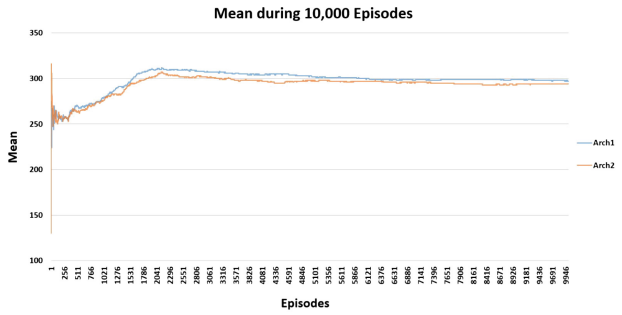
Our investigations indicate that, at the present time, suggesting concrete architectures for specific tasks is unfeasible. The suitability of an architecture is dependent on a variety of different criteria, e.g. amount and type of training data, reward function, computational and time limits, convergence properties and comprehensibility of learning behaviour. For the time being, deciding for a concrete architecture remains a trial and error process (cf. [4] p. 192). Though, we can still derive general ideas of suitable application areas for the different DNN types. CNNs are obviously suitable for all image recognition related tasks and present a good starting point, because efficient exemplary implementations and sophisticated frameworks are available. A RL approach is commonly used in games because of the lack of labelled training data. Although the learning efficiency continually increases through refining architectures and learning algorithms, long training times and uncertain results are still to be expected. Application opportunities in (visually) complex game scenarios are supported by input preprocessing or additional game information. For preprocessing, DL techniques like autoencoders can be considered. Furthermore, non-visual game information can be transformed into visual representation. We assume that e.g. simplified top-down views on strategy game scenarios are interpretable by a CNN. This can be extended by 2D presentations of context information, like in the upcoming Google Startcraft2 DL project [15]. The use of RNN in games seems to be still at the beginning. Applications for translation and language modelling are quite conceivable. Additionally, generating NPC behaviour in combination with CNN as shown in [9] is encouraging. Generative models seem promising for application in the prevalently challenging area of automated (procedural) content generation.

3 Experiments

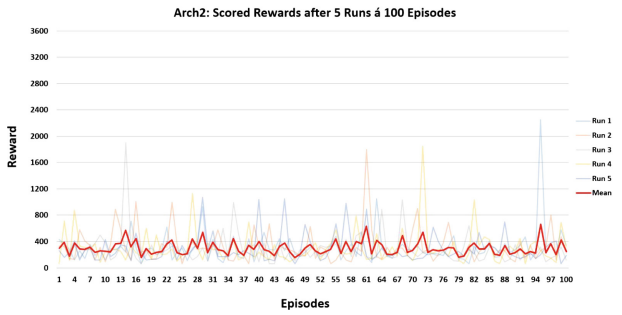
Within the scope of two bachelor theses [6,17] we examined the DQN approach of [13] in video games concerning architectural structure and analysis possibilities of CNNs. We used Tensorflow on a desktop PC on the Atari games Ms PacMan and Breakout. As input, the NN received a scaled, grayscale screenshot of the game and had to compute actions as Atari controller moves.

3.1 Architecture Comparison

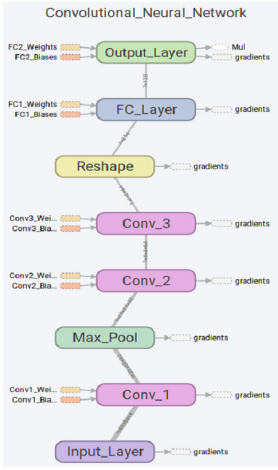
We compared two architectures during 2 × 2 days of training to identify the influence of an additional pooling layer. Figure 1(c) shows the architecture *Arch2* with an extra pooling layer. In contrast, *Arch1* didn't have the 3rd layer and was therefore 12.25× bigger than *Arch2*. During the first two days of training, both architectures' scores rose to 300 game reward points on average and stagnated subsequently (Fig. 1(a)). This performance corresponds roughly to a human beginner. The maximum value, measured each 100 episodes, was regularly between 1000 and 2000, with spikes of 3870 (*Arch1*) and 4370 (*Arch2*). During the same time (2 runs in 4 days) and with the same configuration, *Arch1* trained 23 million timesteps and *Arch2* 29 million. In the test runs after 4 days of training (5 × 100 episodes without learning), *Arch1* reached a maximum episode mean value of 3380, and *Arch2* of 2250; with episode means in general between 200 and 400 (Fig. 1(b)).



(a) *Arch2* training means over 10000 episodes



(b) *Arch2* mean rewards in 5 test runs



(c) Architecture *Arch2* with Pooling Layer

Fig. 1. Architecture comparison

3.2 Architecture Analysis

For architecture analysis, we examined different visualization methods and applied the method of Harley (cf. [7]) to a network trained on Breakout. Visualization can reveal shortcomings of architecture choice and training process.

For example, noisy filters could mean an unconverged network or a wrongly adjusted learning rate. By contrast, clean structures are an indication for effective training. The layer activation visualization comprehensibly shows activated neurons and reveals adaptation to specific features, e.g. detection of particular game objects.

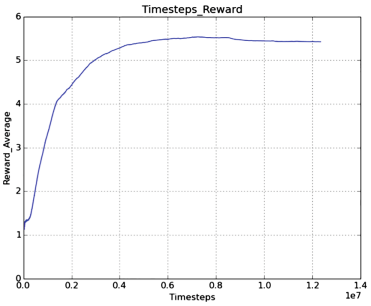


Fig. 2. Average game reward during 12 million timesteps

Fig. 3. Exemplary layer 1 filter at the beginning of training

Fig. 4. Same filter after 2 million timesteps

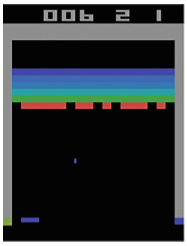


Fig. 5. Game situation in Breakout

Fig. 6. Activation maps of layer 1 at the beginning of training

Fig. 7. Activation maps of layer 1 after 2 million timesteps

As in the first experiment, learning stagnated relatively early in training (Fig. 2). Figures 3 and 4 show the first of four 8×8 filters of the first layer before and during the training. Figures 6 and 7 show the 32 activation maps (due to 32 filters) of the first convolutional layer in the game state of Fig. 5. The filter weights exhibit emerging of a clear structure and the feature maps show that important game features, the paddle and the ball, are recognized.

3.3 Experiment Conclusions

As already mentioned, CNN architectures are a mighty tool, but their design relies on assumptions and a process of trial and error. A lot of configuration

options are available that heavily influence the learning performance (from general architecture design to specific parameters and different learning algorithms and reward function). We assume the latter and small computing capacity to be the reason why we couldn't achieve Google's performance. Additionally, we imagine Ms Pacman to be a challenging object of investigation because of contextual game-state change (ghost vulnerability isn't clearly graphically represented). Regarding architectures, bigger models could possibly convey more distinct state features in games but would also require more training than the simplistic approach. An encouraging finding is that our additional pooling layer had no negative influence on performance but allowed for faster training time. DNN largely remain blackboxes but visualization can offer a significant improvement in interpreting their behaviour. Visualization entails the potential of a powerful developer tool; it can help to optimize nets and reduce time for fault diagnostics. Comprehensive toolboxes are already available and can potentially be used as an essential part of neural network training. Nonetheless, interpretation by experienced persons is still necessary to detect anomalies. Furthermore, various other not visualisable possible sources of error exist.

4 Conclusions and Future Work

We aimed to inquire the usability of DL in SGs and investigated if general recommendations for applying DL in diverse SG application scenarios can be derived dependent on architecture. We presented selected architectures and illustrated their apparent and also imaginable further application possibilities. In our experiments, we examined architecture design and comparison and presented visualization as a method for NN analysis. Although we have to conclude initially that we can't offer precise statements about the applicability of different architectures, we nonetheless found promising first recommendations for further research. Architectures like CNNs, LSTMs, autoencoders and GANs already exhibit amazing successes and could probably be used for NPC control, player modelling or content creation. Some game adaptations could also improve the applicability of DL, e.g. creating a visual state representation or using preprocessing autoencoders for CNN application. Although DL methods aren't popular in the SG scene yet, we are convinced that our findings can inspire further research for beneficial applications in games.

References

1. Brisson, A., Pereira, G., Prada, R., Paiva, A., Louchart, S., Suttie, N., Lim, T., Lopes, R., Bidarra, R., Bellotti, F., et al.: Artificial intelligence and personalization opportunities for serious games. In: Eighth AIIDE Conference, vol. 2012, pp. 51–57 (2012)
2. Dobrovsky, A., Borghoff, U.M., Hofmann, M.: Applying and augmenting deep reinforcement learning in serious games through interaction. Periodica Polytech. Electr. Eng. Comput. Sci. **61**(2), 198–208 (2017). https://doi.org/10.3311/PPee.10313. ISSN 2064-5279

3. Doujak, G.: Serious Games und Digital Game Based Learning. Spielebasierte E-Learning Trends der Zukunft. GRIN Verlag, Munich (2015)
4. Goodfellow, I., Bengio, Y., Courville, A.: Deep Learning. Adaptive Computation and Machine Learning Series. MIT Press, Cambrige (2017)
5. Goodfellow, I.J.: NIPS 2016 tutorial: generative adversarial networks. CoRR abs/1701.00160 (2017). http://arxiv.org/abs/1701.00160
6. Hahn, P.: Visualisierung von Convolutional Neural Networks. Bachelorarbeit, Universität der Bundeswehr München (2017)
7. Harley, A.W.: An interactive node-link visualization of convolutional neural networks. In: Bebis, G., et al. (eds.) ISVC 2015. LNCS, vol. 9474, pp. 867–877. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-27857-5_77
8. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. In: Pereira, F., Burges, C.J.C., Bottou, L., Weinberger, K.Q. (eds.) Advances in Neural Information Processing Systems 25, pp. 1097–1105. Curran Associates, Inc. (2012)
9. Lample, G., Chaplot, D.S.: Playing FPS games with deep reinforcement learning. arXiv preprint arXiv:1609.05521 (2016)
10. Lara-Cabrera, R., Nogueira-Collazo, M., Cotta, C., Fernández-Leiva, A.J.: Game artificial intelligence: challenges for the scientific community. In: Proceedings 2st Congreso de la Sociedad Española para las Ciencias del Videojuego Barcelona, Spain (2015)
11. Martinez, H.P., Bengio, Y., Yannakakis, G.N.: Learning deep physiological models of affect. IEEE Comput. Intell. Mag. **8**(2), 20–33 (2013)
12. Min, W., Ha, E., Rowe, J.P., Mott, B.W., Lester, J.C.: Deep learning-based goal recognition in open-ended digital games. In: AIIDE. Citeseer (2014)
13. Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A.A., Veness, J., Bellemare, M.G., Graves, A., Riedmiller, M., Fidjeland, A.K., Ostrovski, G., et al.: Human-level control through deep reinforcement learning. Nature **518**(7540), 529–533 (2015)
14. Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., Rabinovich, A.: Going deeper with convolutions. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1–9 (2015)
15. Vinyals, O.: DeepMind and Blizzard to release Starcraft II as an AI research environment (2016). https://deepmind.com/blog/deepmind-and-blizzard-release-starcraft-ii-ai-research-environment/. Accessed 23 May 2017
16. Wang, X., Gupta, A.: Unsupervised learning of visual representations using videos. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 2794–2802 (2015)
17. Wilczak, C.: Convolutional Neural Networks: Betrachtung und Vergleich verschiedener Architekturen und ihrer Merkmale in Computerspielen. Bachelorarbeit, Universität der Bundeswehr München (2017)