Université Paris-Dauphine

Mathématiques et Informatique de la Décision et des Organisations

Minimalistic Residual Network for Computer Go: RemGo

Authors:
Hosseinkhan Rémy

Advisor:
Cazenave Tristan

M2 Intelligence Artificielle Systèmes Données

March 2020

Acknowledgements

It was a real pleasure to discover the Computer Go universe through this project while building a value-policy network using Tensorflow 2: more than 30 different models have been created with many training hours to obtain an efficient agent. Thanks to Tristan Cazenave for proposing this research topic that has opened our scientific minds. A lot of knowledge has been gained through this work.

Contents

1 Model selection

In order to choose the model architecture, a manual grid-search was performed on a relatively small dataset (700 000 observations) from Facebook ELF opengo Go program self played games [1]. Each of the tested models derive from the model presented in the *Residual Networks for Computer Go* paper [2].

Basically these are two-headed (one for policy and one for value) residual networks whose residual layer number goes from 3 to 7 (depth property) and their number of filter are choosed between 42 and 128 (width property). The underlying challenge this project involves is to re-scale (since a limit in the parameter number is fixed to 1 million) and slightly improve efficient architectures that already exist. The training is done for 20 epochs to globally evaluate the learning potential of each architecture. Different optimizers have been used to perform the gradient descent: SGD, Nesterov Momemtum as it is suggested in [3] and Adam because of its fast convergence property.

Only a sample of the different models is presented in the figure 1 and 2.
To be concise, only losses are shown but the same conclusion concerning the best model selection would be made by studying the accuracy graphs.

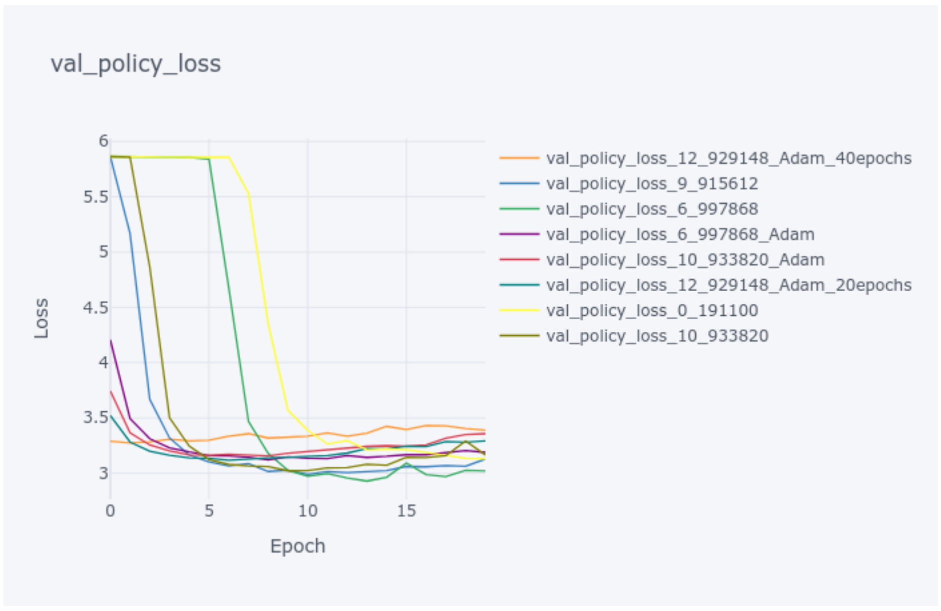

Figure 1: Validation policy accuracy of a subset of the tested models.

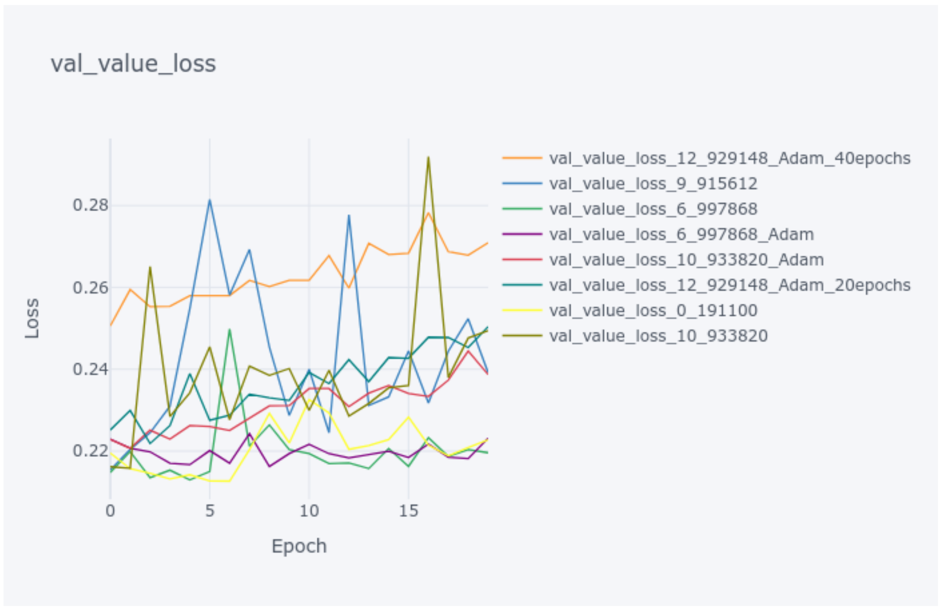

Figure 2: Validation value accuracy of a subset of the tested models.

2 RemGo5

From the above study, the first selected model (green curve with id: 6_997868) is the one having the best performances on the validation set. Its residual layer is presented in the following figure.

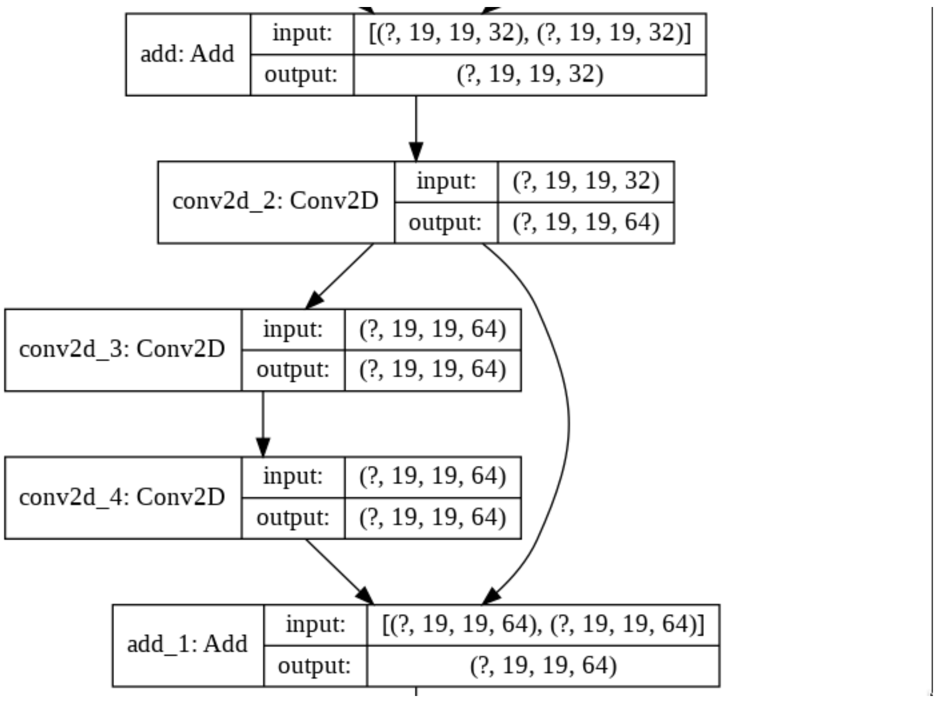

Figure 3: Residual block of RemGo5

The residual layer of RemGo5 performs one transformation (1x1 convolution) before two traditional 5x5 convolutions and consider the result of the 1x1 convolution as the input in the addition layer instead of the classical input. Note that at his point, there is no batch normalization. The model has **4 residual blocks of 64 filters** and has a 16 filter layer just before entering in the value and policy heads in order to reduce the dimension before flattening. Those value and policy head use 1 filter convolution layer before the signal is flattened. Its input layer use the same principle but has one only convolution. It has 997,868 weights. This given model, RemGo5 is trained over 200 batches of 500 000 observations with Nesterov SGD and a learning rate of 0.01 then 0.001 after 100 epochs. Its accuracy on validation set is presented later in table 3. RemGo5 seems to have a strong potential since it has performances close to Golois in tournaments despite its poor training (due to our lack of knowledge at this point of the experiments).

3 RemGo8

The next and final model RemGo8 use the same (1x1 convolution) principle but is augmented by batch normalization as in [2]. The convolutions in the residual blocks use 3x3 kernels as it is common practice in computer Go. Also, the value head contains now multiple average pooling to reproduce results from [4] and [5]. The model has **3 residual blocks of 108 filters** but keep almost the same number of non-linearities because it has a slightly different structure. Here, the network has classical addition shortcuts. A non-linearity is added within blocks through 1x1 convolutions. As it was previously experimented, a 44 filter layer is set just before entering in the value and policy heads in order to reduce the space dimension before flattening. The residual layer and both the policy and value head are presented in the figures 4 and 5 below:

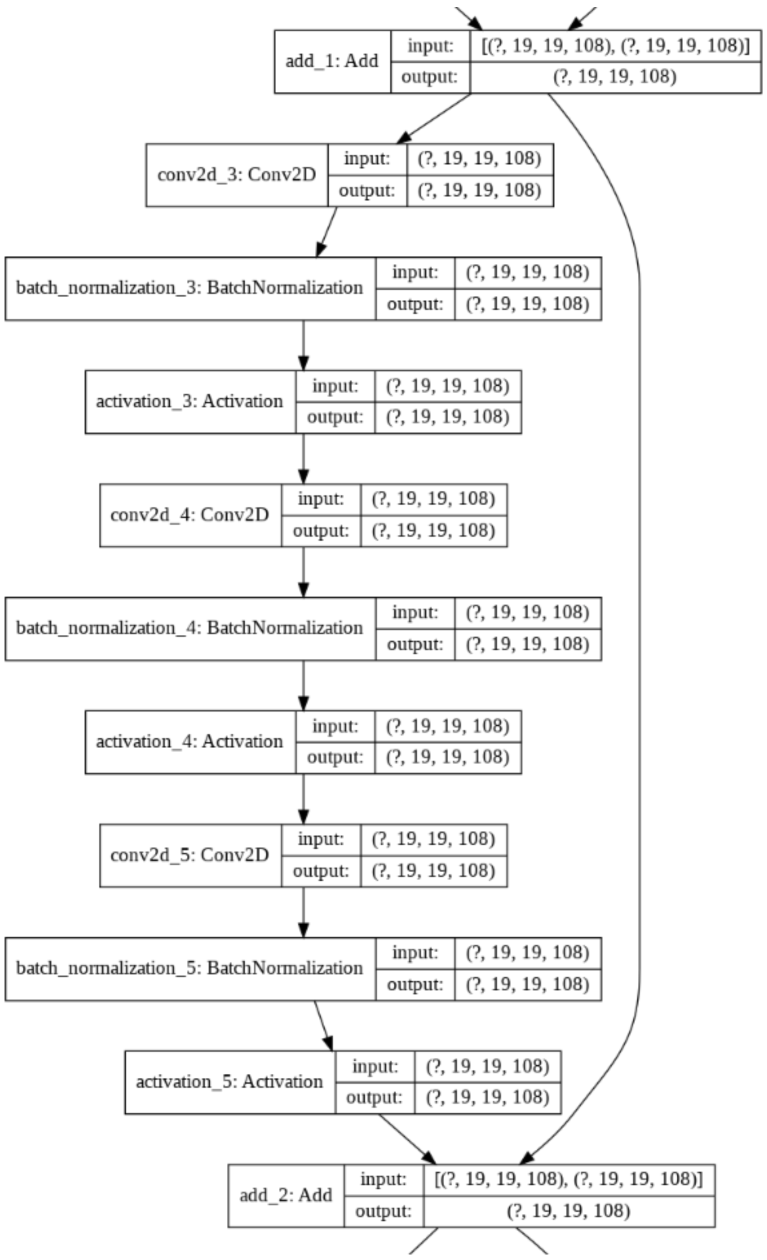

Figure 4: RemGo8 residual layer

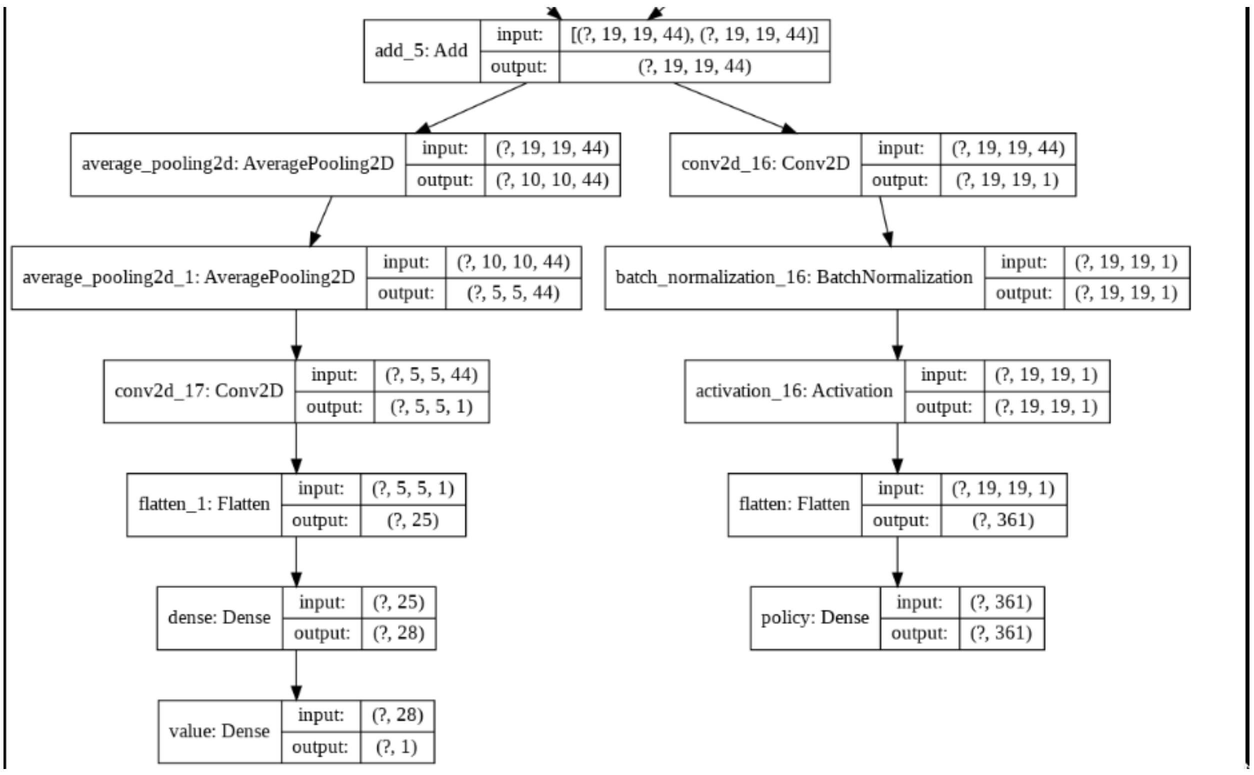

Figure 5: RemGo8 value and policy heads

The model is trained over 330 batches of size 500 000 with Nesterov SGD and a learning rate of 0.01 then 0.001 after 180 epochs. Note SGD has been preferred to Adam optimizer since it has not produced very accurate models, see [6]. One observe that effectively, the average pooling has improved value accuracy. This model was ranked higher than Golois for a single tournament. Note that some models having higher policy accuracies have been trained but they did not performed well during tournaments, thus only results for those both models are presented.

RemGo5	0.4015	0.6556
RemGo8	0.4078	0.6611

Table 1: Accuracies of the two models

4 Conclusion

Two models were built in order to reach golois tournament performances. Huge training and data understanding is require to perform in this task. Overfitting kills model predictions during games since no reasonable amount of data can describe the entire Go complexity.

References

[1] Tristan Cazenave. *Deep Learning Project*. 2020. URL: `https://www.lamsade.dauphine.fr/~cazenave/DeepLearningProject.html`.

[2] Tristan Cazenave. *Residual Networks for Computer Go*. 2017. URL: `https://www.lamsade.dauphine.fr/~cazenave/papers/resnet.pdf`.

[3] M.Sarigül and M.Avci. *Performance comparison of different momentum techniques on deep reinforcement learning*. 2018.

[4] Tristan Cazenave. *Spatial Average Pooling for Computer Go*. 2019. URL: `https://www.lamsade.dauphine.fr/~cazenave/papers/sap.pdf`.

[5] Jane Street Group David J. Wu. *Accelerating Self-Play Learning in Go*. 2020. URL: `https://arxiv.org/pdf/1902.10565.pdf`.

[6] A. C. Wilson R. Roelofs M. Stern N. Srebro B. Recht. *The Marginal Value of Adaptive Gradient Methods in Machine Learning*. 2017. URL: `https://arxiv.org/abs/1705.08292`.