Masked Modeling in Chess

Seyed Sepehr Seyed Ghasemipour
Department of Computer Science
University of Toronto
{sepehr.seyedghasemipour}@mail.utoronto.ca

Abstract

Masked modeling has found much success as a representation learning strategy for natural language and computer vision applications. This paper explores the possibilities for masked modeling in chess. We construct our investigation through a piece prediction task where we mask squares on a chess board and ask our model to predict the identity of the masked squares. We evaluate the learned representations from this procedure by applying them to two different downstream tasks. We find that the representations do not improve downstream task performance mainly because the pretraining task proves to be more trivial than expected: with 100% of pieces masked, our model predicts piece identity with over 70% accuracy. We note that the pretraining procedure yields the most non-trivial representation when 50% of the pieces are masked, following a similar trend to masking techniques in other domains.

1 Introduction

Masked modeling has seen promise across a range of different domains. In natural language processing, predicting masked words in sentences has resulted in models that demonstrate an understanding of semantics in text. Similarly in computer vision, filling in masked patches in images has resulted in models that portray an understanding of visual concepts. Such masked techniques show great potential in obtaining general representations: representations that capture the underlying structure of the domain and that, through methods such as transfer learning, can be fine tuned to more specialized tasks. This paper is based on this observation and seeks to explore it in the context of chess.

2 Related works

Transfer Learning Transfer learning is one of the central methodologies in deep learning. Pretraining on related tasks enables neural networks to learn robust features that can be transferred to downstream problems, and often result in significant performance gains even with little data availability for downstream tasks. Thus, pretraining and transfer learning have become key enablers for the practical application of neural networks in many fields. In computer vision, pretraining on classification datasets such as ImageNet results in improved performance on new classification tasks [Kolesnikov et al., 2019]. Unsupervised pretraining techniques such as SimCLRv2 [Chen et al., 2020] enable models to obtain competitive ImageNet classification performance while finetuning on only 1% of the ImageNet dataset. In NLP, it is now the de facto standard to use pretrained word embedding such as BERT [Devlin et al., 2018] when training models for new language task. Pretraining useful representations can also be done by drawing upon multiple modalities. In CLIP [Radford et al., 2021], image-caption pairs are used to train an image and language encoder to produce similar representations. The resulting representations learn intriguing features drawing information from both modalities, and have found many downstream applications such as language-conditional image generation [Ramesh et al., 2022] as well as robotics [Shridhar et al., 2022].

Preprint. Under review.

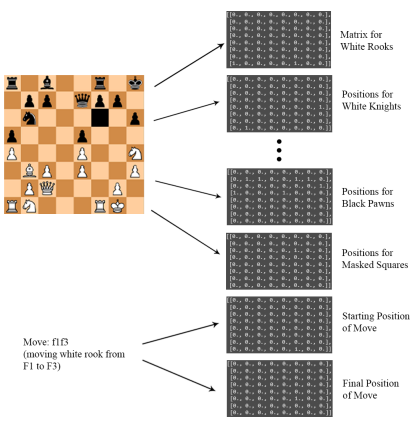

Figure 1: Matrices describing the positions of each chess piece, the masked pieces, and the move played

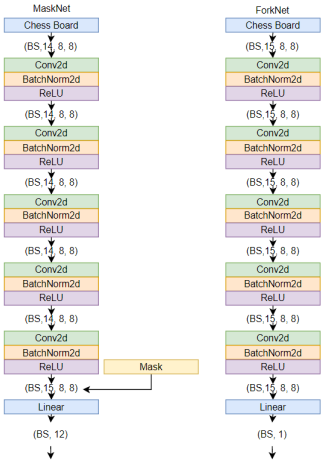

Figure 2: MaskNet and ForkNet architectures

Masking and Reconstruction as Pretraining Tasks Reconstructing masked or noisy inputs have been widely used as pretraining tasks in a variety of domains. The motivation for such approaches is that in order to accurately reconstruct masked or corrupted inputs, models would need to learn robust features about the input domain. Early works trained autoencoders to reconstruct inputs that were corrupted by random noise [Vincent et al., 2008]. These works, dubbed denoising autoencoders, were used to obtain robust features that could be used to initialize the layers of neural networks for downstream tasks such as classification. In NLP, Word2Vec [Mikolov et al., 2013] and BERT [Devlin et al., 2018] are two popular methods for training word-embeddings based on reconstructing masked inputs. Word2Vec operates by training a classification model to predict the word that belongs in the middle of an input N-gram. One of the key components of BERT is to arbitrarily mask words in a sentence, and train the model to fill in the blanks. In recent Computer Vision literature, self-supervised pretraining methods based on masking have lead to state-of-the-art results on downstream tasks. One such approach is the work of He et al. [2021] wherein large portions of images are masked out, and the models are trained to reconstruct the masked portions of the images. Masking techniques have also found valuable use cases in other domains such as Graph Neural Networks. Hu et al. [2019] present a graph pretraining approach wherein nodes and edges of graphs are masked, and models are trained to reconstruct their features. Representations learned through this pretraining approach lead to state-of-the-art results in downstream graph problems such as predicting molecule attributes.

3 Method

We apply a masked modeling approach in attempt to obtain a general representation for chess boards. Given a chess position, we mask a predefined ratio of pieces. Accompanied by the move that was played in that position, we then ask our model to predict the identity of the masked pieces. We hypothesize that solving this task will equip the masked model with an overall understanding of chess. To test this, we transfer the learned representations to two downstream tasks and compare the results to end to end training.

3.1 Pre-Training

Data. We represent a chess board by a 12 x 8 x 8 tensor: one channel for each chess piece type. In this tensor an entry will be 1 only if the associated piece type is present in that square, and 0 otherwise. To construct the input to our neural networks, first we randomly mask pieces from the board. Specifically, for each piece in the board, we decide whether to mask the piece by sampling from a Bernoulli distribution with probability MASK_PROB. We then concatenate 3 additional channels to the masked board. The first two additional channels represent the move that was played – one-hot for start and end square respectively. The third channel represents which squares have been masked: 1's for masked and 0 otherwise. Figure 1 gives a visual representation of the data structure.

MaskNet: a model to predict masked pieces. We use a CNN architecture, MaskNet, to perform the multi-class classification task of predicting the masked pieces. It consists of five convolutional layers. Since our downstream tasks do not rely on masked squares, MaskNet uses the first 14 channels as input, and concatenates the mask channel after the fourth layer. As described below, in our downstream tasks we transfer MaskNet layers that are prior to the concatenation. A kernel size of 5 and 64 channel dimensions was used in all four convolution layers to ensure that the convolution receptive field covered the entire board. The final layer of MaskNet consists of a convolution layer with kernel size of 1 that results in an output tensor dimension of 12 x 8 x 8. Intuitively, this tensor represents classification logits for each square of the chess board.

3.2 Transfer Learning

Algorithm 1 Transferring MaskNet to ForkNet

for each puzzle P in the pretraining dataset **do**
    mask a fixed percentage of pieces in P
    separate P into 15x8x8 matrices
end for
feed the above matrices into MaskNet
train the model to predict the masked pieces
save the pre-trained model
for each puzzle P in the downstream dataset **do**
    separate P into 14x8x8 matrices
end for
transfer the top four layers of MaskNet
tune two additional layers on top extending it to the downstream task

Downstream Tasks. Our downstream tasks focus on the concept of forks. A fork in chess is when a piece simultaneously attacks two of the opponent's pieces. Processing forks requires an understanding of how pieces interact in chess, hence why we deemed it an appropriate test bed for our evaluation. More specifically, our tasks are *fork detection* — the binary classification task of determining if a position includes a fork — and *fork puzzle solving* — finding the move that would cause a fork. We framed the puzzle solving task as a multi-class classification task with 3670 classes, where 3670 represents the number of possible legal moves in chess.

Transferring to forks. The pretrained model that was trained to predict masked pieces is now the starting point for the new downstream task models. We freeze the first four layers of the model, and fine-tune a convolution and linear layer on top. We implement global average pooling before the linear layer. We subsequently use binary cross entropy and regular cross entropy loss for the *fork detection* and *fork puzzle solving* tasks respectively.

4 Results and Discussion

4.1 Experiments

4.1.1 Masking One Square

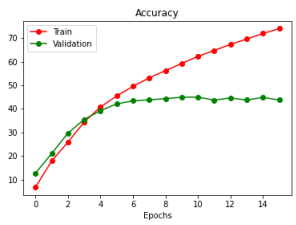

Figure 3: Accuracy of model trained end to end (*puzzle solving task*)

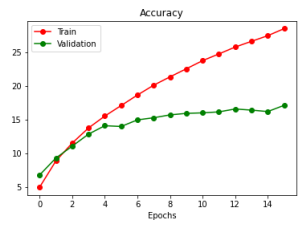

Figure 4: Accuracy of model with pretraining (masking 1 piece) (*puzzle solving task*)

Our earlier experiments involved using a pre-trained model that only masked one square. However, this model achieved a 64% accuracy on the fork detection task and a 17% accuracy on the puzzle solving task, which is quite poor compared to a model trained end to end, which achieved an accuracy of 76% and 40% on the fork detection and puzzle solver tasks respectively. This went against our initial hypothesis, suggesting that the pretraining procedure did not yield a meaningful representation. One thing to note from masking in other domains is that more masking and in general more difficult pretraining tasks were the ones that produced non-trivial representations. Namely, in their work *He et al.* showed that masking 75% of image pixels produced the most meaningful representation. So, we launched a further investigation into pre-trained models that masked a higher proportion of pieces on the chess board.

4.1.2 Masking Multiple Squares

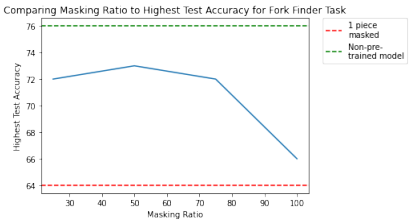

Figure 5: Accuracy of fork finding model as masking ratio increases

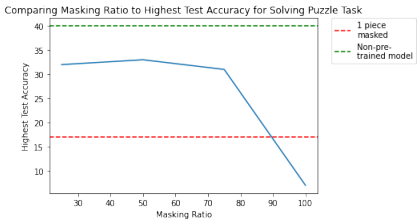

Figure 6: Accuracy of puzzle solving model as masking ratio increases

From the plots in Figure 5 and Figure 6, we can see that masking a higher proportion of chess pieces does improve the generalizability of the pre-trained model. Aside from the model that had a masking ratio of 100%, these models all performed better in both tasks than the model that only masked one chess piece. It is worth noting that there is overfitting as the masking ratio tends to 100% in which case the model is likely memorizing board configurations. Overall, these models still underperformed compared to the baseline end to end training, suggesting no meaningful representation was learned.

Note on Datasets All data was gathered from the puzzle dataset on lichess.com. We filtered out 150K fork positions for pretraining, 100K fork positions for puzzle solving task, and 50K fork positions + 50K non-fork positions for the fork finding task.

4.2 Limitations

We could not make use of the full dataset on lichess.com because of Google Colab GPU and RAM usage limits; in fact, we had to truncate the dataset to approximately a tenth of what it originally was in order to use the data in Google Colab. The smaller datasets could have very well caused overfitting issues, which could have impacted the generalization of the models - especially when a model is extended to different tasks. Future experiments should try and make use of a larger portion of the dataset to reduce the possibility of overfitting from too little data.

Additionally, pre-trained models generalize well to downstream tasks when those tasks are similar to the task that the model was pre-trained upon. In our experiments, it is quite possible that the task of predicting a masked chess piece is not as related to the tasks of detecting a fork or solving a fork puzzle as we had hypothesized. Thus, freezing the layers of the pretrained model when training it for the new task becomes detrimental to performance, since the parameters being frozen are far from optimal for the new task. Furthermore, it is possible that the pretrained model task is too easy. Even when 100% of the pieces were masked, the model was still able to predict the identity of the pieces with over 70% accuracy; not to mention, the models on average attained close to 70% validation accuracy after just one epoch of training. This suggests that a trivial representation was learned. Because the task is simple, the model does not learn enough to generalize to other tasks. One way to make the task harder is to train a network to detect which pieces it has difficulty predicting, and then train a model on just those hard to predict pieces.

5 Conclusion

In conclusion, our formulation of masked modeling in chess did not create meaningful representations that can improve downstream task performance. The pretrained models all underperformed in comparison to a baseline end to end training model. A possible shortcoming in our approach include choosing a pretrained model task that is too easy, so a trivial, not meaningful representation is learned. Future research should choose a harder task, such as first training a network to detect specific pieces the model has difficulties predicting, and then training a model specifically on those difficult to predict pieces.

6

References

A Kolesnikov et al. Big transfer (bit): general visual representation learning, vol. 6, no. 2. *arXiv preprint arXiv:1912.11370*, page 8, 2019.

Ting Chen, Simon Kornblith, Kevin Swersky, Mohammad Norouzi, and Geoffrey E Hinton. Big self-supervised models are strong semi-supervised learners. *Advances in neural information processing systems*, 33:22243–22255, 2020.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, pages 8748–8763. PMLR, 2021.

Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. Hierarchical text-conditional image generation with clip latents. *arXiv preprint arXiv:2204.06125*, 2022.

Mohit Shridhar, Lucas Manuelli, and Dieter Fox. Cliport: What and where pathways for robotic manipulation. In *Conference on Robot Learning*, pages 894–906. PMLR, 2022.

Pascal Vincent, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. Extracting and composing robust features with denoising autoencoders. In *Proceedings of the 25th international conference on Machine learning*, pages 1096–1103, 2008.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. *arXiv preprint arXiv:2111.06377*, 2021.

Weihua Hu, Bowen Liu, Joseph Gomes, Marinka Zitnik, Percy Liang, Vijay Pande, and Jure Leskovec. Strategies for pre-training graph neural networks. *arXiv preprint arXiv:1905.12265*, 2019.

7 Contributions of Group Members

All group members spent approximately equal time and effort on this paper.