ECAI 2024
U. Endriss et al. (Eds.)

doi:10.3233/FAIA240574

Empirical Evaluation of Concept Probing for Game-Playing Agents

Aðalsteinn Pálsson[a,*] **and Yngvi Björnsson**[a]

[a]Department of Computer Science, Reykjavik University
ORCID (Aðalsteinn Pálsson): https://orcid.org/0000-0003-4229-0387, ORCID (Yngvi Björnsson): https://orcid.org/0000-0001-5366-2639

Abstract. Concept probing is one prominent methodology for interpreting and analyzing (deep) neural network models. It has, for example, formed the backbone of several recent works to understand better the high-level knowledge learned and employed by game-playing agents, particularly in chess. However, some recent theoretical and empirical studies have questioned the methodology's reliability and highlighted some limitations. Here, in the game-playing domain of chess, we investigate the effectiveness of several different probing architectures and look into the reliability of methods for interpreting their results. We use a world-class chess-playing agent as our test domain, which allows us, via self-play, to quantify the importance of the concepts identified in the agent's neural network by the concept probes. Our results demonstrate that the widespread practice of using linear probes and interpreting their accuracy to indicate concept importance is somewhat unreliable and needs to be revised. We demonstrate several ways of doing that in our domain, particularly by using more complex probes and amnesic-like probing.

1 Introduction

Artificial intelligence systems employing machine-learning models are increasingly deployed in real-life settings, partly because of the recent successes of deep-learning neural-network-based approaches. Unfortunately, one drawback of that approach is the black-box nature of the neural networks and the need for more interpretability and explainability of the decisions they make. Explainable AI (XAI) seeks to rectify this, offering several methodologies to help improve the models' interpretability.

Concept probing is one prominent methodology for interpreting and analyzing (deep) neural network models [2]. Given a neural network trained on some task, concept probing aims to gain insights into the extent to which the network's internal layers have learned to represent various (high-level) concepts. This is done by training a separate classifier/regression model predicting a concept of interest using a layer's activations as input. If the resulting classifier performs well, the presumption is that the layer has learned to represent the given concept. This methodology has gained momentum and is now widely used to gain insights into neural network models in diverse domains such as natural language processing [5], image recognition [9], and game-playing [18]. However, some recent theoretical and empirical studies have cast doubt on how reliable this methodology is and highlighted some limitations [3, 6, 26].

* Corresponding Author. Email: adalsteinn19@ru.is.

Here, we investigate the effectiveness and reliability of concept probing in game playing using a world-class chess agent as our testbed, comparing different probing architectures and interpretation methods. The reasons for choosing this domain are threefold: first, there have been several recent high-profile works in XAI in game playing [18, 14, 21]; second, to address concerns raised in that work as to how reliable the probing results are; and three, because the domain allows us to quantify (via self-play) the importance of the identified concepts for the agent's playing strength (we contrast that information to the concepts' importance as judged by the probes).

The primary contributions of the work for our domain are: *i)* We empirically demonstrate that the widespread practice of using linear probes and interpreting their accuracy to indicate concept importance could be more reliable and should be revisited; *ii)* We propose ways of doing that, by using more complex probes and amnesic-like probing techniques, where applicable; *iii)* We offer additional insights into the pitfalls and best practices of concept probing, including concrete and relevant examples of potential failures.

The remainder of the paper is organized as follows. The next section introduces the terminology and background, followed by our methods and empirical result sections, respectively. Finally, we conclude and discuss future work.

2 Background

This section gives a brief background of model interpretability, generally and in games, and discusses some of the criticisms raised.

2.1 Interpretability

Interpretability of neural networks has received much-added attention in recent years due to the popularity of DNNs and the increasing use of ML in real-life situations (see e.g. [4, 19] for a survey). Interpretability of (black-box) models may be broadly categorized as either *global* or *local* and *model-specific* or *model-agnostic*. Global methods create interpretations valid across all input instances, whereas local methods' focus is on interpreting individual instances. Model-agnostic methods explain any black-box models, while model-specific methods leverage the model's architecture (thus, in reality, not treating the model as an absolute black-box).

Most prior work on local methods for interpreting models use feature-based explanations, which alter the input features (e.g., occlude or perturb them) [15, 27]. Such approaches are especially helpful in image-based domains and can reveal how different pixels or

regions in an image contribute to the network's output classification. Similar approaches have been used to explain chess positions [24].

However, the feature-based methods are not readily applicable to intricate concepts and are known to be unreliable [10] and susceptible to confirmation bias [9]. Thus, to overcome this shortcoming, more recent explanation techniques, often referred to as *concept-based*, use high-level human concepts as their interpretable units.

2.2 Concept Probing

A recent concept-based model-specific interpretability methodology for "peeking" into DNNs is *probing* [2]. Based on the intuition that deep neural networks are primarily about distilling computationally useful representations, one can monitor the output of different layers within the network for how well they represent various (high-level) concepts. By training classification or regression surrogate models (on a dataset not used for training the network itself) — called probes — to predict a given concept from a layer's activations, one can measure how much information that layer carries regarding the given concept. The higher the prediction accuracy of the probe, the more information that layer carries for representing the concept.

When concept probing we train a model, or a probe, g to map representations $f_l(x)$ at layer l to concepts z. As a proxy for how well f represents the concepts, one evaluates

$$performance\_metric(g(f_l(x)), z) \tag{1}$$

on unseen test data, and reports the results using a relevant performance metric, such as accuracy, as we do in this paper. This means that an accuracy value of 0.5 infers layer l does not contain information regarding concept z. In contrast, an accuracy value of one indicates that concept z can be fully decoded from layer l in model f. The choice of probing architecture, g, will depend on the experiment and whether the concept is binary or continuous. Researchers still debate the specific choices of probing architecture: some advocate for simpler linear probes, while others advocate for more complex ones, such as neural networks. The detailed nuances related to the specific choice of probing architecture will be discussed further in Section 2.3. For more details on concept probing see e.g. [3].

The model training and probing process requires two separate datasets: the training dataset $D_t = \{x^i, y^i\}$ used to train the model, f, and a separate probing dataset $D_p = \{f_l(x^i), z^i\}$, used to train the probe g. Where $f_l(x^i)$ is the internal representations in layer l for sample x^i and z^i is the concept's value for the corresponding sample. The probe is usually trained on a balanced dataset for binary concepts, meaning that the majority class is undersampled.

When designing the probing dataset D_p, a few things need to be considered, including that the more complex probe architectures (e.g., neural networks) typically call for larger training datasets than the simpler ones (e.g., linear). Thus, we opt for an extensive training dataset D_p of one million positions to ensure that all the probing architectures we compare have had sufficient training data.

2.3 Challenges Related to Concept Probing

What kind of probes are most useful for concept probing is still actively debated. While some researchers have advocated for simpler probes [2, 7, 13, 16, 17], others have advocated for more complex probes [5, 1, 23]. The debate is mainly focused on what kind of results we can safely interpret from the probes; for example, that too powerful of a probe could learn its own mappings between the representations and the concept (not present in the original model), and conversely, that too simple probes might not accurately reveal complex (non-linear) representation that the model has learned.

To gain further insight into the probed model, some additional experiments have been proposed to identify to what extent the model is learning a concept. For example, evaluating a lower- and upper bound for the value has been proposed to put the probing accuracy's numerical value into perspective. A lower bound can be found by probing an untrained network; this can help identify a few concerns, for example, to help attribute to which extent the probing accuracy is because of model learning [5] rather than other factors.

An upper bound can be seen as a way to show how well, theoretically, a mapping from the input to the concept could perform; this can be achieved by training a dedicated model $h(x)$ to predict the concept z given the input, x. In theory, this might seem like a viable option, but it is very resource-intensive because the models required for such an experiment are likely larger than needed for regular probing and thus more data-intensive. After all, they might require similar amount of data to the one used to train the original model f.

If concept probing is a reliable tool for identifying the presence of concepts, in theory, it should be able to guide the removal of information from the network. However, the work in [11] demonstrated that it fails to remove the concepts entirely and, in some cases, destroys other task-relevant features because the probing classifier is likely to use non-concept features to predict the presence of the concept.

Although concept probing only measures how well model representations can be mapped to a concept, it is tempting to infer that the model uses the concept. However, researchers have found evidence of false positives, meaning that high-accuracy probing does not necessarily mean that the concept benefits model's f task [6]. In that respect, a nice property of the game-playing domain we use is that we can quantify via self-play how useful a concept is for a model.

To further investigate how to interpret these experiments, researchers have proposed various interventions that provide further contrast to the results; one such experiment is to erase the concept from the model training data (as we also do in this paper) $D_o = \{x^i, y^i\}$ [26]. By erasing the concept from the training data, we can monitor performance degradation on the original task, which puts the importance of the concept into perspective, as well as changes in probing accuracy. Interestingly, Ravichander et al. [26] showed that by removing a concept from the training data, the probe may still be able to predict some properties of the concept.

Elazar et al. [6] coined the term *amnesic probing*, an extension to conventional probing, to describe an approach for studying behavioral changes of a model after removing concept information from its representations using Iterative Nullspace Projection (INLP) [25]. They show that even after removal, subsequent layers can recover some of the removed properties which, to some extent, speaks to the importance of non-linear information within the model because INLP only removes its linear components. In this paper we will refer to probing for concepts removed from the training set as *amnesic-like probing* due to its similarity with amnesic probing [25] – both methods probe for removed information.

Since this is still active research, a few things are still uncertain, e.g., to what extent can we infer that the model uses the concept while predicting, and second, perhaps more importantly, what are some actionable insights that may be gained through concept probing?

2.4 Interpretability in Games

Interpretability research in game-playing has received added attention in recent years. Game-playing is a domain with diverse chal-

lenges; for example, chess, which we use as a testbed in this research, has pieces with different roles and relatively simple rules yet sophisticated playing strategies, which makes for exciting challenges related to explainability.

The saliency method SARFA [24] is a perturbation approach to evaluating action-focused saliency that tackles specifically the problem that pieces affect value function differently. E.g., if removing a queen affects all future states similarly, including the proposed action, it is irrelevant to the proposed action.

The concept-probing work reported in McGrath et al. [17, 18] investigates the concepts that AlphaZero's neural network learned. They use Stockfish's hand-crafted concepts (among others) as a proxy for human-understandable knowledge. Using concept probing, they show that the network represents many human-understandable concepts and how and when they emerge during training. They demonstrate that material concepts get represented early in training, while more complex and subtle ones appear later. Furthermore, the paper briefly discusses alternative probing architectures; they advocate for simpler linear classifiers due to the danger of the concept probe learning its own complex relationships rather than capturing the structure in AlphaZero's representations.

The work in Pálsson and Björnsson [21] showcases how it is possible to identify what Stockfish's neural network has learned using interpretability methods, including concept probing. It exposed Stockfish's neural network emphasis on dynamic concepts and how it evaluates king safety differently than its hand-crafted counterpart.

Lovering et al. [14] analyzes high-level concepts learned by an AlphaZero-style Hex playing agent, demonstrating that the search discovers concepts during self-play before the neural network learns to encode them. Furthermore, it shows that short-term endgame planning is encoded in the final layers of the model, while long-term planning concepts are encoded in the middle layers of the model.

Tomlin et al. [32] extracted keywords from move-by-move Go commentary and compared them with simpler pattern-based concepts. Using a linear probe, they show that the keyword-based concepts (which are used as a proxy for concepts of higher-level abstraction) are better represented in the model's later layers. In contrast, the pattern-based concepts are better represented in the earlier layers.

Most concept-based methods now rely on predefined concepts; however, Schut et al. [28] explores ways to extract and successfully teach top chess grandmasters concepts beyond human knowledge.

3 Methods

In this section, we first describe Stockfish's neural-network evaluation function, followed by a description of the probing methods we use. Subsequently, we describe the Autoencoder used in the first experiment and the calculation of piece-specific importance, as well as discuss the chess-specific concepts implemented for the research.

3.1 Stockfish' Neural Network

Stockfish (since version 12) uses a neural network called NNUE [20] (EUNN Efficiently Updatable Neural Network) for evaluating game states. The network architecture was invented for the game Shogi but later ported to chess/Stockfish, immediately resulting in an 80 Elo point increase in playing strength (and more since then) [30, 31].

The NNUE architecture uses a (shallow) design with linear and clipped ReLU layers, as shown in Figure 1. Notable design choice is routing the inference through different LayerStacks, or subnetworks, depending on the game's phase (number of pieces on the board). Thus at each time, only one of eight LayerStack is used to calculate the evaluation. The LayerStacks follow only a single linear layer shared between all LayerStacks. This design choice is interesting from a concept-probing perspective because one would expect each LayerStack to re-implement much of the same concepts.

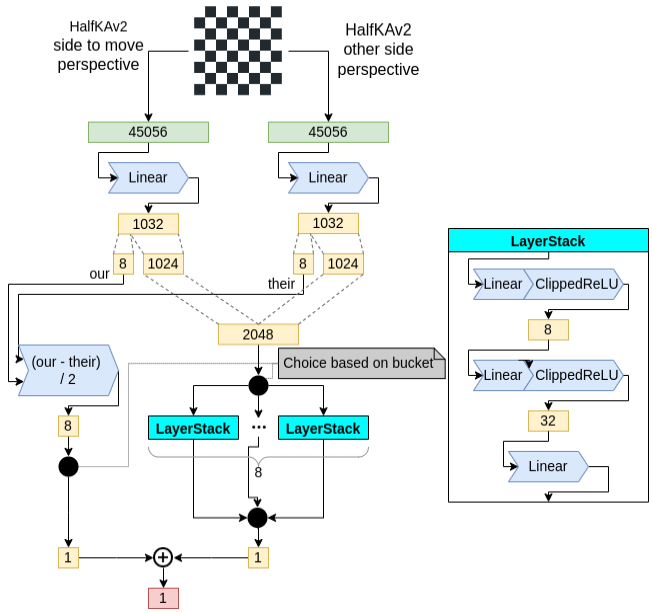

Figure 1. Stockfish' NNUE architecture [31]. It processes the values differently depending on the game phase. Depending on the total number of pieces left on the board, it will assign a bucket value and choose the corresponding path depending on the value. The bucket value is calculated by $bucket = (piece\_count - 1)/4)$.

Table 1. Autoencoder design, output sizes of each layer of the autoencoder.

Layer	Size
Input	768
Layer 1	1024
Layer 2	64
Layer 3	32
Layer 4	64
Layer 5	1024
Layer 6 (reconstruction)	768

The NNUE model is relatively shallow, and in our experiments, we probe the model solely after two layers (Layer 2). After a single linear layer, static information (invariant of the learning process) from the input is still relatively accessible. The probe only needs to decode the weights of a single linear layer; therefore, we want to look deeper. However, going even deeper into the neural network introduces new challenges; the computation splits into multiple pathways depending on the game phase, making the experimental evaluation more intricate. Thus, we chose layer two as the best practical compromise.

3.2 Probing Architectures

We experiment with three types of probing architectures in this paper; a linear probe using ridge [22] classification (Ridge), a neural network probe (NN), and a LightGBM [8] decision tree probe (LGBM) . The neural network probe uses feedforward architecture with ReLu nonlinearity and two hidden layers of sizes 100 and 10. For the linear probe, we perform a hyperparameter search over alpha values (the L2 term multiplier) of [0.01, 0.1, 0.5, 1, 5, 10, 50, 100, 500, 1000]. The decision tree probe is used with default parameters.

Table 2. The name and description of concepts used in the probing experiments.

Concept	Description
white_has_queen	White has a queen
white_queen_on_ initial_square	White has a queen on D1
Queen Advantage	Only one has a queen(s)
Rook Pair Advantage	Only one has a rook pair
Bishop Pair Advantage	Only one has a bishop pair
Knight Pair Advantage	Only one has a knight pair
Opposite Color Bishop	Both have only one bishop of opposite color
5 Dark Square Pawns	5 pawns are on dark squares

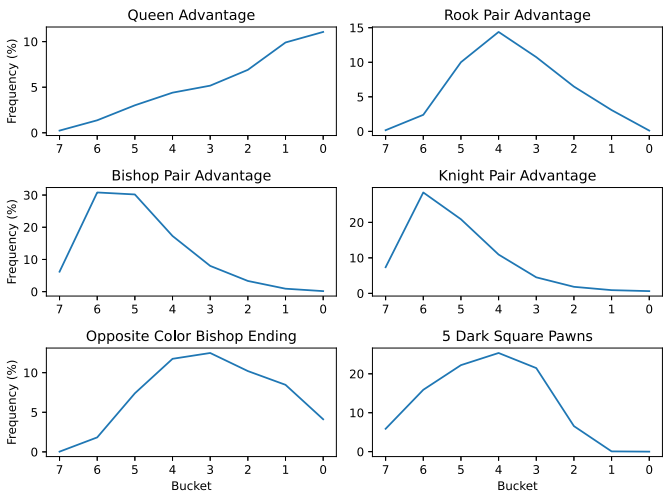

Figure 2. The frequency of the concepts in relation to the phase of the game, or bucket. Each value on the graph represent the proportion of each bucket that the concepts' value was true(and thus removed during training). The bucket indicates the number of pieces left on the board, where bucket 7 has the most and 0 has the fewest.

3.3 Autoencoder

We designed an autoencoder experiment to contrast the different probing architectures. Within this controlled environment, we can investigate whether the different probes are a good proxy for the information stored within the representations. When we encode and decode information, we can observe the reconstructed state to know how much information passes through the network - for us to decode information successfully, it must also be available in the encoded state. This method monitors the probing accuracy during training in the most compressed and reconstructed states. It allows us to observe inconsistencies in probing accuracy values and trends.

Inside the autoencoder, we expect considerable compression of information, which is relevant when probing a small neural network, such as Stockfish's NNUE, where the concept representations are likely more compressed than in a larger neural network.

The autoencoder consists of two parts, an encoder and a decoder, which are trained to compress and reconstruct the input space. In this case, the encoder and decoder mirror each other in terms of the number of layers and neurons in each layer. The design uses a sequence of fully connected linear layers with ReLU activations.

The input into the autoencoder is a tensor of size 768, or 64 (squares) $\times$ 6 (piece types) $\times$ 2 (color), and the output size of each layer is listed in the Table 1.

Table 3. Description of Δ piece value concepts.

Concept Name	Description
ΔP	Difference in number of pawns
ΔN	Difference in number of knights
ΔB	Difference in number of bishops
ΔR	Difference in number of rooks
ΔQ	Difference in number of queens

3.4 Piece Importance

By training and interpreting a linear surrogate model, we can evaluate how different piece-specific concepts impact the evaluation relative to each other. We perform this analysis to illustrate the model's behavior and alleviate concerns that we might be causing more harm to the model than intended.

For example, to identify the relative importance of different pieces, we can train the linear surrogate model:

$$w_p * \Delta P + w_n * \Delta N + w_b * \Delta B + w_r * \Delta R + w_q * \Delta Q = f(x) \quad (2)$$

where $f(x)$ is the model evaluation, the ΔP ... ΔQ concepts (seen in Table 3) indicate the difference in a number of pieces, e.g., ΔB indicates the difference in the number of bishops present on the board.

In this case, the linear weights for the different concepts will serve as a proxy for importance, and we can, for example, compare the weights w_p and w_q. Thus, we interpret $\frac{w_q}{w_p}$ as the queen value evaluated in a number of pawns as we do in Figure 4.

3.5 Custom Concepts

We implemented several chess-specific concepts for this research, all listed in Table 2.

First, we implement two straightforward concepts for the autoencoder experiment, *white_queen_on_initial_square* and *white_has_queen*. The concept *white_queen_on_initial_square* is represented by a single bit in the autoencoder's input, while *white_has_queen* can be represented by any of a total of 64 bits, assigned to represent the queen(s) position(s) on the board.

The latter concepts, also listed in Table 2, are used for training handicapped agents where we hide each one of these concepts, in turn, from the agents. This approach allows us to objectively evaluate the importance of each of these concepts by comparing the resulting agents' playing strengths. We chose concepts known from the chess literature either as essential or not. For example, having a pair of bishops or knowing how to dynamically evaluate the queen's strength compared to the other pieces (e.g., vs. two rooks or three minor pieces) is critical, whereas having a pair of knights or rooks is less so. The concept of having exactly five pawns on a dark square was explicitly contrived with the expectation of being irrelevant to evaluating a chess position. Although the choice of the above concepts was inspired by established human-based chess knowledge of what is important and what is less so, we furthermore ran experiments to quantify objectively how important these concepts were for the neural network model (see Section 4).

When defining the concepts for the restricted agents, we also had a few considerations in mind. First, the concept can not be too frequently active, as we do not want to reduce the size of the training set too much when removing a concept (in the worst case, an altered dataset got reduced by less than 14%). Removing positions from the training dataset with a given concept will inevitably affect the distribution of seen training positions over the different game phases;

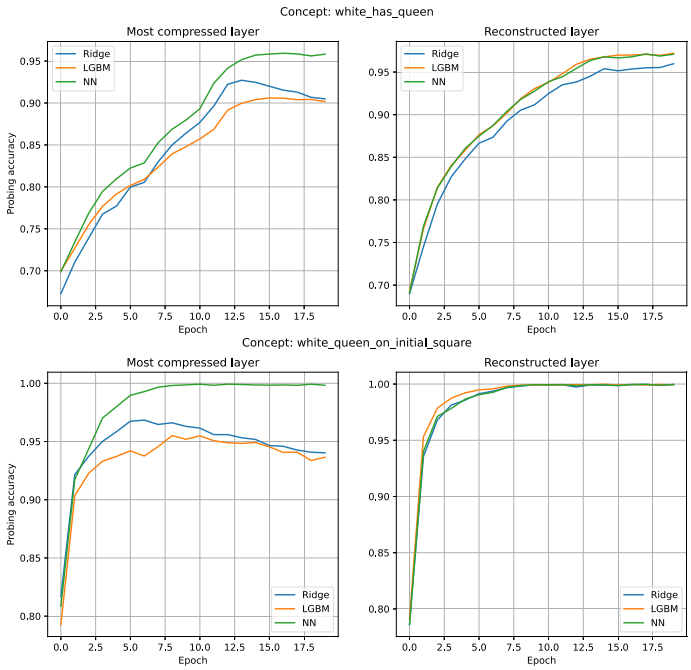

Figure 3. Probing accuracy results performed during training of the autoencoder. Results are for the concept white_has_queen (top) and white_queen_on_initial_square (bottom).

however, we verified that all game phases are still well represented, as seen in Figure 2. Second, it is important that when the concepts become active, they can also become deactive, so we do not remove all endgames associated with the concept and other concepts that may arise later in the game.

4 Results

Here, we present the results of our experiments. First, we describe the experimental setup, followed by the results of the autoencoder, concept importance, and probing accuracy with respect to concept importance experiments, respectively.

4.1 Experimental Setup

We use version 15.1 of Stockfish, which was the latest release at the start of the research.

For training, we use a dataset generated by *Leela Chess Zero* that is listed as a quality dataset (*training_data* at [29]). For concept probing, we sample one million chess positions from a Lichess dataset generated from standard rated chess games [12]. When training a modified agent, f_{res,c_i}, where concept c_i is hidden, we check each sample for concept c_i. If it is present, we discard the sample and continue sampling until the desired batch size is reached. Each agent is trained for 500 epochs using Stockfish's official training script (with slight modifications to remove chosen concepts during training) with recommended hyperparameters: batch size was set to 16384, initial learning rate as 8.75e-4, and γ (learning rate decay rate) as 0.992.

The autoencoder was trained using Adam with learning rate 5e-4. It achieved an 0.9973 accuracy and a 7.27e-3 Binary Cross Entropy loss, meaning that it was able to reconstruct its input almost perfectly.

All models were trained on a GeForce RTX 3080 graphics card. The error bars provided in the figures show one standard deviation above and below the mean from cross-validation over five splits.

4.2 Autoencoder Experiment

In this section, we present the results from the concept probing experiment where we monitor what happens during autoencoder training, shown in Figure 3. By observing the concept probing accuracy at the reconstructed layer, we can form an expectation of a lower bound for the amount of information present in the most compressed layer; it should be at least as much as in the reconstructed layer. Because if it is present at the reconstructed layer, it should also be present at the most compressed layer. Therefore, we base our interpretation of the results on observing how the concept probing accuracy in the reconstructed state evolves during training and comparing it to the concept probing accuracy in the most compressed layer.

At each training epoch, we probe for two simple concepts, *white_has_queen* and *white_queen_on_initial_square*. The concept's value is true if the statement is true and false otherwise. The probing is performed at the most compressed layer as well as the reconstructed layer. The encoded size is 32, while the reconstructed size is 768 – like the input.

The most important takeaway is that interpreting the probing accuracy at the most compressed state using the linear probe will lead to misleading results. For both concepts, the linear probe's (Ridge) probing accuracy decreases after saturating in the most compressed layer. At the same time, we see that at the reconstructed layer, the probing accuracy is either increasing or has reached saturation at 100% accuracy. Meanwhile, the neural network probe more accurately represents the presence and trend of information passing through the autoencoder as reflected by the reconstructed layer.

Albeit this is a somewhat contrived experimental scenario, this result nonetheless shows that linear probes may be insufficient to detect information present in the internal layers of a neural network, especially when one expects the information to be highly compressed.

4.3 Estimating Concept Importance

The attractive characteristic of the game-playing domain is that we can easily compare agents by making them compete, thus assessing their strength without relying on a potentially biased datasets. For this experiment, we choose six concepts well-suited for removal during training.

In order to identify the importance of a concept, we will: i) train an agent with and without the knowledge of a concept, ii) make them compete to assess the performance decline, and iii) expose how they evaluate positions differently. The tournament will be in a gauntlet style, where the restricted agents play against the regular agent (trained on the entire dataset). From the results, we then calculate the relative Elo rating of all agents, where the difference in Elo points serves as (a proxy for) the importance of the concept.

In Table 4, we see the results from the tournament. The *Queen Advantage* concept is the most important, resulting in a drop in performance of 31.6 Elo points, while the concept *5 Dark Square Pawns*

Table 4. Results in a gauntlet style tournament, where all agents only play against the Regular agent. Each modified agent plays 3000 matches at search depths 14, 15 and 16 each, thus a total of 9000 matches.

Description	Win-rate %	Elo drop
Queen	45.51(±0.54)	31.6(±3.8)
Bishop Pair	46.91(±0.51)	21.7(±3.6)
Rook Pair	48.33(±0.49)	11.7(±3.4)
Knight Pair	48.57(±0.49)	10.0(±3.8)
Opposite Color Bishop	49.04(±0.48)	6.7(±3.3)
5 Dark Square Pawns	49.74(±0.48)	1.8(±3.3)

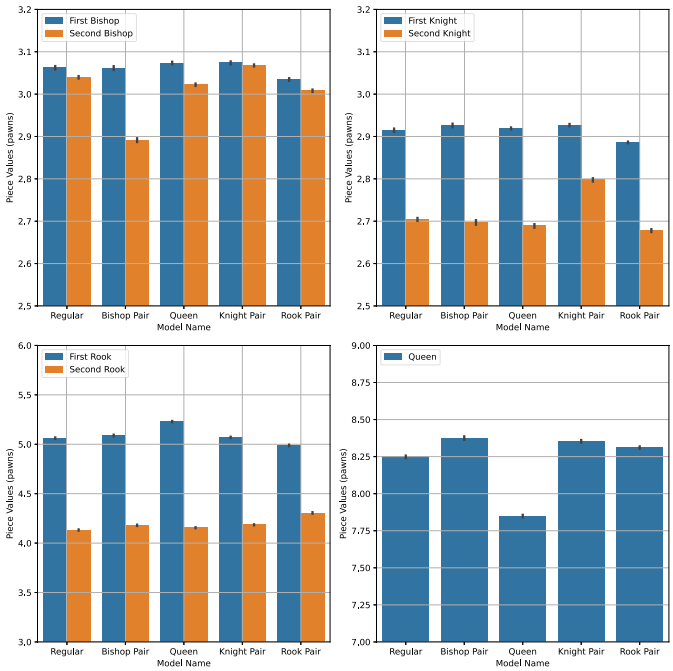

Figure 4. (upper left) When we remove a bishop pair advantage from the training data, the model associates lower importance to the second bishop. (upper right) Removing a knight pair advantage from the training data the model puts greater importance on the second knight. (lower left) Removing a rook pair advantage from the training data has little impact on the value it puts on having a second rook. (lower right) Never seeing examples where only one player has a queen, the model associates a lower value to the queen.

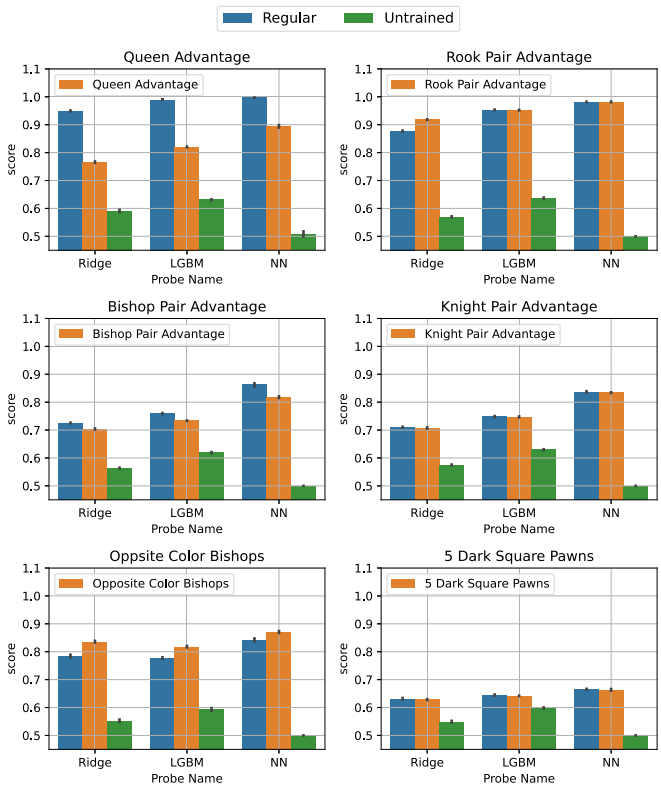

Figure 5. The probing accuracy of the regular model as well as models trained without seeing a concept.

shows a non-significant drop in performance. The difference between *Knight Pair Advantage* and *Rook Pair Advantage* is also non-significant, and it is approximately half of the importance of *Bishop Pair Advantage*. Overall, this result correlates well with the established human-knowledge chess literature.[1]

To demonstrate the behavioral changes that result from the interventions and alleviate some concerns that we might be causing unnecessary harm to the agents, we apply an explainable surrogate model to evaluate the piece-specific values. In Figure 4, we can see how not learning about these different piece-specific advantages affects the value it places on the pieces. For example, the agent that did not learn how valuable a bishop pair advantage is will underestimate the value of the second bishop (compared to the other agents). Conversely, the agent that did not learn the value of the knight pair advantage overestimates the value of the second knight. Furthermore, not learning about having a queen advantage results in the queen's value being approximately half a pawn less.

4.4 Concept Importance and Probing Accuracy

In this section, we will interpret the concept importance, how it correlates to probing accuracy, and what kind of recommendation we can give about interpreting the results from such experiments.

[1] For example, the chess literature strongly emphasizes the synergy of having the bishop pair (influencing both the light and the dark squares) in contrast to, e.g., having a pair of knights. The only (slight) surprise is that one might have expected the model to show more benefits for having the pair of rooks (e.g., as contrasted to a knight pair), as rooks typically work well together when connected, for example, on open files or in the endgame.

Figure 5 shows different probing results. We probe i) the Regular model, f_{reg}, trained on the whole dataset, ii) the restricted models, f_{res,c_i}, with concept c_i removed from the training set, and iii) an untrained model, f_{unt} (with weights uniformly initialized). For all models, we probe with the three probing architectures. Furthermore, in Figure 6, we see the relationship between the probing accuracy (in Layer 2) and the concept's importance, measured in Elo rating points.

We observe a moderate correlation between the probing accuracy and the importance of the concepts, as also emphasized in Table 5, which we will analyze later in this section. The most important concept, *Queen Advantage*, has the highest probing accuracy, and the unimportant one, *5 Dark Squared Pawns*, has the lowest, with the other concepts falling in between, which is reassuring. Nonetheless, we also see that concepts that receive near identical probing accuracy may differ significantly in importance, such as the concepts *Bishop Pair Advantage* and *Knight Pair Advantage*. The former concept is more important in our model as our self-play experiments clearly showed. Furthermore, this is in line with the established chess literature, which strongly prefers having the bishop pair over other minor piece combinations, including the knight pair. This result clearly shows that one has to take all probing results with some grain of salt and be extra cautious in interpreting their results.

In Figure 5, we compare the probing accuracy of the regular model versus a model trained on data with the concept in question removed from the training data. For the most important concepts, *Queen Advantage* and *Bishop Pair Advantage*, there is a notable drop in probing accuracy; in contrast, the accuracy remains similar for the less important concepts.

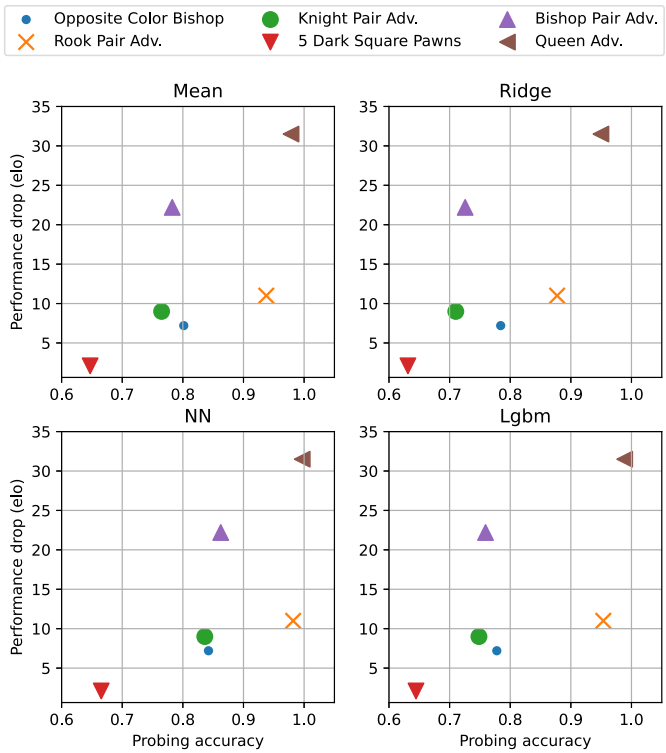

Figure 6. The relationship between probing accuracy and the importance measured in Elo points for all probes, as well as the mean value of all three probes.

Table 5. Pearson correlation between concept probing results and importance.

Probe, j	$P_j(f_{reg})$	$P_j(f_{reg}) - P_j(f_{res,c_i})$	$P_j(f_{reg}) - P_j(f_{unt})$
Ridge	0.69	0.83	0.64
LGBM	0.68	0.87	0.67
NN	**0.73**	**0.92**	**0.72**

To see what concept probing results mostly correlate with the importance, we analyze the Pearson correlation between the importance and i) $P_j(f_{reg})$, ii) $P_j(f_{reg}) - P_j(f_{res,c_i})$ and iii) $P_j(f_{reg}) - P_j(f_{unt})$. Where $P_j(f)$ is the probing accuracy of probe P_j applied to model f. The results can be seen in Table 5, from which we can draw two conclusions. First, when analyzing the probing results of the Regular model, $P_j(f_{reg})$, the neural network probe has the highest correlation with importance. Second, the change in probing accuracy after removing a concept from the training set, $P_j(f_{reg}) - P_j(f_{res,c_i})$, offers an even higher correlation with importance for all probes, and again with the neural network probe performing best.

To check to what extent we can predict the importance using the probing results, and provide a bit more context to previous results, we train a linear model to predict the importance and report the mean absolute error using leave-one-out cross-validation. The results are shown in Table 6 where we see again that the neural network probe is performing the best, and using the difference in probing accuracy between the Regular and Restricted models results in the highest accuracy, with a mean absolute error of 3.68 Elo points.

Table 6. The mean absolute error in Elo points, using a leave-one-out cross validation, of a linear model predicting the importance, given the concept probing results.

Probe, j	$P_j(f_{reg})$	$P_j(f_{reg}) - P_j(f_{res,c_i})$	$P_j(f_{reg}) - P_j(f_{unt})$
Ridge	9.61	5.68	10.48
LGBM	9.77	5.83	10.15
NN	**8.84**	**3.68**	**8.90**

5 Conclusion

In this work, we cast some light on how effective and reliable concept probing is, using a contrived autoencoder and a state-of-the-art game-playing agent as our test domain. The main takeaways are:

- The widespread practice of using linear probes and interpreting their accuracy to indicate concept importance gives some, albeit limited, insights into concepts learned by the network. The probes' accuracy and feature importance is only moderately correlated. Additionally, interpreting concept importance based on probe accuracy alone is unreliable. For example, we saw a concrete example of concepts indistinguishable that way — possessing the Bishop or Knight pair, the former highly beneficial and the latter not (as judged by both our self-play experiments and established chess-domain knowledge).
- However, a far better correlation is achievable using a more complex probe (neural network) and reading into the results using amnesic-like probing techniques. Doing this, where applicable, is recommended; however, we acknowledge that this may not always be possible (e.g., when unable to retrain the model).
- Using an untrained model as a baseline for the probing (e.g., as suggested in [5, 33]) did not prove helpful in our domain.

We see this work as valuable input into the ongoing discussion of the pros and cons of different probing architectures, especially in the game-playing domain. Furthermore, it provides some guidance for best practices in interpreting probing results.

As for future work, we plan to address some of the limitations of this study. First, we use only a handful of chess concepts, and adding more would improve confidence in the correlation metric. Second, we use the effects on playing strength by omitting a concept from model training as an importance metric. Albeit useful, as demonstrated, this is not necessarily the entire truth; we might be causing unintended harm to the models by removing other valuable correlated concepts. Although we were mindful of this when defining the concepts and when examining the models for unexpected behavior, such as the experiment in Figure 4, a more in-depth investigation of such correlated concepts is warranted.

Finally, retraining a model from scratch while holding out some concepts is not always feasible cost-wise or even impossible; thus, developing more practical ways to restrict a model without having to retrain from scratch might be worthwhile.

References

[1] Y. Adi, E. Kermany, Y. Belinkov, O. Lavi, and Y. Goldberg. Fine-grained analysis of sentence embeddings using auxiliary prediction tasks. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017. URL https://openreview.net/forum?id=BJh6Ztuxl.

[2] G. Alain and Y. Bengio. Understanding intermediate layers using linear classifier probes. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Workshop*

Track Proceedings. OpenReview.net, 2017. URL https://openreview.net/forum?id=HJ4-rAVtl.

[3] Y. Belinkov. Probing classifiers: Promises, shortcomings, and advances. *Computational Linguistics*, 48(1):207–219, 2022.

[4] F. Bodria, F. Giannotti, R. Guidotti, F. Naretto, D. Pedreschi, and S. Rinzivillo. Benchmarking and survey of explanation methods for black box models. *CoRR*, abs/2102.13076, 2021. URL https://arxiv.org/abs/2102.13076.

[5] A. Conneau, G. Kruszewski, G. Lample, L. Barrault, and M. Baroni. What you can cram into a single \$&!#* vector: Probing sentence embeddings for linguistic properties. In I. Gurevych and Y. Miyao, editors, *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 1: Long Papers*, pages 2126–2136. Association for Computational Linguistics, 2018. doi: 10.18653/v1/P18-1198. URL https://aclanthology.org/P18-1198/.

[6] Y. Elazar, S. Ravfogel, A. Jacovi, and Y. Goldberg. Amnesic probing: Behavioral explanation with amnesic counterfactuals. *Trans. Assoc. Comput. Linguistics*, 9:160–175, 2021. doi: 10.1162/tacl_a_00359. URL https://doi.org/10.1162/tacl_a_00359.

[7] D. Hupkes, S. Veldhoen, and W. H. Zuidema. Visualisation and 'diagnostic classifiers' reveal how recurrent and recursive neural networks process hierarchical structure. *J. Artif. Intell. Res.*, 61:907–926, 2018. doi: 10.1613/jair.1.11196. URL https://doi.org/10.1613/jair.1.11196.

[8] G. Ke, Q. Meng, T. Finley, T. Wang, W. Chen, W. Ma, Q. Ye, and T. Liu. Lightgbm: A highly efficient gradient boosting decision tree. In I. Guyon, U. von Luxburg, S. Bengio, H. M. Wallach, R. Fergus, S. V. N. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 3146–3154, 2017. URL https://proceedings.neurips.cc/paper/2017/hash/6449f44a102fde848669bdd9eb6b76fa-Abstract.html.

[9] B. Kim, M. Wattenberg, J. Gilmer, C. Cai, J. Wexler, F. Viegas, and R. Sayres. Interpretability beyond feature attribution: Quantitative Testing with Concept Activation Vectors (TCAV). In *35th International Conference on Machine Learning, ICML 2018*, volume 6, pages 4186–4195, 2018. ISBN 9781510867963.

[10] P. Kindermans, S. Hooker, J. Adebayo, M. Alber, K. T. Schütt, S. Dähne, D. Erhan, and B. Kim. The (un)reliability of saliency methods. In W. Samek, G. Montavon, A. Vedaldi, L. K. Hansen, and K. Müller, editors, *Explainable AI: Interpreting, Explaining and Visualizing Deep Learning*, volume 11700 of *Lecture Notes in Computer Science*, pages 267–280. Springer, 2019. doi: 10.1007/978-3-030-28954-6_14. URL https://doi.org/10.1007/978-3-030-28954-6_14.

[11] A. Kumar, C. Tan, and A. Sharma. Probing classifiers are unreliable for concept removal and detection. *Advances in Neural Information Processing Systems*, 35:17994–18008, 2022.

[12] Lichess. Lichess data set. https://database.lichess.org/, 2023. Accessed: 2023-08-13.

[13] N. F. Liu, M. Gardner, Y. Belinkov, M. E. Peters, and N. A. Smith. Linguistic knowledge and transferability of contextual representations. In J. Burstein, C. Doran, and T. Solorio, editors, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 1073–1094. Association for Computational Linguistics, 2019. doi: 10.18653/v1/n19-1112. URL https://doi.org/10.18653/v1/n19-1112.

[14] C. Lovering, J. Forde, G. Konidaris, E. Pavlick, and M. Littman. Evaluation beyond task performance: Analyzing concepts in alphazero in hex. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, editors, *Advances in Neural Information Processing Systems*, volume 35, pages 25992–26006. Curran Associates, Inc., 2022. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/a705747417d32ebf1916169e1a442274-Paper-Conference.pdf.

[15] S. M. Lundberg and S. I. Lee. A unified approach to interpreting model predictions. In *Advances in Neural Information Processing Systems*, volume 2017-Decem, pages 4766–4775, 2017. URL https://github.com/slundberg/shap.

[16] R. H. Maudslay, J. Valvoda, T. Pimentel, A. Williams, and R. Cotterell. A tale of a probe and a parser. In D. Jurafsky, J. Chai, N. Schluter, and J. R. Tetreault, editors, *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 7389–7395. Association for Computational Linguistics, 2020. doi: 10.18653/v1/2020.acl-main.659. URL https://doi.org/10.18653/v1/2020.acl-main.659.

[17] T. McGrath, A. Kapishnikov, N. Tomasev, A. Pearce, D. Hassabis, B. Kim, U. Paquet, and V. Kramnik. Acquisition of chess knowledge in alphazero. *CoRR*, abs/2111.09259, 2021. URL https://arxiv.org/abs/2111.09259.

[18] T. McGrath, A. Kapishnikov, N. Tomašev, A. Pearce, M. Wattenberg, D. Hassabis, B. Kim, U. Paquet, and V. Kramnik. Acquisition of chess knowledge in alphazero. *Proceedings of the National Academy of Sciences*, 119(47):e2206625119, 2022.

[19] J. X. Mi, A. D. Li, and L. F. Zhou. Review study of interpretation methods for future interpretable machine learning. *IEEE Access*, 8: 191969–191985, 2020. ISSN 21693536. doi: 10.1109/ACCESS.2020.3032756.

[20] Y. Nasu. Efficiently updatable neural-network-based evaluation functions for computer shogi. *The 28th World Computer Shogi Championship Appeal Document*, 2018.

[21] A. Pálsson and Y. Björnsson. Unveiling concepts learned by a world-class chess-playing agent. In *Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence, IJCAI 2023*. ijcai.org, 2023.

[22] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[23] T. Pimentel, J. Valvoda, R. H. Maudslay, R. Zmigrod, A. Williams, and R. Cotterell. Information-theoretic probing for linguistic structure. In D. Jurafsky, J. Chai, N. Schluter, and J. R. Tetreault, editors, *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 4609–4622. Association for Computational Linguistics, 2020. doi: 10.18653/v1/2020.acl-main.420. URL https://doi.org/10.18653/v1/2020.acl-main.420.

[24] N. Puri, S. Verma, P. Gupta, D. Kayastha, S. Deshmukh, B. Krishnamurthy, and S. Singh. Explain your move: Understanding agent actions using specific and relevant feature attribution. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020. URL https://openreview.net/forum?id=SJgzLkBKPB.

[25] S. Ravfogel, Y. Elazar, H. Gonen, M. Twiton, and Y. Goldberg. Null it out: Guarding protected attributes by iterative nullspace projection. In D. Jurafsky, J. Chai, N. Schluter, and J. R. Tetreault, editors, *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 7237–7256. Association for Computational Linguistics, 2020. doi: 10.18653/V1/2020.ACL-MAIN.647. URL https://doi.org/10.18653/v1/2020.acl-main.647.

[26] A. Ravichander, Y. Belinkov, and E. H. Hovy. Probing the probing paradigm: Does probing accuracy entail task relevance? In P. Merlo, J. Tiedemann, and R. Tsarfaty, editors, *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume, EACL 2021, Online, April 19 - 23, 2021*, pages 3363–3377. Association for Computational Linguistics, 2021. doi: 10.18653/v1/2021.eacl-main.295. URL https://doi.org/10.18653/v1/2021.eacl-main.295.

[27] M. T. Ribeiro, S. Singh, and C. Guestrin. "Why should i trust you?" Explaining the predictions of any classifier. *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 13-17-Augu:1135–1144, 2016. doi: 10.1145/2939672.2939778.

[28] L. Schut, N. Tomasev, T. McGrath, D. Hassabis, U. Paquet, and B. Kim. Bridging the human-ai knowledge gap: Concept discovery and transfer in alphazero. *CoRR*, abs/2310.16410, 2023. doi: 10.48550/ARXIV.2310.16410. URL https://doi.org/10.48550/arXiv.2310.16410.

[29] Stockfish. Training data sets. https://github.com/glinscott/nnue-pytorch/wiki/Training-datasets, 2022. Accessed: 2022-01-30.

[30] Stockfish. introducing-nnue-evaluationo. https://blog.stockfishchess.org/post/625828091343896577/introducing-nnue-evaluation, 2022. Accessed: 2022-04-05.

[31] Stockfish. nnue readme. https://github.com/glinscott/nnue-pytorch/blob/master/docs/nnue.md, 2022. Accessed: 2022-04-05.

[32] N. Tomlin, A. He, and D. Klein. Understanding game-playing agents with natural language annotations. In S. Muresan, P. Nakov, and A. Villavicencio, editors, *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers), ACL 2022, Dublin, Ireland, May 22-27, 2022*, pages 797–807. Association for Computational Linguistics, 2022. doi: 10.18653/V1/2022.ACL-SHORT.90. URL https://doi.org/10.18653/v1/2022.acl-short.90.

[33] K. W. Zhang and S. R. Bowman. Language modeling teaches you more syntax than translation does: Lessons learned through auxiliary task analysis. *CoRR*, abs/1809.10040, 2018. URL http://arxiv.org/abs/1809.10040.