SENNs and Adversarial Robustness

Valerio Pepe, Giovanni D'Antonio, Martin Dimitrov
School of Engineering and Applied Sciences
Harvard University
Cambridge, MA 02138
{valeriopepe, giovannidantonio, martin_dimitrov}@college.harvard.edu

December 10, 2024

Abstract

Self-Explaining Neural Networks (SENNs, [1]) promise interpretable, intrinsically explainable modeling by decomposing predictive processes into understandable concepts and relevance weights. In this paper, we present a systematic investigation into the adversarial robustness of SENNs under the Fast Gradient Sign Method (FGSM, [2]). We find that while SENNs can withstand minor, unstructured noise with minimal performance degradation, more targeted "chunked" perturbations severely compromise the model's accuracy. Our analyses reveal distinct vulnerability profiles: the aggregator exhibits non-monotonic accuracy patterns explained by its manipulation of dot products. In contrast, the conceptizer and parameterizer display more predictable yet ultimately devastating accuracy declines due to the aforementioned vulnerabilities to concept-based noise, which we term the **viable alternative hypothesis**. We conclude by outlining opportunities for future work, including exploring stronger attacks and more robust defenses, and checking if these results hold for non-gradient-based adversarial attacks. Code for the paper can be found in the 'report.ipynb' file at https://anonymous.4open.science/r/2822-SENN-Project-9D0E[1]

1 Introduction

Over the last decade, modern machine learning techniques have vastly improved on previous state-of-the-art computational performance across a variety of different tasks and modalities (from image recognition to text generation). The main edge these techniques have over previous ones – their exploitation of complex nonlinear relationships between representations – allows for superior modeling capabilities and greater flexibility in learning, both of which are important desiderata for any successful machine learning model. However, this additional performance is not without its drawbacks: deep nonlinear relationships between neurons also make the interpretation of the model's inner workings much harder, spurring the birth of machine learning interpretability as an academic discipline.

Most techniques for ML interpretability are post-hoc, focusing on inferring the best possible explanation for a certain output given some feature of the model (its gradient, its performance on similar perturbed inputs, etc.). These types of explanations are easy to produce and do not compromise on the model's expressivity, but have one fatal flaw; if they are meant for human consumption, explanations can be unfaithful to the actual workings of the model but still convince human operators, potentially leading to the wrong outcome being applied to an input (e.g. the wrong diagnosis in the medical field, the wrong sentence in legal applications, a loan denial in financial ones).

On the other hand, models that use explanations as part of their decision-making process do not have such risks: a manipulated explanation would lead to the wrong outcome for the model, so unfaithful explanations are harder to change to the benefit of a third party. However, issues can still arise if these models are vulnerable to adversarial attacks on the input: if an input can be changed in a manner imperceptible to humans and lead to a different explanation

[1]This is a fork of the base repository we took the SENN implementation from, hence why the README has other people's names on it. We also use anonymous.4open.science to avoid having to un-private the GitHub Repo, since that is usually considered a violation of the Honor Code.

(and therefore outcome) anyway, such interpretability methods are still *de facto* vulnerable to the manipulation of explanations, and have no particular advantage over post-hoc explanation methods.

For this reason, in this paper we investigate the adversarial robustness of one such co-generative architecture, the Self-Explaining Neural Network (SENN) [1]. Additionally, we focus on the Fast Gradient Sign Method as our adversarial attack given its simplicity and popularity in the adversarial machine learning literature.

2 Related Work

Self-Explaining Neural Networks. Self-Explaining Neural Networks (SENNs) are a class of models that integrate interpretability directly into their architecture, making explanations intrinsic to the model rather than relying on post-hoc methods [1]. SENNs achieve this by decomposing the prediction process into three components: a concept encoder, a relevance parametrizer, and an aggregation function. The concept encoder maps raw inputs into a small set of interpretable basis concepts, which represent higher-level, semantically meaningful features derived from the data. The relevance parametrizer then computes scores for each of these concepts, representing their individual contributions to the final prediction. Lastly, the aggregation function combines the concepts and their corresponding relevance scores to produce the model's output, ensuring the process remains transparent and interpretable.

The design of SENNs is grounded in three interpretability desiderata: explicitness, faithfulness, and stability [1]. Explicitness ensures that explanations are immediately understandable by humans, with each concept and its relevance being clearly linked to the prediction. Faithfulness guarantees that the relevance scores accurately reflect the true influence of the corresponding concepts on the output. Stability, a critical property for robustness, requires that similar inputs yield consistent explanations, ensuring reliability even in the presence of small input perturbations.

This paper explores the adversarial robustness of SENNs, focusing on how their modular architecture responds to attacks. Unlike traditional neural networks, SENNs provide a broader attack surface due to their separate components, each of which can be individually targeted. For example, an adversarial perturbation to the concept encoder might distort the semantic representation of the input, while an attack on the relevance parametrizer could manipulate the assigned importance scores, leading to misleading explanations. By studying the impact of these attacks across different components, we aim to shed light on the vulnerabilities and strengths of interpretable models like SENNs. Our analysis is particularly relevant given the increasing emphasis on deploying machine learning models in high-stakes domains where both robustness and interpretability are paramount.

Fast Gradient Sign Method. The 'Fast Gradient Sign Method' (FGSM) adversarial attack, introduced in [2], uses the gradient of the loss with respect to the input to modify the input to move it away from its decision boundary.

For some loss function J, input x, model parameters θ, and label y, an FGSM-perturbed input $\tilde{x}$ with attack strength $0 \leq \varepsilon \leq 1$ is:

$$\tilde{x} = x + \varepsilon \cdot \text{sign}(\nabla_x J(\theta, x, y))$$

where '+' denotes an element-wise addition of the perturbation to the original input. The bounds of ε are meant to capture the minimum and maximum pixel values, so 0 and 1 are not specific, and scale with the images used. In this work, our data was scaled between 0 and 3, so we just scaled ε by 3 to match.

Notice that by construction, any attack with strength $\varepsilon \geq 0.5$ can make the input uniformly one value (since we can always construct some gradient where all values 0 are perturbed by 0.5, all values 1 are perturbed by -0.5, and the pixels in between are scaled appropriately, outputting an adversarial input that is uniformly 0.5).

This is a simple attack, but it forces the input to move exactly orthogonal to the decision boundary, meaning it is provably the attack that moves the input the furthest away from it. It is also a very popular attack due to this simplicity, and forms the basis for more sophisticated attacks (e.g. Projected Gradient Descent, [4]), thus informing our choice for this work. Finally, in previous work SENNs have been perturbed by special, purpose-built attacks to exploit their concept-based nature [5] [6]. In contrast, a standard attack like FGSM has never been used, making our work novel and more intellectually interesting.

MNIST. For our evaluation, we employ MNIST [3], a collection of 28×28-pixel, grayscale images representing handwritten digits taken from a mixture of data from the American Census Bureau and American high school students. This is a very common dataset in computer vision due to its simplicity and the interesting semantic distinctiveness of each of its classes. We also use it to ensure direct comparison to other works in the SENN literature – due to the simplicity of the concepts the model finds, both [1] and most further work on the architecture uses it.

3 Methods

As an adversarial attack, FGSM is only well-motivated in its use on the last layer of a neural network: it is designed to move the input as far from a decision boundary as possible, and decision boundaries are ill-defined before the last layer of a network (it is not guaranteed that inputs going to different classes cluster differently in prior layers, for instance). Thus, in traditional neural network architectures, there is only one possible mode of attack for FGSM. This is not true for SENNs: since each component makes decisions independently of the others, each one must have its own well-defined decision boundaries in order to generate meaningful concepts, relevance scores, etc.

How to apply FGSM to SENNs is therefore not immediately obvious: given we are interested in the general robustness of SENNs, we try as many angles of attack as possible. These can be split into two main categories, 'single-component perturbations' and 'multiple-component perturbations'.

3.1 Single-Component Perturbations

Single-component perturbations attack a single one of the three components of the SENN through FGSM. This is a superset of the canonical FGSM attack: usually, we would only be attacking the layer that determines the input class. Here, that is the aggregator, but we can also attack the conceptizer and parameterizer, giving us a broader surface of attack.

Here, we treat each component completely separately. This means that our method is as follows: for the component we are interested in attacking, we take its input and output, and simply perform an FGSM attack with regards to *this component's* loss, not the full SENN's loss. This means every component has a different loss metric we are optimizing over: the conceptizer is an autoencoder, so it has a mean squared error (MSE) loss between the input and reconstruction; the parameterizer is another autoencoder, but trained an MSE loss with an L_1 penalty to incentivize sparsity in the relevances, presumably to make training easier; and the aggregator is trained with standard cross-entropy loss given it is a classifier.

The main consequence of the single-component approach is that not all attacks propagate completely back to the input. The conceptizer and the parameterizer both take in the image as (part of) their input and map it to some vector in latent space. This means that their loss with regards to their input will yield a new, perturbed image. In contrast, the aggregator only takes in these latent space vectors, so FGSM on it will return perturbed latent space vectors, not a perturbed image. Arguably, this is out-of-scope for an adversarial attack since it is not a representation a human will usually be able to manipulate directly. However, given the modularity of SENNs, this is not impossible – hosting each of a SENN's components on a different server and selectively injecting perturbed data into one is a plausible attack vector, so we will be investigating it anyway.

3.2 Multiple-Component Perturbations

Multiple-component perturbations attack two of the components: either the conceptizer+parameterizer, the conceptizer+aggregator, or parameterizer+aggregator. [2]

The perturbations with the aggregator work as described in Algorithm 1 (which describes the conceptizer+aggregator attack specifically, but the parameterizer+aggregator attack is exactly the same if one swaps out the conceptizer for the parameterizer in every line after the first). We first run the input through the entire SENN. Then, we calculate the loss on the aggregator and perturb the concepts/relevances using it ($\nabla_{concepts}$ in Algorithm 1). Since the concepts/relevances are part of the latent space and not the input space and given they are autoencoders, we then decode these latent space representations back into their 'reconstructed' images [3] ($x_{perturbed,reconstructed}$ in Algorithm 1). Finally, we use this image to calculate the loss of the relevant autoencoder (conceptizer or parameterizer). Now, since both of the autoencoders act directly on the input, we run FGSM on *this* gradient to obtain a perturbed input ($x_{perturbed}$ in Algorithm 1). $x_{perturbed}$ is then the perturbed input that we re-run through the SENN to check the network's accuracy.

The conceptizer+parameterizer joint attack is structured differently. Contrary to the relationship between the autoencoders and the aggregator, the conceptizer and parameterizer are not as straightforwardly serial, so we cannot use the same strategy of 'forcing' one input to be the other's output and propagate the gradient in this way. Instead, we opt to

[2] Note that attacks that attack all three components are not well-defined, since a SENN is sequential, so it is unclear how the attack to the aggreator should propagate back into the input if both other components are also being attacked and therefore impart their own attacks onto the input. We leave this open for future work.

[3] Essentially, we pretend that these representations are the product of some real input, and try to get the relevant autoencoder to reconstruct that input from these representations.

Algorithm 1 The conceptizer+aggregator joint attack

Require: Image x, Label y, Conceptizer conc(), Parameterizer para(), Aggregator agg()

$concepts, relevances \leftarrow \text{conc}(x), \text{para}(x)$

$prediction \leftarrow \text{agg}(concepts, relevances)$

$agg\_loss \leftarrow CrossEntropyLoss(prediction, label)$

$\nabla concepts = agg\_loss.backwards()$

$concepts_{perturbed} = FGSM(concepts, \varepsilon, \Delta_{concepts})$ ▷ Perturbs concepts using the aggregator's gradient

$x_{perturbed,reconstructed} = \text{conc}.decode(concepts_{perturbed})$

$conc\_loss = MSELoss(x_{perturbed,reconstructed}, x)$

$\nabla_x = conc\_loss.backwards()$

$x_{perturbed} = FGSM(x, \varepsilon, \nabla_x)$ ▷ Perturbs input using the conceptizer's gradient

average the two: take the gradient of the loss with respect to one component and then the other, separately, and average the two into a single meta-gradient we then sum to the original image to produce the FGSM-perturbed input.

4 Results

4.1 Experimental Details

To produce the result graphs, each perturbation method (the single perturbations and the joint perturbations) was tested on the same batch of 1000 images from MNIST's validation set, so we could ensure that none of them had been seen by the SENN during training. We perturbed each image at intervals of $\varepsilon = 0.1$ (thus testing $0, 25/255, 50/255, \ldots, 255/255$, images of the effect of these perturbations on the images can be seen below in Fig. 1).

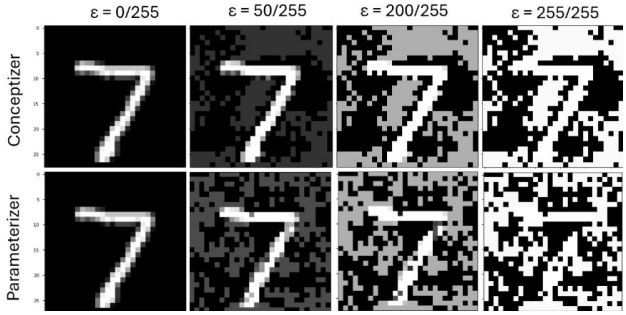

Figure 1: The same image of a 7, perturbed across different strengths of FGSM and across both autoencoders (aggregator perturbations perturb the latent space, not the input space, and are not included here, see Fig. 3).

We also seeded PyTorch's randomness such that the results would be deterministic across runs, and the base SENN class implementation itself was provided by this [4] repo, a 2020, Python 3.7 PyTorch re-implementation of the original 2018, TensorFlow codebase in [1].

4.2 Experimental Results

Figure 2(A) shows the results of the single-component perturbations. The trends in accuracy are quite cleanly split by architecture, with the aggregator's curve being completely different from the autoencoders'. Note the unusual shape of the aggregator's curve: the attack is meant to make the model worse as its strength increases, which is what happens with the conceptizer and parameterizer, so it is *a priori* unclear why the aggregator would behave so differently.

[4] https://github.com/AmanDaVinci/SENN

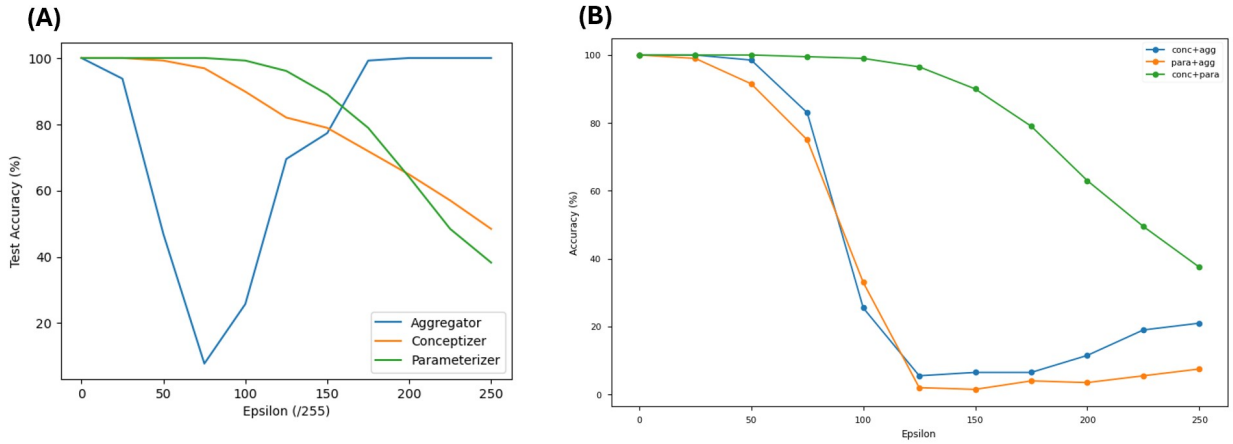

Figure 2: The SENN's accuracy on MNIST as a function of the strength of both the single-component (panel (A)) and multi-component (panel (B)) FGSM. Every line represents the perturbation of a different component or combination of components.

Because of this, it is also not trivial to say which component is the most robust to FGSM, since each of the three components is the worst for some value of ε (the aggregator for $\varepsilon \leq 150$, the conceptizer for $150 \leq \varepsilon \leq 200$, and the parameterizer for the remainder of the graph), so a decision would depend entirely on the definition of adversarial robustness used. From a qualitative perspective, though, we say the aggregator is the least robust, since it causes the lowest test accuracy of any of the three and it does so at a far lower ε than either of the two attacks.

This uncertainty in which method is the most robust warrants additional study, however, which we can better tease apart using the multiple-component perturbations as seen in Figure 2 (B).

Figure 2(B) is interesting because the joint perturbation of the components confirms the intuitions given in Figure 2(A). As can be clearly seen, the perturbations including the aggregator are by far the least robust ones, dipping to below chance (10%) accuracy by $\varepsilon = 125$, something the conceptizer+parameterizer condition never does. The two autoencoders are also again similar in their trends, ending up with very similar accuracies at the highest values of ε, and take more similar paths to such accuracies here than they did in the previous figure (likely due to the aggregator modulating these curves more than either of the autoencoders).

5 Discussion

The results shown in the previous section suggest that SENNs can be quite vulnerable to FGSM, mostly when jointly attacked. The figure below shows inputs that have been jointly attacked and misclassified by the SENN: these are clearly still intelligible for humans, so the network's failure to classify them correctly stems mainly from a lack of robustness to this type of semantically irrelevant noise.

Therefore, given these weaknesses and the unintuitive trends for accuracy shown in Fig. 2, it is important to explore why the SENN reacts in the way it does. To this end, we designed a set of two additional experiments (one for the aggregator, and one for the autoencoders) to test these trends, and understand why they may emerge in the first place.

5.1 Aggregator Trends

Figure 3 shows the inputs to the aggregator for different values of ε. The main notable point of the figure is the progression of the values for different FGSM strengths: as FGSM gets stronger, the concepts and relevances which had high values in latent space (white pixels) flip to having low values in latent space (black pixels), and vice versa, with an intermediate step where everything is roughly uniform.

To understand why this potentially causes the trend seen in Fig. 2, recall that the aggregator just takes the dot product of the concept vector and each of the classes' relevance vector and takes the softmax the resulting vector, picking the class with the largest softmax value as the SENN's output. Note that since the softmax is a monotonic function and we

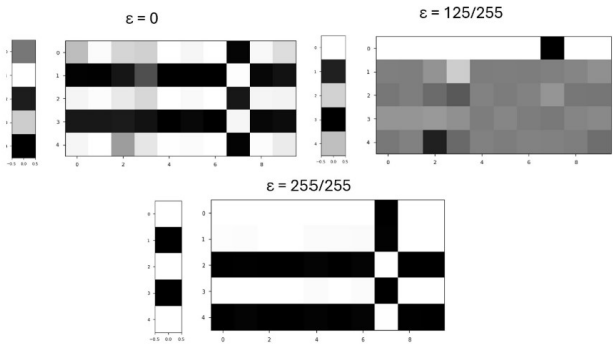

Figure 3: Concept vectors and relevance matrices as input into the aggregator for various values of ε. The 5×1 vector is the activation of each of the concepts, while the 5×10 matrix is the relevance of each concept for each class. Lighter pixels indicate values closer to 1, and darker pixels indicate values closer to 0.

greedily pick the largest softmax value (no sampling or other stochastic decoding occurs), this is equivalent to picking the class with the highest dot product.

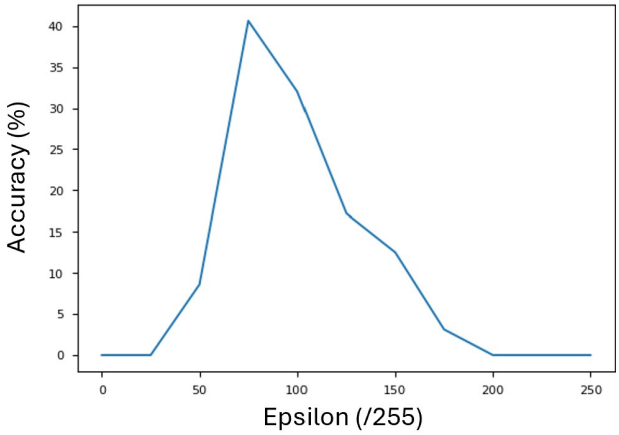

Figure 4: SENN accuracy for different values of ε when perturbing the aggregator and clamping the concept vector.

Further note that at $\varepsilon = 0$ and $\varepsilon = 255/255$, where the aggregator accuracy is the highest, the dot products for the correct class are by far the highest. Thus, the switching of these values from one extreme to the other in both the concept and relevance scores seems to be what allows the accuracy to increase again – the switching happens at the same rate in both the conceptizer and parameterizer, so the dot product remains approximately the same, and when the noise from the other dot products subsides, the correct class' dot product is once more the highest. The change in the value itself is due to the way FGSM works, in moving inputs in the opposite direction from their gradient's: if a value is low, it is likely that moving it opposite to its gradient will move it to be higher, and viceversa.

We test the causality of this dot product-switching hypothesis as follows. Let $c \in \mathbb{R}^{5\times 1}$ be the concept vector produced by the conceptizer and fed into the aggregator; for this experiment, instead of inputting c into the aggregator, we will input $\tilde{c} = 1 - c$. Since we hypothesize that this synchronous changing of values between the concept vector and the relevance vector for each class is what causes the shape of the accuracy curve, by disrupting this we should see the accuracy drop to 0 almost everywhere.

Fig. 4 shows the results of this experiment, and they are exactly what we would hope for if this switch in the values is what causes the accuracy to fluctuate as shown in Fig. 2. The peak in the center (which nonetheless does not exceed 40% accuracy) can be explained by noting that around those values of ε, the values in both the concept and relevance vectors are all close to 0.5, so $1 - c \approx c$ and the results are closer to the original benchmark. This peak is actually further proof of our hypothesis: if it was not there, and the line was fully flat at 0 accuracy throughout even when the dot product changed, it would be a sign that the dot product is correlated with the change, but not necessarily causal. This result, on the other hand, strongly motivates the causality of the hypothesis.

5.2 Autoencoder Trends

Similarly to the aggreagator's trends, it is now worth looking at why the two autoencoders' accuracy curves take the shapes that they do, and what this implies about the SENN architecture. Recall from Figure 2 that the conceptizer and parameterizer arrive at similar accuracies at the maximum value of ε, but relatively different curves.

We hypothesize this is due to what we term the **viable alternative hypothesis** – the hypothesis that SENNs are more vulnerable to noise that resembles the classes or the learned concepts ('chunked' or 'concept-based' noise) than they are random noise.

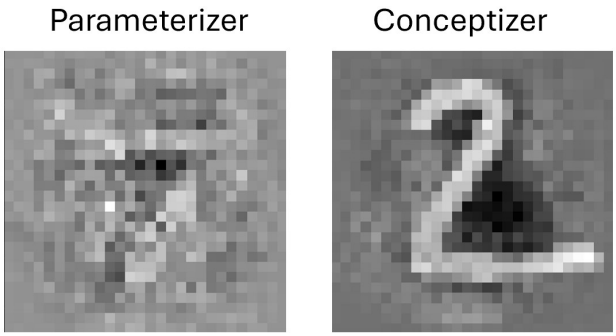

Figure 5: Visualizations of the gradients of the parameterizer and conceptizer when an image of a 7 is passed through them.

This is plausible because, as Fig. 5 shows, the conceptizer's gradient is concept-based, so this noise always provides a 'viable alternative' classification which consistently steers the model towards an incorrect prediction. Since the parameterizer's gradient is more noisy, at low ε it does not appear similar to any given concept for the SENN, so it noises all classes by about the same amount, the correct class is still usually by far the highest and accuracy is unaffected. Eventually, though around $\varepsilon \approx 125/255$, the noise becomes so strong that the SENN wrongly believes it to contain conceptual information, so it provides some 'viable alternative', and the component's accuracy starts to rapidly decrease.

To prove this, we can observe the distances between the probability of the most likely class for some given input and the second most likely class (as well as the third, fourth, etc.). If our hypothesis is correct, what we should see is that the gap between the probabilility of the first and second-highest classes decreases stably as ε increases for the conceptizer. For the parameterizer, we should see a bi-phasal trend: at the beginning, all distances should not change among themselves, only for the distance between the first and second classes to drop drastically in the right half of the graph.

Figure 6 shows a plot of the distances between the probabilities against the strength of the FGSM. Note that for the parameterizer, the distances between the top 2 and the first and third probabilities remain almost the same, meaning that the FGSM noise is being spread out across all classes instead of targeting one specific class. Recalling Fig. 2, in this low-epsilon regime, the parameterizer attack keeps the SENN's accuracy the same; in fact, the accuracy starts decreasing around $\varepsilon = 125/255$, which, as shown in Fig. 6, is when the distance between the top 2 probabilities starts decreasing much faster than the distance between the first and third probability. This means that, after this threshold, the SENN starts to interpret the parameterizer's FGSM noise as targeted towards a single class, and the accuracy starts plummeting.

This is strong evidence for our theory, and the conceptizer data strengthens it further: notice that here, because the gradient of the conceptizer resembles a concept in and of itself, a 'viable alternative' is always given, so the distance between the first and second probabilities always decreases faster than the distance between the first and third probabilities in a stable way, which matches up with the accuracy graph again, providing more evidence for the hypothesis.

6 Conclusion and Future Work

In conclusion, therefore, our investigation of the adversarial robustness of SENNs reveals potential features unique to the architecture that exploit every step of the process, from concept extraction to the final feature aggregation, in order to attack it.

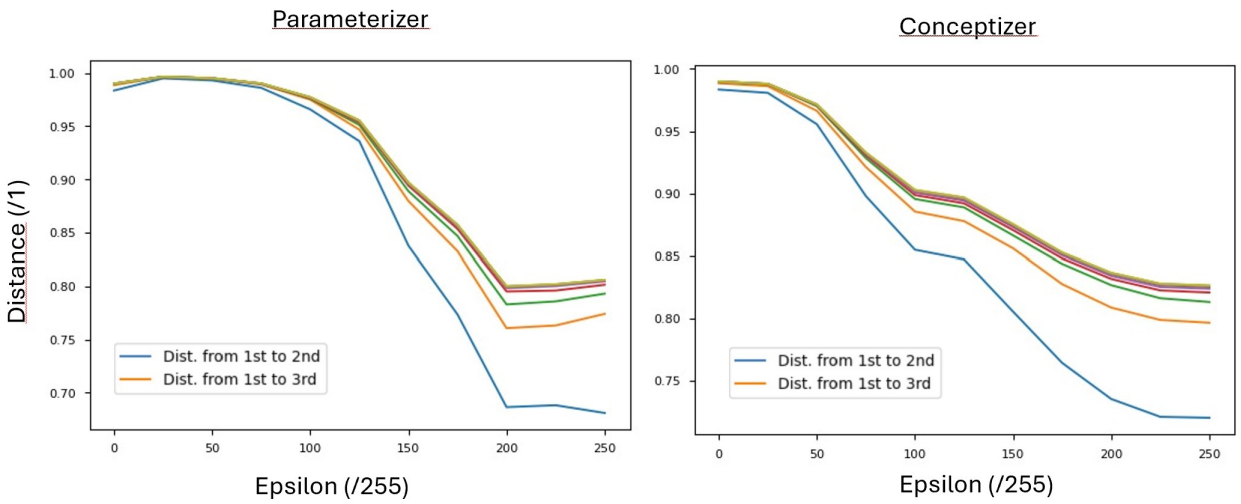

Figure 6: The distance between the top probability and all of the others (the second-highest, third-highest, etc.) across various values of ε for both parameterizer and conceptizer. We highlight the distances from the top probability to the second (in blue) and the top probability to the third (in orange).

When perturbing them with a traditional FGSM attack on MNIST, SENNs' weak point is their aggregator, which causes drastic drops in performance if its gradient is used to perturb the input (Fig. 2), and especially when used in conjunction with other components (Fig. 2).

Additionally, a more thorough analysis of all components' accuracy curves shows that gradient-based attacks may be particularly effective against SENNs due to gradients' potential to resemble concepts when used to perturb the input, a type of noise which we find SENNs are particularly weak to (Figs. 5 and 6). We call this weakness to concept-like noise the *viable alternative hypothesis*.

Therefore, users of SENNs should be aware that even attacks which do not specifically target the architecture can still be very effective against it, and that especially when there is a potential for attacks to a single component (e.g. SENN inference across a distributed system), users should be prepared to detect and contrast potential injections into the aggregator.

However, when compared to other methods, SENNs are quite robust – for the same FGSM attack, for example, the CNN baseline in [7] falls to 30% accuracy by $\varepsilon = 25/255$, while even with the worst attack (the parametrizer + aggregator joint perturbation) our SENNs are 97% accurate for the same ε. This is clearly an advantage of SENNs, but not necessarily a very practically useful one: CNNs are a much broader and more flexible base upon which to build defenses such as distillation [8], adversarial training [2], or gradient regularization [7], while SENNs' more rigid architecture limits the modifications and defenses possible and may therefore not be able to adapt as easily for other types of attacks.

In this work, we lay a rich foundation for future work. Here, we have shown that SENNs are more robust to random noise over 'chunked', concept-like noise: further work can explore targeted attacks for SENNs that use this limitation to render them even more vulnerable. On the other side, it would also be interesting to see if common adversarial defenses, like adversarial training, would prove effective: if we have to do adversarial training with concept-based noise, does that impact the concepts the SENN learns in trying to defend itself? And if it does, how does this affect the interpretability of the model, the entire *raison d'etre* of SENNs in the first place?

The results we have drawn are also quite reliant on the fact that FGSM is a gradient-based attack (e.g. the aggregator's accuracy profile is entirely based on this); this means that similar attacks, like PGD [4], are likely to behave similarly, but others, like DeepFool, [9], are not. Are there thus more general statements we can draw by incorporating these attacks and benchmarking SENNs on them?

References

[1] David Alvarez-Melis and Tommi S. Jaakkola. Towards robust interpretability with self-explaining neural networks. *Advances in Neural Information Processing Systems*, 31:7775–7784, 2018.

[2] Ian J. Goodfellow, Jonathon Shlens and Christian Szegedy. Explaining and Harnessing Adversarial Examples *International Conference on Learning Representations*, 2015.

[3] Yann LeCun, Léon Bottou, Yoshua Bengio and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86:11:2278-2324, 1998.

[4] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras and Adrian Vladu Towards Deep Learning Models Resistant to Adversarial Attacks *International Conference on Learning Representations*, 2018.

[5] Sanchit Sinha, Mengdi Huai, Jianhui Sun and Aidong Zhang Understanding and Enhancing Robustness of Concept-based Models *Conference of the Association for the Advancement of Artificial Intelligence*, 2023.

[6] Haizhong Zheng, Earlence Fernandes and Atul Prakash Analyzing the Interpretability Robustness of Self-Explaining Models *arXiv Preprint*, 1905.12429v3, 2019.

[7] Andrew Ross and Finale Doshi-Velez Improving the Adversarial Robustness and Interpretability of Deep Neural Networks by Regularizing Their Input Gradients *Conference of the Association for the Advancement of Artificial Intelligence: Humans and AI Track*, 2018.

[8] Nicolas Papernot, Patrick McDaniel, Xi Wu, Somesh Jha and Ananthram Swami Distillation as a Defense to Adversarial Perturbations against Deep Neural Networks *IEEE Symposium on Security and Privacy*, 2016.

[9] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi and Pascal Frossard DeepFool: a simple and accurate method to fool deep neural networks *IEEE Conference on Computer Vision and Pattern Recognition*, 2016.