# Robust Text Classification: Analyzing Prototype-Based Networks

Anonymous ACL submission


## Abstract

Downstream applications often require text classification models to be accurate and robust. While the accuracy of the state-of-the-art Language Models (LMs) approximates human performance, they often exhibit a drop in performance on noisy data found in the real world. This lack of robustness can be concerning, as even small perturbations in the text, irrelevant to the target task, can cause classifiers to incorrectly change their predictions. A potential solution can be the family of Prototype-Based Networks (PBNs) that classifies examples based on their similarity to prototypical examples of a class (prototypes) and has been shown to be robust to noise for computer vision tasks. In this paper, we study whether the robustness properties of PBNs transfer to text classification tasks under both targeted and static adversarial attack settings. Our results show that PBNs, as a mere architectural variation of vanilla LMs, offer more robustness compared to vanilla LMs under both targeted and static settings. We showcase how PBNs' interpretability can help us to understand PBNs' robustness properties. Finally, our ablation studies reveal the sensitivity of PBNs' robustness to how strictly clustering is done in the training phase, as tighter clustering results in less robust PBNs.


## 1 Introduction

Language models (LMs) are widely used in various NLP tasks and exhibit exceptional performance (Chowdhery et al., 2022; Zoph et al., 2022). In light of the need for real-world applications of these models, the requirements for robustness and interpretability have become urgent for both Large Language Models (LLMs) and fine-tuned LMs. More fundamentally, robustness and interpretability are essential components of developing trustworthy technology that can be adopted by experts in any domain (Wagstaff, 2012; Slack et al., 2022). However, LMs have limited interpretability by design (Zhao et al., 2023; Gholizadeh and Zhou, 2021), which cannot be fully mitigated by posthoc explainability techniques (Zini and Awad, 2022). Moreover, LMs lack robustness when exposed to text perturbations, noisy data, or distribution shifts (Jin et al., 2020; Moradi and Samwald, 2021). Reportedly, even LLMs lack robustness when faced with out-of-distribution data and noisy inputs (Wang et al., 2023), a finding that is supported by the empirical findings of this paper, too.

On this ground, NLP research has increasingly focused on benchmarks, methods, and studies that emphasize robustness and interpretability (e.g., Zhou et al., 2020; Jang et al., 2022; Liu et al., 2021). This has also been accompanied by the surge of focus on models that are inherently and architecturally interpretable and robust (e.g., Koh et al., 2020; Papernot and McDaniel, 2018; Keane and Kenny, 2019). An example of such models is the family of Prototype-Based Networks (PBNs) that is designed for robustness and interpretability (Li et al., 2018b). PBNs are based on the theory of categorization in cognitive science (Rosch, 1973), where it is governed by the graded degree of possessing prototypical features of different categories, with some members being more central (*prototypical*) than others. Consider, for example, classifying different types of birds. Then, pelican classification can be done through their prototypical tall necks and similarity to a prototypical pelican (Nauta et al., 2021a). Computationally, this idea is implemented by finding prototypical points or examples in the shared embedding space of data points and using the distance between prototypes and data points to accomplish the classification task. Aligned with how humans approach classification (Linzen, 2020), classifications in PBNs are expected to have human-like robustness because they classify through distances to prototypical examples found in the data. Leveraging distance between points helps to quantify prototypicality,

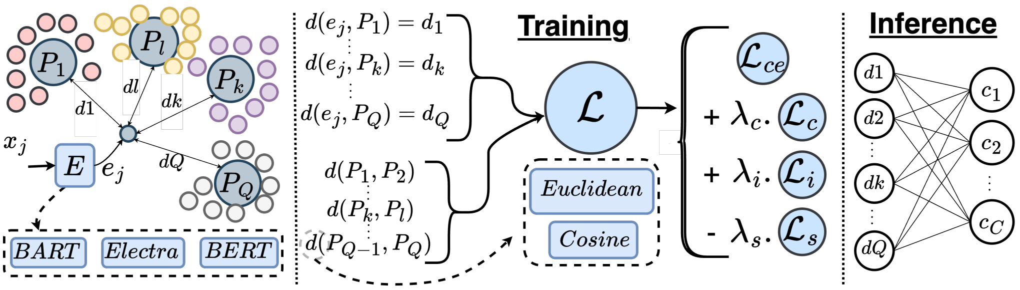


Figure 1: Classification by a PBN. The model computes distances between the new point and prototypes, $d(e_j, P_k)$, and distances within prototypes, $d(P_k, P_l)$, for both inference and training. During training, the model minimizes the loss term, $\mathcal{L}$, consisting of $\mathcal{L}_{ce}$, $\lambda_c \mathcal{L}_c$, $\lambda_i \mathcal{L}_i$, $\lambda_s \mathcal{L}_s$, controlling the importance of accuracy, clustering, interpretability, and separation of prototypes, based on all the computed distances; during inference, distances between the new point and prototypes are used for classification by a fully connected layer.

which then facilitates identifying noisy or out-of-distribution samples (Yang et al., 2018).

PBNs have been popular in Computer Vision (CV) tasks, including image classification (Angelov and Soares, 2020) and novel class detection (Hase et al., 2019). Inspired by PBNs in CV, NLP researchers have also developed PBN models for text classification, in particular, for sentiment classification (Pluciński et al., 2021; Ming et al., 2019; Hong et al., 2021), few-shot relation extraction (Han et al., 2021; Meng et al., 2023), and propaganda detection (Das et al., 2022). Yet, while competitive performance and interpretability of PBNs have been studied in both NLP (Das et al., 2022; Hase and Bansal, 2020) and CV (Gu and Ding, 2019; van Aken et al., 2022), their robustness advantages over vanilla models have only been investigated in CV (Yang et al., 2018; Saralajew et al., 2020; Vorácek and Hein, 2022).

In this study, *we investigate whether the robustness properties of PBNs transfer to NLP classification tasks*. In particular, our contributions are: (1) We adopt a modular and comprehensive approach to evaluate PBNs' robustness properties against various well-known adversarial attacks under both targeted and static adversarial settings; (2) We conduct a comprehensive analysis of the sensitivity of PBNs' robustness w.r.t. different hyperparameters.

Our experiments show that PBNs' robustness transfers to realistic perturbations in text classification tasks under both targeted and static adversarial settings and can, thus, enhance the text classification robustness of LMs. We note that the robustness boost that adversarial augmented training brings to LMs with access to additional pieces of relevant data, is higher than the boost caused by PBNs' architecture. Nevertheless, considering that the robustness boost in PBNs is only caused by their architecture without any additional resources (data or parameters), and this architecture is interpretable by design, the merits of such models can contribute to the field. Finally, benefiting from inherent interpretability, we showcase how PBN interpretability properties help to explain PBNs' robust behavior.

## 2 Prototype-Based Networks

PBNs classify data points based on their similarity to a set of *prototypes* learned during training. These prototypes summarize prominent semantic patterns of the dataset through two mechanisms: (1) prototypes are defined in the same embedding space as input examples, which makes them interpretable by leveraging input examples in their proximity; and (2) prototypes are designed to cluster semantically similar training examples, which makes them representative of the prominent patterns embedded in the data and input examples. The PBN's decisions, based on quantifiable similarity to prototypes, are robust as noise and perturbations are better reflected in the computed similarity to familiar prototypical patterns (Hong et al., 2020). Additionally, prototypes can provide insight during inference by helping users explain the model's behavior on input examples through the prototypes utilized for the model's prediction (Das et al., 2022).

**Inference.** Classification in PBNs is done via a fully connected layer applied on the measured distances between embedded data points and prototypes. As shown in Figure 1, given a set of data points $x_j, j \in \{1, \ldots, N\}$ with labels $y_j \in \{1, \ldots, C\}$, and $Q$ prototypes, PBNs first encode examples with a backbone $E$, resulting in the embedding $e_j = E(x_j)$. Next, PBNs compute the distances between prototypes and $e_j$ using the function $d$. These distances get fed into a fully connected layer to compute class-wise logits, incorporating the similarities to each prototype. Applying a softmax on top, the final outputs are $\hat{y}_c(x_j)$: probability that $x_j$ belongs to class $c \in \{1, \ldots, C\}$.

**Training.** The model is trained using objectives that simultaneously tweak the backbone parameters and the (randomly initialized) prototypes, thus promoting high performance and meaningful prototypes. To compute a total loss term $\mathcal{L}$, PBNs use the computed distances within prototypes $d(P_k, P_l)_{k \neq l}$, distances between all $Q$ prototypes and $N$ training examples given by $d(e_j, P_k)_{j \in \{1,\ldots,N\}; k \in \{1,\ldots,Q\}}$, and the computed probabilities $\hat{y}_c$. The prototypes and the weights in the backbone are adjusted according to $\mathcal{L}$. The total loss $\mathcal{L}$ consists of different inner loss terms that ensure high accuracy, clustering, interpretability, and low redundancy among prototypes; i. e., the classification loss $\mathcal{L}_{ce}$, the clustering loss $\mathcal{L}_c$ (Li et al., 2018b), the interpretability loss $\mathcal{L}_i$ (Li et al., 2018b), and separation loss $\mathcal{L}_s$ (Hong et al., 2020):

$$\mathcal{L} = \mathcal{L}_{ce} + \lambda_c \mathcal{L}_c + \lambda_i \mathcal{L}_i - \lambda_s \mathcal{L}_s, \quad (1)$$

where $\lambda_c, \lambda_i, \lambda_s \geq 0$ are regularization factors to adjust the contribution of the auxiliary loss terms.

*Classification loss* $\mathcal{L}_{ce}$ is defined as the cross-entropy loss between predicted and true labels:

$$\mathcal{L}_{ce} = -\sum_{j=1}^{N} \log(\hat{y}_{y_j}(x_j)). \quad (2)$$

*Clustering loss* $\mathcal{L}_c$ ensures that the training examples close to each prototype form a cluster of similar examples. In practice, $\mathcal{L}_c$ keeps all the training examples as close as possible to at least one prototype and minimizes the distance between training examples and their closest prototypes:

$$\mathcal{L}_c = \frac{1}{N} \sum_{j=1}^{N} \min_{k \in \{1,\ldots,Q\}} d(P_k, e_j). \quad (3)$$

*Interpretability loss* $\mathcal{L}_i$ ensures that the prototypes are interpretable by minimizing the distance to their closest training sample:

$$\mathcal{L}_i = \frac{1}{Q} \sum_{k=1}^{Q} \min_{j \in \{1,\ldots,N\}} d(P_k, e_j). \quad (4)$$

Keeping the prototypes close to training samples allows PBNs to represent a prototype by its closest training samples that are domain-independent and enable analysis by task experts.

| Original text | Perturbed text |
|---|---|
| A gentle breeze rustled the leaves. | A gèntle wind rustled the lEaves. |
| rescue Engineer Company | Res©ue operation Company |
| embarrassingly foolish | embarrassing1y fo0lish |

Table 1: Examples of adversarial perturbations, with the perturbed tokens highlighted.

*Separation loss* $\mathcal{L}_s$ maximizes the inter-prototype distance to reduce the probability of redundant prototypes:

$$\mathcal{L}_s = \frac{2}{Q(Q-1)} \sum_{k,l \in \{1,\ldots,Q\}; k \neq l;} d(P_k, P_l). \quad (5)$$

## 3 Robustness Evaluation

We assess PBNs' robustness against adversarial perturbations of original input text that are intended to preserve the text's original meaning. The perturbations change the classification of the target model upon confronting these perturbed examples from the correct behavior to an incorrect one in an effective and efficient way (Dalvi et al., 2004; Kurakin et al., 2017a,b; Li et al., 2023). Automatic approaches of finding these perturbations vary (Zhang et al., 2020): perturbations can be focused on different **granularities**, i.e., *character-level*, *word-level*, or *sentence-level*; their **generation** can be done in different ways, e.g., *replacing*, *inserting*, *deleting*, *swapping* tokens; they can have different **searching strategies** for their manipulations, such as *context-aware* or *isolated* approaches; and also various **salient token identification strategies** to maximize their adversarial effect.

Orthogonally, these adversarial perturbations are divided into **targeted** and **static**. In the targeted setting, the attacker has access to the target model and can attack it directly (Si et al., 2021). However, in the static setting, the attacker does not have access to the target model. Hence, adversarial perturbations are gathered while attacking external models that the attacker has access to, and the gathered successful perturbations would be used to assess the robustness of the target model (Wang et al., 2022a).

With numerous adversarial perturbation strategies in the literature (Zhang et al., 2020; Wang et al., 2022c), each with unique advantages (e.g., effectiveness vs. efficiency), we use a wide range of existing perturbation strategies in this study. These cover the aforementioned **granularities**, **generation strategies**, **searching strategies**, and **salient token identification strategies**, under both **tar-**

**geted**, and **static** settings. See examples of adversarial perturbations covered in our study in Table 1.

# 4 Experimental Setup

## 4.1 Datasets

PBNs classify instances based on their similarity to prototypes learned during training that summarize prominent semantic patterns in a dataset. Thus, with more classes, we might need more prototypes to govern the more complex system between instances and prototypes (Yang et al., 2018). To study the interplay between the number of classes and robustness, we employ three datasets: (1) *IMDB reviews* (Maas et al., 2011): a binary sentiment classification dataset; (2) *AG_NEWS* (Gulli): a collection of news articles that can be associated with four categories; (3) *DBPedia*:[1] a dataset with taxonomic, hierarchical categories for Wikipedia articles (Lehmann et al., 2015), with nine classes. We use these three datasets to study the robustness of PBNs under both targeted and static adversarial settings. As an additional source of static adversarial perturbations, we adopt the SST-2 binary classification split from the existing *Adversarial GLUE (AdvGLUE)* dataset (Wang et al., 2022a), consisting of perturbed examples of different granularities, filtered both automatically and by human evaluation for more effectiveness. For statistics of the datasets and their perturbations, see Appendix A.

## 4.2 Perturbations

**Attacking strategies.** We selected five well-established adversarial attack methods: BAE (Garg and Ramakrishnan, 2020), TextFooler (Jin et al., 2020), TextBugger (Li et al., 2018a), DeepWordBug (Gao et al., 2018), and PWWS (Ren et al., 2019).[2] As mentioned in Section 3, these attacks cover a wide range of **granularities** (e.g., character-based in DeepWordBug and word-based in PWWS), **generation strategies** (e.g., word substitution in PWWS and TextFooler and deletion in TextBugger), **searching strategies** (e.g., context-aware in BAE and isolated synonym-based in TextFooler), and **salient token identification strategies** (e.g., finding the important sentences first and then words in TextBugger and finding the important words to change in BAE).

**Targeted perturbations.** In this setting, the adversarial attacks are directly conducted against PBNs and vanilla LMs trained on original datasets. For each attack strategy, we aim for 800 successful perturbations and report the robustness of PBNs against adversarial attacks by Attack Success Rate (ASR; Wu et al., 2021) and Average Percentage of Words Perturbed (APWP; Yoo et al., 2020) to reach the observed ASR. Successful perturbations are those that change the prediction of a target model already fine-tuned on that dataset from the correct prediction to the wrong prediction.

**Static perturbations.** In this setting, the adversarial attacks are conducted on external models: BERT (Devlin et al., 2018), RoBERTa (Liu et al., 2019), and DistilBERT (Sanh et al., 2019), which are trained on the original datasets, and a compilation of the successful perturbations on those models is used to assess the robustness of PBNs against the studied adversarial attacks by their accuracy on the perturbations, similar to the study by Wang et al. (2022a). To obtain the perturbations, each model is fine-tuned on each dataset, and 800 successful perturbations for each attack strategy are obtained. We focus on examples whose perturbations are predicted incorrectly by all three models to maximize the generalizability of this static set of perturbations to a wider range of unseen target models. In principle, the perturbations for each model are different, yielding three variations per original example for a dataset-perturbation pair. For instance, focusing on DBPedia and BAE attack strategy, after 800 successful perturbations for each of the three target models, the perturbations of 347 original examples could change all models' predictions, resulting in a total of 1401 ($3 \times 347$) perturbations compiled for BAE attack strategy and DBPedia dataset.

## 4.3 PBNs' Hyperparameters

**Backbone ($E$).** Prototype alignment and training are highly dependent on the quality of the latent space created by the backbone encoder $E$, which in turn affects the performance, robustness, and interpretability of PBNs. We consolidate previous methods for text classification using PBNs (Pluciński et al., 2021; Das et al., 2022; Ming et al., 2019; Hong et al., 2020) and consider three backbone architectures: BERT (Devlin et al., 2018), BART encoder (Lewis et al., 2019), and Electra (Clark et al., 2020). Based on our empirical evidence, fine-tuning all the layers of the backbone

[1] https://bit.ly/3RgX41H

[2] We also employed paraphrased-based perturbations (Lei et al., 2019), generated by GPT3.5 (OpenAI, 2022). However, both our baselines and PBNs were robust to these perturbations, and we include them in the Appendix in Table 6.

was causing the PBNs' training not to converge. Hence, we freeze all the layers of the backbones except for the last layer when training.

**Distance function ($d$).** The pairwise distance calculation quantifies how closely the prototypes are aligned with the training examples (Figure 1). In recent work, Euclidean distance has been shown to be better than Cosine distance for similarity calculation (van Aken et al., 2022; Snell et al., 2017) as it helps to align prototypes closer to the training examples in the encoder's latent space. However, with some utilizing Cosine distance (Chen et al., 2019) while others prioritizing Euclidean distance (Mettes et al., 2019), and the two having incomparable experimental setups, conclusive arguments about the superiority of one over the other cannot be justified, and the choice of distance function is usually treated as a hyperparameter. Accordingly, we hypothesize that the impact of $d$ will be significant in our study of robustness, and hence, we consider both Cosine and Euclidean distance functions when training PBNs.

**Number of prototypes ($Q$).** Number of prototypes in PBNs is a key factor for mapping difficult data distributions (Yang et al., 2018; Sourati et al., 2023). Hence, to cover a wide range, we consider five values for $Q = \{2, 4, 8, 16, 64\}$.

**Objective functions ($\mathcal{L}$).** Given the partly complementary goals of loss terms, we investigate the effect of interpretability, clustering, and separation loss on PBNs' robustness, keeping the accuracy constraint ($\mathcal{L}_{ce}$) intact. To do so, we consider three values, $\{0, 0.9, 10\}$ for $\lambda_i$, $\lambda_c$, and $\lambda_s$. $0$ value represents the condition where the corresponding loss function is not being utilized in the training process. $0.9$ value was empirically found to offer good accuracy, clustering, and interpretability, across datasets and was also motivated by prior works (Das et al., 2022). $10$ value was chosen as an upper bound dominating the corresponding loss objective (e.g., interpretability) in the training process.

### 4.4 Baselines

Since PBNs are architectural enhancements of vanilla LMs using learned prototypes for classification instead of a traditional softmax layer used in vanilla LMs, **vanilla LMs** employed as PBNs' backbones serve as a baseline for comparing the robustness of PBNs. We also employ **adversarial augmented training** (Goyal et al., 2023) on top of the vanilla LMs as another baseline. Note that the same layers frozen for PBNs' training are also frozen for the baselines. As we need additional data for such extra training, we use this baseline under static perturbations, where the set of perturbations has already been compiled beforehand. Finally, although we note that LLMs are more appropriate choices for generic chat and text generation due to their decoder-only architecture, and fine-tuned LMs are still superior to LLMs when it comes to task-oriented performance (Chang et al., 2024), we compare PBNs with two LLMs, namely, GPT4o (AI, 2024) and Llama3 (AI@Meta, 2024).

## 5 Results

### 5.1 Robustness of PBNs

The robustness report of PBNs under both targeted adversarial attacks and static attacks under different experimental setups (i.e., datasets, backbones, and attack strategies), using the best hyperparameters is presented in Table 2. [3] [4] Best hyperparameters were chosen among the permutation of all hyperparameters presented in Section 4.3 to yield the highest robustness (lowest ASR or highest accuracy). Under the targeted adversarial attack setting, our results showed that PBNs are more robust than vanilla LMs (having lower ASR) regardless of the utilized backbone, dataset, or attacking strategy. We also saw similar trends analyzing the robustness of PBNs compared to vanilla LMs, averaging over all PBN hyperparameters (find the details in Table 8). Focusing on the APWP metric, we observed that in $71.0\%$ of the conditions, the PBNs' robustness was greater than vanilla LMs (having higher APWP), and this superiority dropped to $31.0\%$ of the conditions when averaging over all the hyperparameters (find the details in Table 7), which suggested that PBNs' robustness is sensitive to hyperparameters involved in training.

We observed similar trends under static adversarial attacks, where the PBNs' robustness was higher than vanilla LMs (having higher accuracy under attack) in the majority of the conditions ($93.7\%$ of all variations of experimental setups and hyperparameters). We observed that in every experimental condition (dataset and attack strategy), a PBN exists with a robustness outperforming LLMs like GPT4o (AI, 2024) and Llama3 (AI@Meta,

[3] The semantic similarity between original and perturbed texts using OpenAI text-embedding-ada-002 across all datasets and attack types was 0.97 ($SD = 0.01$).

[4] Our results showed that adversarial perturbations from TextFooler and PWWS were more effective than others.

Targeted Attacks; Attack Success Rate (ASR %) reported

| | AG_News | | | | | DBPedia | | | | | IMDB | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | BAE | DWB | PWWS | TB | TF | BAE | DWB | PWWS | TB | TF | BAE | DWB | PWWS | TB | TF |
| BART | 14.8 | 53.2 | 53.6 | 31.8 | 76.5 | 18.9 | 28.3 | 43.1 | 21.1 | 71.9 | 74.1 | 74.7 | 99.3 | 78.5 | 100.0 |
| + PBN | **11.1** | **32.3** | **41.3** | **23.1** | **62.2** | **15.2** | **14.7** | **28.7** | **12.6** | **45.5** | **36.1** | **41.0** | **75.9** | **41.3** | **73.1** |
| BERT | 17.0 | 78.0 | 69.8 | 45.7 | 88.8 | 13.9 | 24.8 | 31.6 | 22.0 | 61.3 | 82.5 | 79.7 | 99.9 | 83.9 | 99.9 |
| + PBN | **7.7** | **42.6** | **47.0** | **30.4** | **70.5** | **9.8** | **17.3** | **21.6** | **13.0** | **41.0** | **42.8** | **41.0** | **79.7** | **57.7** | **79.8** |
| ELEC. | 24.8 | 89.5 | 69.1 | 87.8 | 87.9 | 14.5 | 42.8 | 45.6 | 42.3 | 75.3 | 52.5 | 49.2 | 95.3 | 67.8 | 99.3 |
| + PBN | **14.0** | **34.9** | **42.9** | **51.8** | **70.2** | **7.8** | **11.5** | **17.8** | **19.1** | **35.6** | **28.9** | **27.4** | **66.6** | **36.8** | **78.0** |

Static Attacks; Accuracy (%) reported

| | AG_News | | | | | DBPedia | | | | | IMDB | | | | | SST2 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | BAE | DWB | PWWS | TB | TF | BAE | DWB | PWWS | TB | TF | BAE | DWB | PWWS | TB | TF | GLUE |
| BART | 53.2 | 76.7 | 83.2 | 77.5 | 85.8 | 55.5 | 68.6 | 58.4 | 72.5 | 71.3 | 74.1 | 80.5 | 83.6 | 85.8 | 87.6 | 29.8 |
| + PBN | 57.6 | **80.6** | 84.8 | **79.2** | 88.8 | 65.0 | 71.6 | 65.7 | 78.4 | 74.8 | 80.4 | 81.3 | 86.3 | 89.3 | 90.4 | **50.4** |
| + Aug. | **71.7** | 78.4 | **85.5** | 77.6 | **90.1** | **84.0** | **79.6** | **89.7** | **88.8** | **94.0** | **85.7** | **86.7** | **92.9** | **89.9** | **96.5** | - |
| BERT | 47.8 | 64.0 | 75.9 | 69.4 | 80.7 | 62.3 | 61.4 | 75.4 | 78.4 | 82.0 | 75.1 | 77.1 | 85.0 | 83.4 | 85.9 | 42.0 |
| + PBN | 52.9 | 70.4 | **78.5** | **73.8** | 84.3 | 66.9 | 66.6 | 80.3 | 82.0 | 85.8 | 77.6 | **79.1** | 85.3 | 85.0 | 86.5 | **51.1** |
| + Aug. | **58.3** | **71.6** | 78.3 | 71.2 | **85.4** | **75.5** | **70.9** | **84.1** | **90.5** | **91.0** | **83.2** | 77.6 | **91.7** | **90.8** | **89.2** | - |
| ELEC. | 50.4 | 65.0 | 73.5 | 63.9 | 77.8 | 79.7 | 66.9 | 80.9 | 81.4 | 84.4 | 89.7 | 90.3 | 94.6 | 94.5 | 95.6 | 44.3 |
| + PBN | **64.6** | **74.1** | **85.1** | **77.2** | **89.0** | 78.7 | 69.8 | 79.3 | 82.5 | 85.8 | **90.0** | 90.8 | 94.6 | **95.5** | **96.3** | **65.6** |
| + Aug. | 55.0 | 59.5 | 71.7 | 61.6 | 79.5 | **86.2** | **73.8** | **88.1** | **84.5** | **92.8** | 89.4 | **93.7** | **95.3** | 94.9 | 95.8 | - |
| GPT4o | 57.1 | 73.3 | 73.0 | 76.5 | 79.9 | 66.0 | 63.4 | 61.0 | 69.0 | 44.0 | 87.0 | 89.5 | 91.2 | 93.7 | 94.2 | 59.8 |
| Llama3 | 57.6 | 56.4 | 55.0 | 65.9 | 62.8 | 44.0 | 53.7 | 37.8 | 45.0 | 44.4 | 82.0 | 86.0 | 93.2 | 89.0 | 91.5 | 56.0 |

Table 2: Comparison of PBNs and vanilla LMs (+ vanilla LMs with adversarial augmented training under static attack setting) under both targeted and static adversarial attack perturbations, using the best hyperparameters for PBNs, on IMBD, AG_News, DBPedia (+ SST-2 from AdvGLUE under static attack setting) datasets, under BAE, DeepWordBug (DWB), PWWS, TextBugger (TB), TextFooler (TF). The highest accuracy and lowest ASR showing the superior model for each architecture is **boldfaced**, and the second best model is underlined for static attacks.

2024) that have orders of magnitude more parameters and are not interpretable by design as opposed to PBNs. Vanilla LMs with adversarial augmented training demonstrated greater robustness than PBNs in 71.2% of the conditions. This highlighted the more effective role of additional data in adversarial augmented training compared to PBNs' robust architecture and makes PBNs a preferable choice when efficiency is prioritized (Goodfellow et al., 2014). Analyzing PBNs' robustness under the static adversarial setting averaging over all PBNs' hyperparameters, our results showed that in only 31.2% of the conditions, PBNs have greater robustness compared to vanilla LMs (find the details in Table 8), which similar to observations on APWP, suggested that PBNs' robustness is sensitive to hyperparameters involved in the training.

To sum up, we observed that PBNs consistently and over different metrics were more robust compared to vanilla LMs and LLMs, using the best hyperparameters without sacrificing performance on the original unperturbed samples (find performance on original datasets in Table 6). We believe that the observed robust behavior is due to the design of the PBN architecture. Standard neural networks for text classification distinguish classes by drawing hyperplanes between samples of different classes that are prone to noise (Yang et al., 2018), especially when dealing with several classes. Instead, PBNs are inherently more robust since they perform classification based on the similarity of data points to prototypes, acting as class centroids. Finally, we observed that the robustness superiority of PBNs compared to vanilla LMs dropped when averaging over all the possible hyperparameters, which is what we investigate further in Section 5.2.

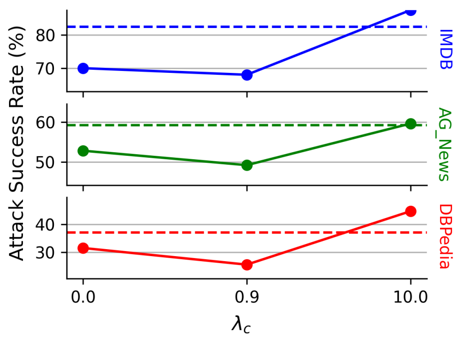

Figure 2: Attack Success Rate (ASR %) of PBNs with different $\lambda_c$ values adjusting the importance of clustering in the trained PBNs, with other hyperparameters set to their best values, and averaged across other possible variables (e.g., backbone and attack type). The dotted line represents the ASR for the non-PBN model.

## 5.2 Sensitivity to Hyperparameters

We studied the sensitivity of PBNs' robustness to the hyperparameters involved in training, covering

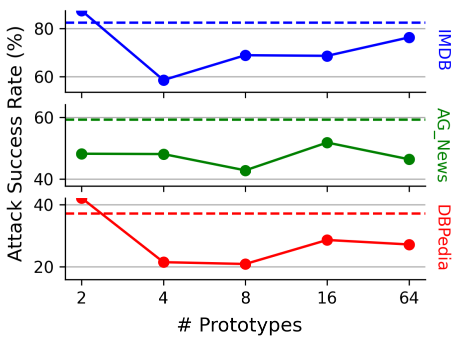


Figure 3: Attack Success Rate (ASR %) of PBNs with different numbers of prototypes, with other hyperparameters set to their best values, and averaged across other possible variables (e.g., backbone and attack type). Dotted line represents the ASR for the non-PBN model.

| Proto. | Representative Training Examples | Label |
|---|---|---|
| $P_0$ | **Handly's Lessee v. Anthony (1820)**: Determined Indiana-Kentucky boundary. | UnitWork |
| | **Rasul v. Bush (2004)**: Decided jurisdiction over Guantanamo detainees. | UnitWork |
| $P_1$ | **Marine Corps Air Station Futenma**: U.S. Marine Corps base, Ginowan, Okinawa; regional military hub. | Place |
| | **Özdere**: Turkish coastal resort town in İzmir Province, popular among tourists. | Place |
| $P_2$ | **Yevgeni Viktorovich Balyaikin**: Russian footballer for FC Tom Tomsk. | Agent |
| | **Gigi Morasco**: Fictional character on ABC's One Life to Live. | Agent |

Table 3: Examples of prototypes, their closest training examples, alongside their label derived from their closest training examples, extracted from a PBN with 16 prototypes and a BART backbone on DBPedia. Note that the presented training examples are the summarization of their longer version for easier interpretation.

values discussed in Section 4.3. Focusing on each hyperparameter, the value for the other ones was selected to yield the best performance so that, overall, we could better depict the sensitivity and limiting effect of the hyperparameter of interest. We did not observe any sensitivity from PBNs with respect to the backbone, interpretability term ($\lambda_i$; see Section C.5), separation term ($\lambda_s$; see Section C.7), and the distance function ($d$; see Section C.4).

However, as presented in Figure 2, we observed that higher values of $\lambda_c$, promoting tighter clustering of input examples around prototypes, hinder PBNs' robustness. Clustering loss is a regularization term that encourages samples to be close to prototypes in the embedding space, further enhancing interpretability but potentially reducing accuracy by narrowing the diversity in embedding space, which is a common phenomenon in loss terms of competing goals. The mean and standard deviation over (transformed) distances between prototypes and samples can be used to describe the spread of embedded data points around prototypes. These values are $(-0.24 \pm 1.7) \times 10^{-7}$ with $\lambda_c = 0.9$, and $(-0.18 \pm 1.5) \times 10^{-6}$ with $\lambda_c = 10$, showing less diverse prototypes indicated by smaller measured distances caused by stronger clustering.

Additionally, as depicted in Figure 3, we observed poor robustness from PBNs when the number of prototypes is as low as two, which is intuitive as a low number of prototypes also means a lower number of semantic patterns learned, which constraints the PBNs' abilities to distinguish between different classes. Noting that more prototypes add to the complexity and size of the network as a whole, the observed stable trend of the robustness with the higher number of prototypes ($> 2$) suggests that as long as the number of prototypes is not too low, PBNs with lower number of prototypes can be preferred. This corroborates with the studies performed by Yang et al. (2018). Finally, note that the same analysis using other metrics (e.g., APWP) and under static adversarial setting (using accuracy as the studied metric) depicted the same trend and can be found in Section C.6 and Section C.8.

### 5.3 PBNs' Interpretability w.r.t. Robustness

PBNs are interpretable by design, and we can understand their behavior through the distance of input examples to prototypes and the importance of these distances, extracted by the last fully connected layer of PBNs transforming vector of distances to log probabilities for classes. Examples of learned prototypes that can be represented by their closest training input examples are shown in Table 3. These input examples help the user identify the semantic features that the prototypes are associated with, which by our observations in our case, were mostly driven by the class label of the closest training examples.

We can also benefit from interpretable properties of PBNs to better understand their robustness properties, regardless of the success of perturbations. Table 4 illustrates predictions of a PBN on three original and perturbed examples from the DBPedia dataset, alongside the top-2 prototypes that were utilized by the PBN's fully connected layer for prediction and prototypes' associated label (by their closest training examples). In the first two examples, PBN correctly classifies both the original and perturbed examples, and from the top-2 prototypes, we observe that this is due to unchanged prototypes

| Text | Activ. Proto.s | Proto.s Labels | Pred. | Label |
|---|---|---|---|---|
| **Roman Catholic Diocese of Barra**: Diocese in Barra, Feira de Santana province, Brazil. | $P_1, P_{14}$ | Place, Place | Place | Place |
| **Roman Catholic Bishop of Barra**: Episcopal seat in Barra, Feira de Santana province, Brazil. | $P_1, P_{14}$ | Place, Place | Place | Place |
| **Inta Ezergailis**: Latvian American professor emerita at Cornell University. | $P_2, P_8$ | Agent, Agent | Agent | Agent |
| **Inta Ezergailis**: Latvian American poet and scholar at Cornell University. | $P_2, P_7$ | Agent, Work | Agent | Agent |
| **Saint Eigrad**: 6th-century Precongregational North Wales saint and Patron Saint of Llaneigrad. | $P_2, P_8$ | Agent, Agent | Agent | Agent |
| ***St Eigrad**: 6th-century Precongregational street of North Wales and Patron Saint of Llaneigrad.* | $P_1, P_{14}$ | Place, Place | Place | Agent |

Table 4: Examples of the original test (top) and adversarially perturbed examples (bottom) of DBPedia using TextFooler, classified by a PBN, alongside the top-2 activated prototypes by the PBN's fully connected layer and their associated labels. Incorrectly predicted examples are in *italic*.

utilized in prediction. However, in the last example, the model incorrectly classifies an example that is associated with an Agent as a Place. Interestingly, this incorrect behavior can be explained by the change in the top-2 activated prototypes, where they are changing from Agent-associated to Place-associated prototypes because of the misspelling of "saint" with "street." Thus, the use of prototypes not only enhances our understanding of the model's decision-making process but also unveils how minor perturbations influence the model's predictions.

## 6 Related Work

**Robustness evaluation.** Robustness in NLP is defined as models' ability to perform well under noisy (Ebrahimi et al., 2018) and out-of-distribution data (Hendrycks et al., 2020). With the wide adoption of NLP models in different domains and their near-human performance on various benchmarks (Wang et al., 2019; Sarlin et al., 2020), concerns have shifted towards models' performance facing noisy data (Wang et al., 2022a,b). Studies have designed novel and effective adversarial attacks (Jin et al., 2020; Zhang et al., 2020), defense mechanisms (Goyal et al., 2023; Liu et al., 2020), and evaluations to better understand the robustness properties of NLP models (Wang et al., 2022a; Morris et al., 2020a). These evaluations are also being extended to LLMs, as they similarly lack robustness (Wang et al., 2023; Shi et al., 2023). While prior work has studied LMs' robustness, to our knowledge, PBNs' robustness properties have not been explored yet. Our study bridges this gap.

**Prototype-based networks.** PBNs are widely used in CV (Chen et al., 2019; Hase et al., 2019; Kim et al., 2021; Nauta et al., 2021b; Pahde et al., 2021) because of their interpretability and robustness properties (Soares et al., 2022; Yang et al., 2018). While limited work has been done in the NLP domain, PBNs have recently found application in text classification tasks such as propaganda detection (Das et al., 2022), logical fallacy detection (Sourati et al., 2023), sentiment analysis (Pluciński et al., 2021), and few-shot relation extraction (Meng et al., 2023). ProseNet (Ming et al., 2019), a prototype-based text classifier, uses several criteria for constructing prototypes (He et al., 2020), and a special optimization procedure for better interpretability. ProtoryNet (Hong et al., 2020) leverages RNN-extracted prototype trajectories and deploys a pruning procedure for prototypes, and ProtoTex (Das et al., 2022) uses negative prototypes for handling the absence of features for classification. While PBNs are expected to be robust to perturbations, this property has not been systematically studied in NLP. Our paper consolidates PBN components used in prior studies and studies their robustness in different adversarial settings.

## 7 Conclusions

Inspired by the lack of robustness to noisy data of state-of-the-art LMs and LLMs, we study the robustness of PBNs, as an architecturally robust variation of LMs, against both targeted and static adversarial attacks. We find that PBNs are more robust than vanilla LMs and even LLMs such as Llama3, both under targeted and static adversarial attack settings. Our results suggest that this robustness can be sensitive to hyperparameters involved in PBNs' training. More particularly, we note that a low number of prototypes and tight clustering conditions limit the robustness capacities of PBNs. Additionally, benefiting from the inherently interpretable architecture of PBNs, we showcase how learned prototypes can be utilized for robustness and also for gaining insights about their behavior facing adversarial perturbations, even when PBNs are wrong. In summary, our work provides encouraging results for the potential of PBNs to enhance the robustness of LMs across a variety of text classification tasks and quantifies the impact of architectural components on PBN robustness.

## Limitations

Although we cover a wide range of adversarial perturbations and strategies for their generation, we acknowledge that more complicated perturbations can also be created that are more effective and help the community have a more complete understanding of the models' robustness. Hence, we do not comment on the generalizability of our study to all possible textual perturbations besides our evaluation on AdvGLUE. Moreover, although it is customary in the field to utilize prototype-based networks for classification tasks, their application and robustness on other tasks remain to be explored. Furthermore, while we attempt to use a wide variety of backbones for our study, we do not ascertain similar patterns for all possible PBN backbones and leave this study for future work. Finally, we encourage more exploration of the interpretability of these models under different attacks to better understand the interpretability benefits of models when analyzing robustness.

## Ethical Considerations

Although the datasets and domains we focus on do not pose any societal harm, the potential harm that is associated with using the publicly available tools we used in this study to manipulate models in other critical domains should be considered. Issues surrounding anonymization and offensive content hold importance in data-driven studies, particularly in fields like natural language processing. Since we utilize datasets like IMDB, AG_News, DBPedia, and AdvGLUE that are already content-moderated, there is no need for anonymization of data before testing for robustness in this study.

## References


Open AI. 2024. Hello GPT-4o. https://openai.com/index/hello-gpt-4o/. [Accessed 15-06-2024].

AI@Meta. 2024. Llama 3 model card.

Plamen Angelov and Eduardo Soares. 2020. Towards explainable deep neural networks (xdnn). *Neural Networks*, 130:185–194.

Yupeng Chang, Xu Wang, Jindong Wang, Yuan Wu, Linyi Yang, Kaijie Zhu, Hao Chen, Xiaoyuan Yi, Cunxiang Wang, Yidong Wang, et al. 2024. A survey on evaluation of large language models. *ACM Transactions on Intelligent Systems and Technology*, 15(3):1–45.

Chaofan Chen, Oscar Li, Chaofan Tao, Alina Jade Barnett, Jonathan Su, and Cynthia Rudin. 2019. *This Looks like That: Deep Learning for Interpretable Image Recognition*. Curran Associates Inc., Red Hook, NY, USA.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, et al. 2022. Palm: Scaling language modeling with pathways. *arXiv preprint arXiv:2204.02311*.

Kevin Clark, Minh-Thang Luong, Quoc V Le, and Christopher D Manning. 2020. Electra: Pre-training text encoders as discriminators rather than generators. *arXiv preprint arXiv:2003.10555*.

Nilesh Dalvi, Pedro Domingos, Mausam, Sumit Sanghai, and Deepak Verma. 2004. Adversarial classification. KDD '04, page 99–108, New York, NY, USA. Association for Computing Machinery.

Anubrata Das, Chitrank Gupta, Venelin Kovatchev, Matthew Lease, and Junyi Jessy Li. 2022. ProtoTEx: Explaining model decisions with prototype tensors. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2986–2997, Dublin, Ireland. Association for Computational Linguistics.

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

Javid Ebrahimi, Anyi Rao, Daniel Lowd, and Dejing Dou. 2018. HotFlip: White-box adversarial examples for text classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 31–36, Melbourne, Australia. Association for Computational Linguistics.

Ji Gao, Jack Lanchantin, Mary Lou Soffa, and Yanjun Qi. 2018. Black-box generation of adversarial text sequences to evade deep learning classifiers. In *2018 IEEE Security and Privacy Workshops (SPW)*, pages 50–56. IEEE.

Siddhant Garg and Goutham Ramakrishnan. 2020. BAE: BERT-based adversarial examples for text classification. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6174–6181, Online. Association for Computational Linguistics.

Shafie Gholizadeh and Nengfeng Zhou. 2021. Model explainability in deep learning based natural language processing.

Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. 2014. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*.

Shreya Goyal, Sumanth Doddapaneni, Mitesh M Khapra, and Balaraman Ravindran. 2023. A survey of adversarial defenses and robustness in nlp. *ACM Computing Surveys*, 55(14s):1–39.

Xiaowei Gu and Weiping Ding. 2019. A hierarchical prototype-based approach for classification. *Information Sciences*, 505:325–351.

Antonio Gulli. AG's Corpus of News Articles. groups.di.unipi.it/~gulli/AG_corpus_of_news_articles.html. Accessed 15 June 2024.

Jiale Han, Bo Cheng, and Wei Lu. 2021. Exploring task difficulty for few-shot relation extraction. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 2605–2616, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Peter Hase and Mohit Bansal. 2020. Evaluating explainable AI: Which algorithmic explanations help users predict model behavior? In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics – ACL2020*, pages 5540–5552, Online. Association for Computational Linguistics.

Peter Hase, Chaofan Chen, Oscar Li, and Cynthia Rudin. 2019. Interpretable image recognition with hierarchical prototypes. In *Proceedings of the AAAI Conference on Human Computation and Crowdsourcing*, volume 7, pages 32–40.

Junxian He, Taylor Berg-Kirkpatrick, and Graham Neubig. 2020. Learning sparse prototypes for text generation. *Advances in Neural Information Processing Systems*, 33:14724–14735.

Dan Hendrycks, Xiaoyuan Liu, Eric Wallace, Adam Dziedzic, Rishabh Krishnan, and Dawn Song. 2020. Pretrained transformers improve out-of-distribution robustness. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2744–2751, Online. Association for Computational Linguistics.

Dat Hong, Stephen S Baek, and Tong Wang. 2020. Interpretable sequence classification via prototype trajectory. *arXiv preprint arXiv:2007.01777*.

Dat Hong, Stephen S. Baek, and Tong Wang. 2021. Interpretable sequence classification via prototype trajectory.

Myeongjun Jang, Deuk Sin Kwon, and Thomas Lukasiewicz. 2022. Becel: Benchmark for consistency evaluation of language models. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 3680–3696.

Di Jin, Zhijing Jin, Joey Tianyi Zhou, and Peter Szolovits. 2020. Is bert really robust? a strong baseline for natural language attack on text classification and entailment. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 8018–8025.

Mark T Keane and Eoin M Kenny. 2019. How case-based reasoning explains neural networks: A theoretical analysis of xai using post-hoc explanation-by-example from a survey of ann-cbr twin-systems. In *Case-Based Reasoning Research and Development: 27th International Conference, ICCBR 2019, Otzenhausen, Germany, September 8–12, 2019, Proceedings 27*, pages 155–171. Springer.

Eunji Kim, Siwon Kim, Minji Seo, and Sungroh Yoon. 2021. Xprotonet: Diagnosis in chest radiography with global and local explanations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 15719–15728.

Pang Wei Koh, Thao Nguyen, Yew Siang Tang, Stephen Mussmann, Emma Pierson, Been Kim, and Percy Liang. 2020. Concept bottleneck models. In *International conference on machine learning*, pages 5338–5348. PMLR.

Alexey Kurakin, Ian Goodfellow, and Samy Bengio. 2017a. Adversarial examples in the physical world.

Alexey Kurakin, Ian Goodfellow, and Samy Bengio. 2017b. Adversarial machine learning at scale.

Jens Lehmann, Robert Isele, Max Jakob, Anja Jentzsch, Dimitris Kontokostas, Pablo N Mendes, Sebastian Hellmann, Mohamed Morsey, Patrick Van Kleef, Sören Auer, et al. 2015. Dbpedia–a large-scale, multilingual knowledge base extracted from wikipedia. *Semantic web*, 6(2):167–195.

Qi Lei, Lingfei Wu, Pin-Yu Chen, Alex Dimakis, Inderjit S Dhillon, and Michael J Witbrock. 2019. Discrete adversarial attacks and submodular optimization with applications to text classification. *Proceedings of Machine Learning and Systems*, 1:146–165.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2019. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension.

Ang Li, Fangyuan Zhang, Shuangjiao Li, Tianhua Chen, Pan Su, and Hongtao Wang. 2023. Efficiently generating sentence-level textual adversarial examples with seq2seq stacked auto-encoder. *Expert Systems with Applications*, 213:119170.

Jinfeng Li, Shouling Ji, Tianyu Du, Bo Li, and Ting Wang. 2018a. Textbugger: Generating adversarial text against real-world applications. *arXiv preprint arXiv:1812.05271*.

Oscar Li, Hao Liu, Chaofan Chen, and Cynthia Rudin. 2018b. Deep learning for case-based reasoning through prototypes: A neural network that explains its predictions. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence and Thirtieth Innovative Applications of Artificial Intelligence Conference and Eighth AAAI Symposium on Educational Advances in Artificial Intelligence*, AAAI'18/IAAI'18/EAAI'18. AAAI Press.

Tal Linzen. 2020. How can we accelerate progress towards human-like linguistic generalization? *arXiv preprint arXiv:2005.00955*.

Pengfei Liu, Jinlan Fu, Yanghua Xiao, Weizhe Yuan, Shuaichen Chang, Junqi Dai, Yixin Liu, Zihuiwen Ye, Zi-Yi Dou, and Graham Neubig. 2021. ExplainaBoard: An Explainable Leaderboard for NLP. In *Annual Meeting of the Association for Computational Linguistics (ACL), System Demonstrations*.

Xiaodong Liu, Hao Cheng, Pengcheng He, Weizhu Chen, Yu Wang, Hoifung Poon, and Jianfeng Gao. 2020. Adversarial training for large neural language models. *arXiv preprint arXiv:2004.08994*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. 2011. Learning word vectors for sentiment analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 142–150, Portland, Oregon, USA. Association for Computational Linguistics.

Shiao Meng, Xuming Hu, Aiwei Liu, Shu'ang Li, Fukun Ma, Yawen Yang, and Lijie Wen. 2023. RAPL: A Relation-Aware Prototype Learning Approach for Few-Shot Document-Level Relation Extraction. *arXiv preprint arXiv:2310.15743*.

Pascal Mettes, Elise Van der Pol, and Cees Snoek. 2019. Hyperspherical prototype networks. *Advances in neural information processing systems*, 32.

Yao Ming, Panpan Xu, Huamin Qu, and Liu Ren. 2019. Interpretable and steerable sequence learning via prototypes. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM.

Milad Moradi and Matthias Samwald. 2021. Evaluating the robustness of neural language models to input perturbations.

John X Morris, Eli Lifland, Jack Lanchantin, Yangfeng Ji, and Yanjun Qi. 2020a. Reevaluating adversarial examples in natural language. *arXiv preprint arXiv:2004.14174*.

John X. Morris, Eli Lifland, Jin Yong Yoo, Jake Grigsby, Di Jin, and Yanjun Qi. 2020b. Textattack: A framework for adversarial attacks, data augmentation, and adversarial training in nlp.

Meike Nauta, Annemarie Jutte, Jesper Provoost, and Christin Seifert. 2021a. This looks like that, because ... explaining prototypes for interpretable image recognition. In *Communications in Computer and Information Science*, pages 441–456. Springer International Publishing.

Meike Nauta, Ron van Bree, and Christin Seifert. 2021b. Neural prototype trees for interpretable fine-grained image recognition. In *Proceedings of the 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition – CVPR 2021*, pages 14933–14943, Nashville, TN, USA. IEEE.

OpenAI. 2022. Chatgpt. https://openai.com/blog/chatgpt. Accessed: April 30, 2023.

Frederik Pahde, Mihai Puscas, Tassilo Klein, and Moin Nabi. 2021. Multimodal prototypical networks for few-shot learning. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pages 2644–2653.

Nicolas Papernot and Patrick McDaniel. 2018. Deep k-nearest neighbors: Towards confident, interpretable and robust deep learning. *arXiv preprint arXiv:1803.04765*.

Kamil Pluciński, Mateusz Lango, and Jerzy Stefanowski. 2021. Prototypical convolutional neural network for a phrase-based explanation of sentiment classification. In *Machine Learning and Principles and Practice of Knowledge Discovery in Databases*, pages 457–472, Cham. Springer International Publishing.

Shuhuai Ren, Yihe Deng, Kun He, and Wanxiang Che. 2019. Generating natural language adversarial examples through probability weighted word saliency. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1085–1097, Florence, Italy. Association for Computational Linguistics.

Eleanor H. Rosch. 1973. Natural categories. *Cognitive Psychology*, 4(3):328–350.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*.

Sascha Saralajew, Lars Holdijk, and Thomas Villmann. 2020. Fast adversarial robustness certification of nearest prototype classifiers for arbitrary seminorms. In *Advances in Neural Information Processing Systems*, pages 13635–13650. Curran Associates, Inc.

Paul-Edouard Sarlin, Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. 2020. Superglue: Learning feature matching with graph neural networks.

Freda Shi, Xinyun Chen, Kanishka Misra, Nathan Scales, David Dohan, Ed Chi, Nathanael Schärli, and Denny Zhou. 2023. Large language models can be easily distracted by irrelevant context.

Chenglei Si, Zhengyan Zhang, Fanchao Qi, Zhiyuan Liu, Yasheng Wang, Qun Liu, and Maosong Sun. 2021. Better robustness by more coverage: Adversarial and mixup data augmentation for robust finetuning. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 1569–1576.

Dylan Slack, Satyapriya Krishna, Himabindu Lakkaraju, and Sameer Singh. 2022. Talktomodel: Explaining machine learning models with interactive natural language conversations.

Jake Snell, Kevin Swersky, and Richard Zemel. 2017. Prototypical networks for few-shot learning. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Eduardo Soares, Plamen Angelov, and Neeraj Suri. 2022. Similarity-based deep neural network to detect imperceptible adversarial attacks. In *2022 IEEE Symposium Series on Computational Intelligence (SSCI)*, pages 1028–1035.

Zhivar Sourati, Vishnu Priya Prasanna Venkatesh, Darshan Deshpande, Himanshu Rawlani, Filip Ilievski, Hông-Ân Sandlin, and Alain Mermoud. 2023. Robust and explainable identification of logical fallacies in natural language arguments. *Knowledge-Based Systems*, 266:110418.

Iulia Turc, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Well-read students learn better: On the importance of pre-training compact models.

Betty van Aken, Jens-Michalis Papaioannou, Marcel G. Naik, Georgios Eleftheriadis, Wolfgang Nejdl, Felix A. Gers, and Alexander Löser. 2022. This patient looks like that patient: Prototypical networks for interpretable diagnosis prediction from clinical text.

Václav Vorácek and Matthias Hein. 2022. Provably adversarially robust nearest prototype classifiers. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of the Proceedings of Machine Learning Research, pages 22361–22383, Baltimore, MD, USA. PMLR.

Kiri Wagstaff. 2012. Machine learning that matters. *arXiv preprint arXiv:1206.4656*.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019. Glue: A multi-task benchmark and analysis platform for natural language understanding.

Boxin Wang, Chejian Xu, Shuohang Wang, Zhe Gan, Yu Cheng, Jianfeng Gao, Ahmed Hassan Awadallah, and Bo Li. 2022a. Adversarial glue: A multi-task benchmark for robustness evaluation of language models.

Jindong Wang, Xixu Hu, Wenxin Hou, Hao Chen, Runkai Zheng, Yidong Wang, Linyi Yang, Haojun Huang, Wei Ye, Xiubo Geng, Binxin Jiao, Yue Zhang, and Xing Xie. 2023. On the robustness of chatgpt: An adversarial and out-of-distribution perspective.

Shiqi Wang, Zheng Li, Haifeng Qian, Chenghao Yang, Zijian Wang, Mingyue Shang, Varun Kumar, Samson Tan, Baishakhi Ray, Parminder Bhatia, Ramesh Nallapati, Murali Krishna Ramanathan, Dan Roth, and Bing Xiang. 2022b. Recode: Robustness evaluation of code generation models.

Xuezhi Wang, Haohan Wang, and Diyi Yang. 2022c. Measure and improve robustness in nlp models: A survey.

Jing Wu, Mingyi Zhou, Ce Zhu, Yipeng Liu, Mehrtash Harandi, and Li Li. 2021. Performance evaluation of adversarial attacks: Discrepancies and solutions. *arXiv preprint arXiv:2104.11103*.

Hong-Ming Yang, Xu-Yao Zhang, Fei Yin, and Cheng-Lin Liu. 2018. Robust classification with convolutional prototype learning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3474–3482.

Jin Yong Yoo, John X. Morris, Eli Lifland, and Yanjun Qi. 2020. Searching for a search method: Benchmarking search algorithms for generating nlp adversarial examples.

Wei Emma Zhang, Quan Z Sheng, Ahoud Alhazmi, and Chenliang Li. 2020. Adversarial attacks on deep-learning models in natural language processing: A survey. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 11(3):1–41.

Haiyan Zhao, Hanjie Chen, Fan Yang, Ninghao Liu, Huiqi Deng, Hengyi Cai, Shuaiqiang Wang, Dawei Yin, and Mengnan Du. 2023. Explainability for large language models: A survey. *arXiv preprint arXiv:2309.01029*.

Pei Zhou, Rahul Khanna, Seyeon Lee, Bill Yuchen Lin, Daniel Ho, Jay Pujara, and Xiang Ren. 2020. Rica: Evaluating robust inference capabilities based on commonsense axioms. *arXiv preprint arXiv:2005.00782*.

Julia El Zini and Mariette Awad. 2022. On the explainability of natural language processing deep models. *ACM Comput. Surv.*, 55(5).

Barret Zoph, Irwan Bello, Sameer Kumar, Nan Du, Yanping Huang, Jeff Dean, Noam Shazeer, and William Fedus. 2022. St-moe: Designing stable and transferable sparse expert models.

# A Dataset Details

The statistics of the datasets we used in this study to test the robustness of PBNs against perturbations are demonstrated in Table 5. We present both statistics about the original dataset and statistics and details about the number of perturbations that we have gathered on each dataset with different attack strategies. All the original datasets we use in this study are gathered by other researchers and have been made public by them, mentioning non-commercial use, which aligns with how we use these datasets. We have included information on their descriptions and how they were gathered:

**IMDB.** This dataset is compiled from a set of 50000 reviews sourced from IMDB in English, limiting each movie to a maximum of 30 reviews. It has maintained an equal count of positive and negative reviews, ensuring a 50% accuracy through random guessing. To align with prior research on polarity classification, the authors specifically focus on highly polarized reviews. A review is considered negative if it scores $\leq 4$ out of 10 and positive if it scores $\geq 7$ out of 10. Neutral reviews are excluded from this dataset. Authors have made the dataset publicly available, and you can find more information about this dataset at https://ai.stanford.edu/~amaas/data/sentiment/.

**AG_News.** This dataset comprises over 1 million English news articles sourced from 2000+ news outlets over a span of more than a year by ComeToMyHead, an academic news search engine operational since July 2004. Provided by the academic community, this dataset aids research in data mining, information retrieval, data compression, data streaming, and non-commercial activities. This news topic classification dataset features four classes: world, sports, business, and science. The details about the intended use and access conditions are provided at http://www.di.unipi.it/~gulli/AG_corpus_of_news_articles.html.

**DBPedia.** DBPedia[5] seeks to extract organized information from Wikipedia's vast content. The gathered subset of data we used offers hierarchical categories for 342782 Wikipedia articles. These classes are distributed across three levels, comprising 9, 70, and 219 classes, respectively. We used the version that has nine classes: Agent, Work, Place, Species, UnitOfWork, Event, SportsSeason, Device, and TopicalConcept. Although the articles are in English, specific names (e.g., the name of a place or person) can be non-English. Find more information about this dataset at https://huggingface.co/datasets/DeveloperOats/DBPedia_Classes.

**AdvGLUE.** Adversarial GLUE (AdvGLUE) (Wang et al., 2022a) introduces a multi-task English benchmark designed to investigate and assess the vulnerabilities of modern large-scale language models against various adversarial attacks. It encompasses five corpora, including SST-2 sentiment classification, QQP paraphrase test dataset, and QNLI, RTE, and MNLI, all of which are natural language inference datasets. To assess robustness, perturbations are applied to these datasets through both automated and human-evaluated methods, spanning word-level, sentence-level, and human-crafted examples. Our focus primarily centers on SST-2 due to its alignment with the other covered datasets in our study and its classification nature. This dataset has been made public by the authors and is released with CC BY-SA 4.0 license.

# B Implementation Details

## B.1 Experimental Environment

For all the experiments that involved training or evaluating PBNs or vanilla LMs, we used three GPU NVIDIA RTX A5000 devices with Python v3.9.16 and CUDA v11.6, and each experiment took between 10 minutes to 2 hours, depending on the dataset and model used. All Transformer models were trained using the Transformers package v4.30.2 and Torch package v2.0.1+cu117. We used TextAttack v0.3.10 (Morris et al., 2020b) for implementing the employed attack strategies and perturbations.

## B.2 Training Details

All prototypes are initialized randomly for a fair comparison, and only the last layer of LM backbones are trainable. The prototypes are trained without being constrained to a certain class from the beginning, and their corresponding class can be identified after training. The transformation from distances to class logits is done through a simple fully connected layer without intercept to avoid introducing additional complexity and keep the prediction interpretable through prototype distances. Both the backbone of PBNs and their vanilla counterparts leveraged the same LM and were fine-tuned separately to show the difference that is only attributed to the models' architecture. Focusing on the BERT-based PBN for evaluation, since BERT-base is one of the models from which we extract static perturbations by directly attacking it, to ensure generalization of the experiments on different backbones in the evaluation step, we use BERT-Medium (Turc et al., 2019) as the backbone for BERT-based PBN and its vanilla counterpart.

[5] https://www.dbpedia.org/

| Dataset | #Classes | #Tokens | #Train | #Val | #Test | BAE | DWB | PWWS | TB | TF | Other |
|---|---|---|---|---|---|---|---|---|---|---|---|
| IMDB | 2 | 234 | 22,500 | 2,500 | 25,000 | 1784 | 1584 | 2816 | 2408 | 2880 | - |
| AG_News | 4 | 103 | 112,400 | 7,600 | 7,600 | 663 | 1287 | 1533 | 1383 | 1893 | - |
| DBPedia | 9 | 38 | 240,942 | 36,003 | 60,794 | 1041 | 1143 | 1401 | 1281 | 1836 | - |
| SST-2 | 2 | 14 | 67,349 | 872 | 1,821 | - | - | - | - | - | 148 |

Table 5: Dataset statistics: number of classes, the average number of tokens, and size of the perturbed datasets under BAE, DeepWordBug (DWB), PWWS, TextBugger (TB), TextFooler (TF), obtained. SST-2 subset comes from the AdvGlue benchmark (Wang et al., 2022a) after removing the human-generated instances that do not belong to either category of perturbation classes.

For all the datasets, the training split, validation split, and test splits were used from `https://huggingface.co/`. During training on the IMDB, SST-2, and DBPedia datasets, the batch size was set to 64. This number was 256 on the AG_News dataset. All the models were trained with the number of epochs adjusted according to an early stopping module with patience of 4 and a threshold value of 0.01 for change in accuracy.

All the Transformer models were fine-tuned on top of a pre-trained model gathered from `https://huggingface.co/`. Details of the models used in our experiments are presented in the following:

- Electra (Clark et al., 2020): google/electra-base-discriminator;
- BART (Lewis et al., 2019): ModelTC/bart-base-mnli, facebook/bart-base, facebook/bart-large-mnli;
- BERT (Devlin et al., 2018): prajjwal1/bert-medium.

Furthermore, the models that were used in the process of gathering static perturbations were also pre-trained Transformer models gathered from `https://huggingface.co/`. Find the details of models used categorized by the dataset below:

- IMDB: textattack/bert-base-uncased-imdb, textattack/distilbert-base-uncased-imdb, textattack/roberta-base-imdb;
- AG_News: textattack/bert-base-uncased-ag-news, andi611/distilbert-base-uncased-ner-agnews, textattack/roberta-base-ag-news;
- DBPedia: dbpedia_bert-base-uncased, dbpedia_distilbert-base-uncased, dbpedia_roberta-base.

Since we could not find models from TextAttack (Morris et al., 2020b) library that were fine-tuned on DBPedia, the models that are presented above were fine-tuned by us on that dataset as well and then used as the target model.

### B.3 GPT4o and Llama3 Baseline

We used GPT4o and Llama3 (AI@Meta, 2024) as baselines in our experiments to compare its performance on original and perturbed examples with PBN and their vanilla models. In this section, we present the prompts that we gave to these models to generate the baseline responses and the reported performance in Table 2. We used the following prompts for the four different datasets:

IMDB: *Identify the binary sentiment of the following text:* **[text]**. *Strictly output only "negative" or "positive" according to the sentiment and nothing else. Assistant:*

AG_News: *Categorize the following news strictly into only one of the following classes: world, sports, business, and science. Ensure that you output only the category name and nothing else. Text:* **[text]**. *Assistant:*

DBPedia: *Categorize the following text article strictly into only one taxonomic category from the following list: Agent, Work, Place, Species, UnitOfWork, Event, SportsSeason, Device, and TopicalConcept. Ensure that you output only the category name and nothing else. Text:* **[text]**. *Assistant:*

SST-2: *Identify the binary sentiment of the following text:* **[text]**. *Strictly output only "negative" or "positive" according to the sentiment and nothing else. Assistant:*

## C Additional Experiments

### C.1 Robustness of PBNs Against Paraphrased-Based Perturbations

Comparison between PBNs and vanilla LMs on the original and paraphrased version of texts from AG_News, DBPedia, and IMDB datasets that GPT3.5 generated are shown in Table 6, which

| | AG_News | | DBPedia | | IMDB | |
|---|---|---|---|---|---|---|
| | Orig | Adv | Orig | Adv | Orig | Adv |
| BART | 93.7 | 92.6 | 91.2 | 91.3 | 97.5 | 96.0 |
| + PBN | 93.2 | 93.8 | 92.0 | 91.6 | 97.2 | 97.0 |
| BERT | 92.5 | 91.0 | 90.8 | 90.5 | 95.5 | 94.2 |
| + PBN | 92.8 | 91.2 | 90.3 | 90.8 | 95.2 | 95.0 |
| ELEC. | 93.0 | 92.1 | 90.5 | 90.0 | 96.0 | 94.5 |
| + PBN | 93.5 | 91.8 | 90.8 | 89.7 | 95.8 | 95.0 |

Table 6: Comparison between PBNs and vanilla LMs on the original and paraphrased version of texts from AG_News, DBPedia, and IMDB datasets that GPT3.5 generated.

illustrated that both PBNs and vanilla LMs are robust to such perturbations.

### C.2 Robustness of PBNs' w.r.t. Average Percentage of Words Perturbed

The Comparison of PBNs and vanilla LMs' robustness with respect to the Average Percentage of Words Perturbed (APWP) under different adversarial settings, different datasets, and perturbation strategies is shown in Table 7. We observed that while using the best hyperparameters, PBNs are more robust than vanilla LMs in the majority of the cases, this superiority is less salient when averaging over all hyperparameters involved in PBNs' training, which entails how PBNs' robustness is sensitive to hyperparameters.

### C.3 Robustness of PBNs' Averaged over Hyperparameters

The comparison of PBNs and vanilla LMs under different adversarial settings, on different datasets, and different attacking strategies, averaged over all hyperparameters of PBNs, is shown in Table 8. Comparing the observed trends with the same trends when using the best hyperparameters for PBNs, our results suggested that PBNs' robustness is sensitive to hyperparameters that are involved in their training.

### C.4 Effect of Distance Function on Robustness

Figure 6, Figure 4, and Figure 5 illustrate the robustness of PBNs compared to vanilla LMs, using different distance functions, showing that PBNs' robustness is not sensitive to this hyperparameter.

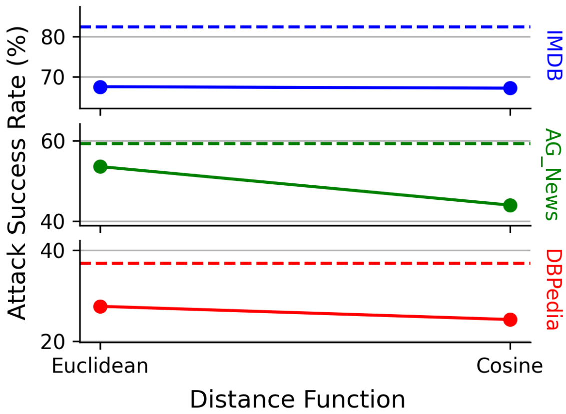


Figure 4: Attack Success Rate (ASR %) of PBNs with different distance functions and other hyperparameters set to their best values and averaged across other possible variables (e.g., backbone and attack type). The dotted line represents the ASR for the vanilla LMs.

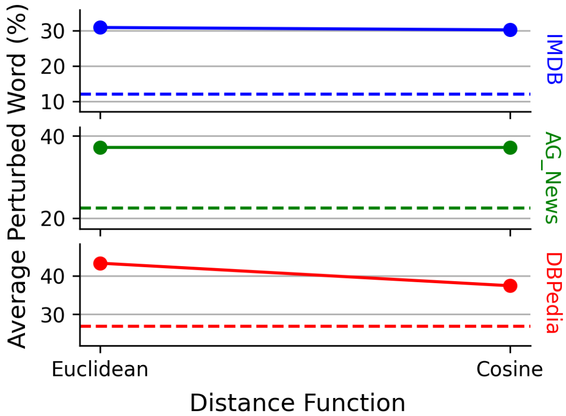


Figure 5: Average Percentage of Words Perturbed (APWP) of PBNs with different distance functions and other hyperparameters set to their best values and averaged across other possible variables (e.g., backbone and attack type). The dotted line represents the ASR for the vanilla LMs.

Using the best hyperparameters

| | AG_News | | | | | DBPedia | | | | | IMDB | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | BAE | DWB | PWWS | TB | TF | BAE | DWB | PWWS | TB | TF | BAE | DWB | PWWS | TB | TF |
| BART | 8.7 | **26.9** | 20.8 | 35.7 | 25.0 | 9.1 | **27.3** | **16.9** | **50.1** | **26.2** | 4.1 | 6.4 | 4.2 | 33.3 | 5.9 |
| + PBN | **9.0** | 24.8 | **22.2** | **37.7** | **27.6** | **10.1** | 17.1 | 15.9 | 43.3 | 26.0 | **4.7** | **6.6** | **8.1** | **33.4** | **13.6** |
| BERT | 7.4 | **26.8** | 21.6 | 37.4 | 24.1 | 9.7 | 27.9 | 19.4 | **53.8** | 28.8 | 4.0 | 5.7 | 4.4 | 30.1 | 5.0 |
| + PBN | **7.7** | 26.6 | **24.1** | **37.7** | **28.8** | **10.9** | **27.9** | **22.4** | 50.0 | **30.6** | **4.6** | **6.7** | **9.3** | **35.9** | **15.4** |
| ELEC. | **8.2** | **23.7** | 17.5 | **32.7** | 20.8 | 10.9 | 24.6 | 17.7 | **58.0** | 22.9 | 5.4 | 8.1 | 8.8 | **44.7** | 11.2 |
| + PBN | 8.1 | 21.2 | **18.9** | 31.8 | **24.0** | **11.9** | **25.1** | **19.4** | 48.5 | **26.8** | **5.6** | **8.4** | **13.3** | 38.6 | **18.5** |

Averaged over all hyperparameters

| | AG_News | | | | | DBPedia | | | | | IMDB | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | BAE | DWB | PWWS | TB | TF | BAE | DWB | PWWS | TB | TF | BAE | DWB | PWWS | TB | TF |
| BART | **8.7** | **26.9** | **20.8** | **35.7** | 25.0 | 9.1 | **27.3** | **16.9** | **50.1** | **26.2** | 4.1 | **6.4** | 4.2 | **33.3** | 5.9 |
| + PBN | 8.3 | 19.3 | 20.5 | 32.6 | **25.2** | **9.7** | 17.1 | 15.9 | 40.4 | 24.7 | **4.4** | 6.1 | **6.5** | 29.5 | **10.1** |
| BERT | **7.4** | **26.8** | 21.6 | **37.4** | 24.1 | **9.7** | **27.9** | **19.4** | **53.8** | **28.8** | 4.0 | **5.7** | 4.4 | **30.1** | 5.0 |
| + PBN | 7.2 | 24.1 | **21.9** | 35.0 | **25.9** | 9.5 | 24.1 | 19.3 | 43.1 | 27.6 | **4.1** | 5.5 | **5.0** | 27.3 | **7.1** |
| ELEC. | **8.2** | **23.7** | **17.5** | **32.7** | **20.8** | **10.9** | **24.6** | **17.7** | **58.0** | 22.9 | 5.4 | **8.1** | 8.8 | **44.7** | 11.2 |
| + PBN | 7.7 | 15.3 | 16.1 | 26.1 | 20.1 | 10.2 | 18.2 | 16.6 | 40.2 | **23.7** | 5.4 | 6.7 | **10.1** | 31.3 | **13.6** |

Table 7: Comparison of PBNs and vanilla LMs' robustness with respect to Average Percentage of Words Perturbed (APWP) under targeted adversarial attack perturbations, both using the best hyperparameters and averaged over all hyperparameters for PBNs, on IMBD, AG_News, and DBPeida datasets, under BAE, DeepWordBug (DWB), PWWS, TextBugger (TB), TextFooler (TF). The highest APWP showing the superior model for each architecture is **boldfaced**.

Targeted Attacks; Attack Success Rate (ASR %) reported

| | AG_News | | | | | DBPedia | | | | | IMDB | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | BAE | DWB | PWWS | TB | TF | BAE | DWB | PWWS | TB | TF | BAE | DWB | PWWS | TB | TF |
| BART | 14.8 | 53.2 | 53.6 | 31.8 | 76.5 | 18.9 | 28.3 | 43.1 | 21.1 | 71.9 | 74.1 | 74.7 | 99.3 | 78.5 | 100.0 |
| + PBN | **14.8** | **40.4** | **50.7** | **29.8** | **76.2** | **17.0** | **14.7** | **28.7** | **12.7** | **49.4** | **55.5** | **49.2** | **86.2** | **49.7** | **88.5** |
| BERT | 17.0 | 78.0 | 69.8 | 45.7 | 88.8 | 13.9 | 24.8 | 31.6 | 22.0 | 61.3 | 82.5 | 79.7 | 99.9 | 83.9 | 99.9 |
| + PBN | **14.0** | **64.7** | **57.0** | **39.3** | **82.1** | **13.5** | **23.4** | **27.6** | **19.6** | **51.3** | **68.4** | **61.8** | **91.3** | **74.0** | **92.4** |
| ELEC. | 24.8 | 89.5 | 69.1 | 87.8 | 87.9 | 14.5 | 42.8 | 45.6 | 42.3 | 75.3 | 52.5 | 49.2 | 95.3 | 67.8 | 99.3 |
| + PBN | **18.5** | **50.4** | **55.7** | **63.6** | **80.0** | **12.6** | **19.4** | **26.1** | **27.1** | **46.5** | **41.0** | **35.9** | **77.7** | **55.6** | **86.2** |

Static Attacks; Accuracy (%) reported

| | AG_News | | | | | DBPedia | | | | | IMDB | | | | | SST2 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | BAE | DWB | PWWS | TB | TF | BAE | DWB | PWWS | TB | TF | BAE | DWB | PWWS | TB | TF | GLUE |
| BART | 53.2 | 76.7 | 83.2 | 77.5 | 85.8 | 55.5 | 68.6 | 58.4 | 72.5 | 71.3 | 74.1 | 80.5 | 83.6 | 85.8 | 87.6 | 29.8 |
| + PBN | 50.4 | 68.3 | 75.7 | 70.5 | 79.6 | 56.4 | 65.8 | 58.7 | 70.9 | 69.5 | 69.2 | 78.7 | 79.7 | 81.9 | 78.3 | **37.6** |
| + Aug. | **71.7** | **78.4** | **85.5** | **77.6** | **90.1** | **84.0** | **79.6** | **89.7** | **88.8** | **94.0** | **85.7** | **86.7** | **92.9** | **89.9** | **96.5** | - |
| BERT | 47.8 | 64.0 | 75.9 | 69.4 | 80.7 | 62.3 | 61.4 | 75.4 | 78.4 | 82.0 | 75.1 | 77.1 | 85.0 | 83.4 | 85.9 | 42.0 |
| + PBN | 49.5 | 66.2 | 76.4 | **71.3** | 82.3 | 63.5 | 61.1 | 73.9 | 76.9 | 79.4 | 71.0 | 73.9 | 81.1 | 80.2 | 79.2 | **47.1** |
| + Aug. | **58.3** | **71.6** | **78.3** | 71.2 | **85.4** | **75.5** | **70.9** | **84.1** | **90.5** | **91.0** | **83.2** | **77.6** | **91.7** | **90.8** | **89.2** | - |
| ELECTRA | 50.4 | **65.0** | 73.5 | 63.9 | 77.8 | 79.7 | 66.9 | 80.9 | 81.4 | 84.4 | **89.7** | 90.3 | 94.6 | 94.5 | 95.6 | 44.3 |
| + PBN | 52.7 | 63.9 | **73.7** | **67.1** | 77.8 | 73.4 | 64.1 | 73.0 | 76.4 | 80.6 | 80.6 | 79.4 | 79.9 | 80.2 | 86.8 | **56.4** |
| + Aug. | **55.0** | 59.5 | 71.7 | 61.6 | **79.5** | **86.2** | **73.8** | **88.1** | **84.5** | **92.8** | 89.4 | **93.7** | **95.3** | **94.9** | **95.8** | - |

Table 8: Comparison of PBNs and vanilla LMs (+ vanilla LMs with adversarial augmented training under static attack setting) under both targeted and static adversarial attack perturbations, averaged over all hyperparameters for PBNs, on IMBD, AG_News, DBPeida (+ SST-2 AdvGLUE under static attack setting) datasets, under BAE, DeepWordBug (DWB), PWWS, TextBugger (TB), TextFooler (TF). The highest accuracy and lowest ASR showing the superior model for each architecture is **boldfaced**, and the second best model is underlined for static attacks.

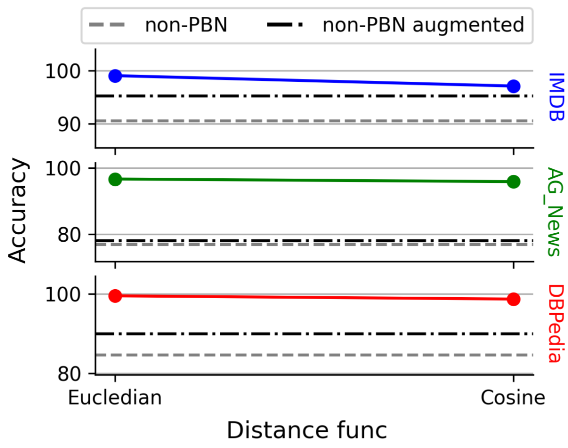


Figure 6: Accuracy of PBNs under static adversarial settings, with different distance functions, with other hyperparameters set to their best values and averaged across other possible variables (e.g., backbone and attack type). The dotted line represents the ASR for the vanilla LMs.

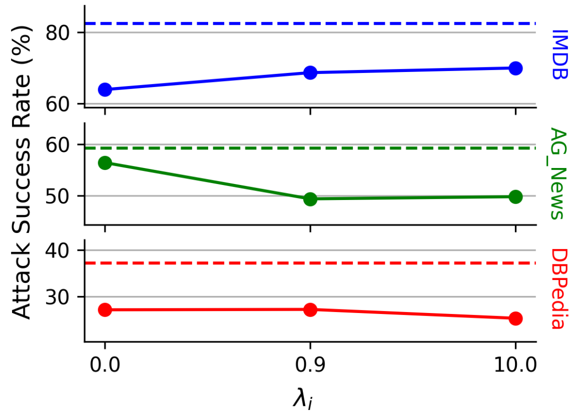


Figure 7: Attack Success Rate (ASR %) of PBNs with different $\lambda_i$ values adjusting the importance of interpretability of the prototypes in training, with other hyperparameters set to their best values, and averaged across other possible variables (e.g., backbone and attack type). The dotted line represents the ASR for the non-PBN model.

### C.5 Effect of Interpretability on Robustness

Figure 9, Figure 7, and Figure 8 illustrate the robustness of PBNs compared to vanilla LMs, using different values of $\lambda_i$ adjusting the importance of interpretability, showing that overall, PBNs' robustness is not sensitive to this hyperparameter.

### C.6 Effect of Clustering on Robustness

Figure 10, Figure 11 illustrate the robustness of PBNs compared to vanilla LMs, using different values of $\lambda_c$ adjusting the importance of clustering, that alongside the trends observed using ASR (see Figure 2), show that overall, PBNs' robustness degrades with tighter clustering in PBNs' training.

### C.7 Effect of Separation on Robustness

Figure 14, Figure 12, and Figure 13 illustrate the robustness of PBNs compared to vanilla LMs, using different values of $\lambda_s$ adjusting the importance of separability between prototypes, showing that overall, PBNs' robustness is not sensitive to this hyperparameter.

### C.8 Effect of Number of Prototypes on Robustness

Figure 15, Figure 16 illustrate the robustness of PBNs compared to vanilla LMs, using different numbers of prototypes, that alongside the trends observed using ASR (see Figure 3), show that overall, PBNs' robustness degrades with low number of prototypes as PBNs can capture lower number of semantic patterns in such conditions.

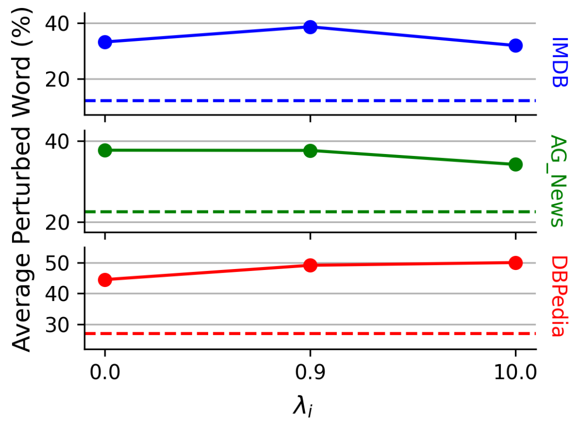


Figure 8: Average Percentage of Words Perturbed (APWP) of PBNs with different $\lambda_i$ values adjusting the importance of interpretability of the prototypes in training, with other hyperparameters set to their best values, and averaged across other possible variables (e.g., backbone and attack type). The dotted line represents the ASR for the non-PBN model.

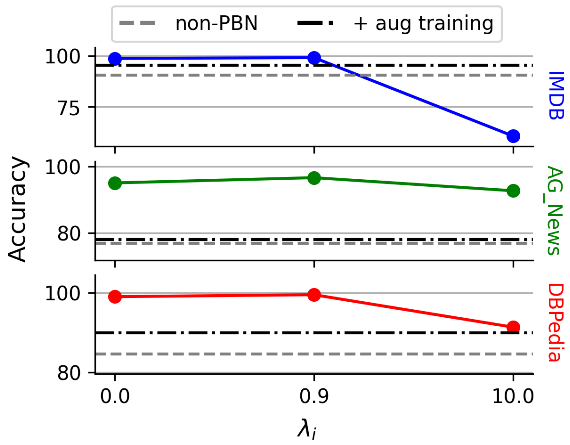


Figure 9: Accuracy of PBNs under static adversarial settings, with different $\lambda_i$ values adjusting the level of interpretability in PBNs, with other hyperparameters set to their best values and averaged across other possible variables (e.g., backbone and attack type). The dotted line represents the ASR for the vanilla LMs.

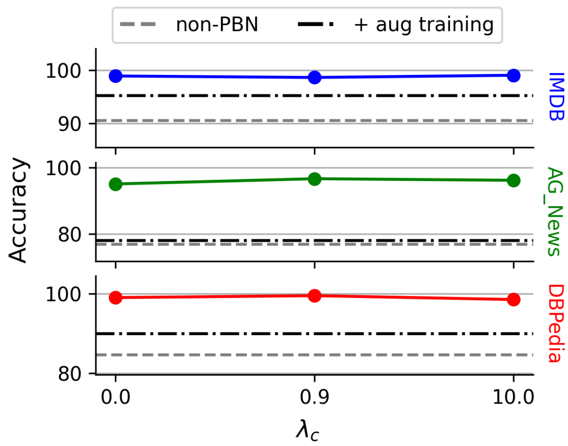


Figure 11: Accuracy of PBNs under static adversarial settings, with different $\lambda_c$ values adjusting the level of clustering in PBNs, with other hyperparameters set to their best values and averaged across other possible variables (e.g., backbone and attack type). The dotted line represents the ASR for the vanilla LMs.

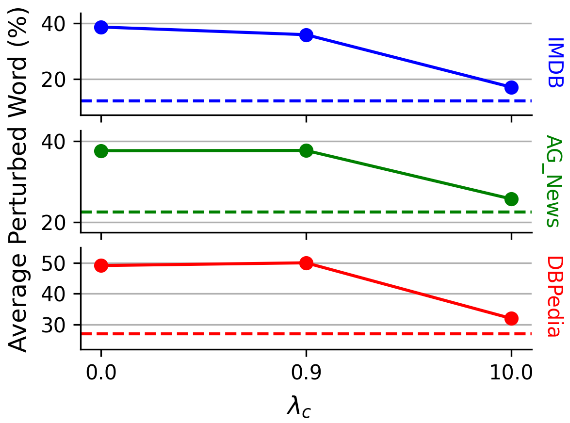


Figure 10: Average Percentage of Words Perturbed (APWP) of PBNs with different $\lambda_c$ values adjusting the importance of clustering of examples in PBNs, with other hyperparameters set to their best values, and averaged across other possible variables (e.g., backbone and attack type). The dotted line represents the ASR for the non-PBN model.

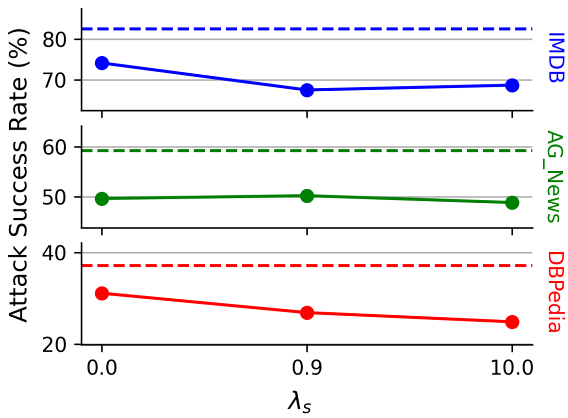


Figure 12: Attack Success Rate (ASR %) of PBNs with different $\lambda_s$ values adjusting the level of separation between the prototypes, with other hyperparameters set to their best values and averaged across other possible variables (e.g., backbone and attack type). The dotted line represents the ASR for the vanilla LMs.

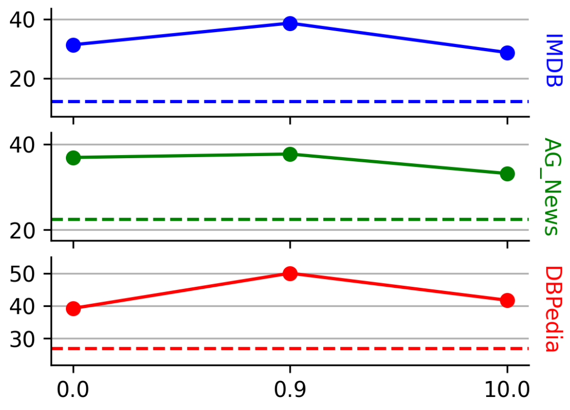


Figure 13: Average Percentage of Words Perturbed (APWP) of PBNs with different $\lambda_s$ values adjusting the level of separation between the prototypes, with other hyperparameters set to their best values and averaged across other possible variables (e.g., backbone and attack type). The dotted line represents the ASR for the vanilla LMs.

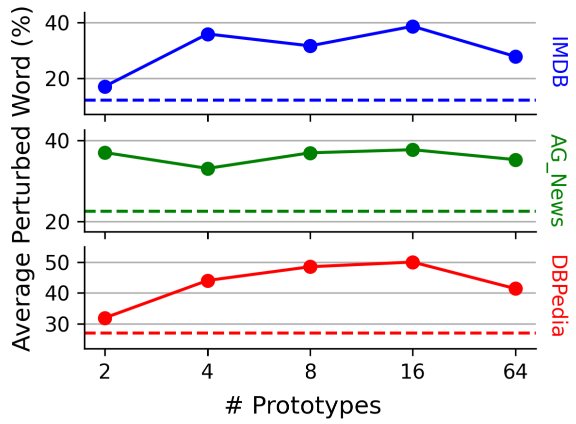


Figure 15: Average Percentage of Words Perturbed (APWP) of PBNs with different numbers of prototypes, with other hyperparameters set to their best values, and averaged across other possible variables (e.g., backbone and attack type). The dotted line represents the ASR for the non-PBN model.

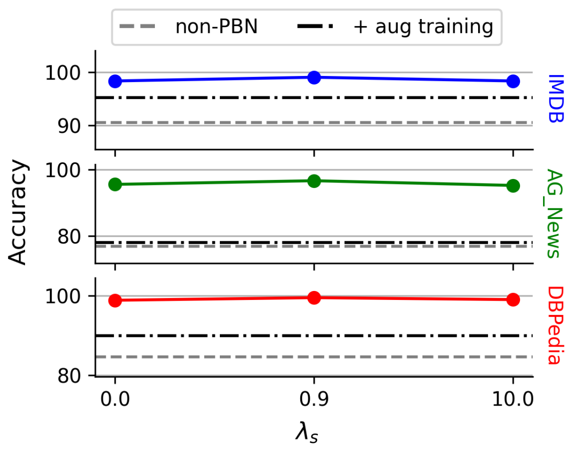


Figure 14: Accuracy of PBNs under static adversarial settings, with different $\lambda_s$ values adjusting the level of separation between the prototypes, with other hyperparameters set to their best values and averaged across other possible variables (e.g., backbone and attack type). The dotted line represents the ASR for the vanilla LMs.

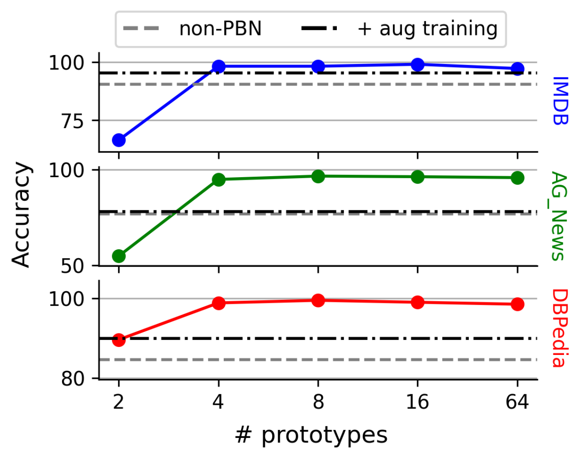


Figure 16: Accuracy of PBNs under static adversarial settings, with different numbers of prototypes, with other hyperparameters set to their best values and averaged across other possible variables (e.g., backbone and attack type). The dotted line represents the ASR for the vanilla LMs.