Achieving Verified Robustness to Symbol Substitutions via Interval Bound Propagation

Po-Sen Huang[†] **Robert Stanforth**[†§] **Johannes Welbl**[‡§] **Chris Dyer**[†]
Dani Yogatama[†] **Sven Gowal**[†] **Krishnamurthy Dvijotham**[†] **Pushmeet Kohli**[†]

[†]DeepMind [‡]University College London
{posenhuang, stanforth, cdyer, dyogatama, sgowal, dvij, pushmeet}@google.com
{j.welbl}@cs.ucl.ac.uk

Abstract

Neural networks are part of many contemporary NLP systems, yet their empirical successes come at the price of vulnerability to adversarial attacks. Previous work has used adversarial training and data augmentation to partially mitigate such brittleness, but these are unlikely to find worst-case adversaries due to the complexity of the search space arising from discrete text perturbations. In this work, we approach the problem from the opposite direction: to formally verify a system's robustness against a predefined class of adversarial attacks. We study text classification under synonym replacements or character flip perturbations. We propose modeling these input perturbations as a simplex and then using Interval Bound Propagation – a formal model verification method. We modify the conventional log-likelihood training objective to train models that can be efficiently verified, which would otherwise come with exponential search complexity. The resulting models show only little difference in terms of nominal accuracy, but have much improved verified accuracy under perturbations and come with an efficiently computable formal guarantee on worst case adversaries.

1 Introduction

Deep models have been shown to be vulnerable against adversarial input perturbations (Szegedy et al., 2013; Kurakin et al., 2016). Small, semantically invariant input alterations can lead to drastic changes in predictions, leading to poor performance on adversarially chosen samples. Recent work (Jia and Liang, 2017; Belinkov and Bisk, 2018; Ettinger et al., 2017) also exposed the vulnerabilities of neural NLP models, e.g. with small character perturbations (Ebrahimi et al., 2018) or paraphrases (Ribeiro et al., 2018; Iyyer et al., 2018). These adversarial attacks highlight often unintuitive model failure modes and present a challenge to deploying NLP models.

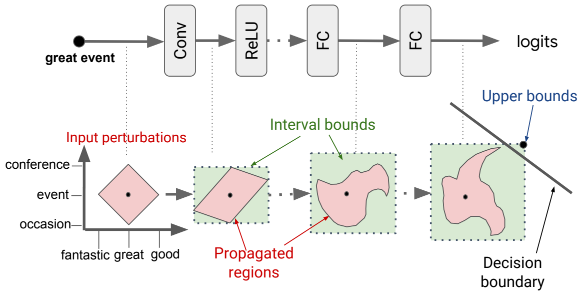


Figure 1: Illustration of verification with the input simplex and Interval Bound Propagation. From the left, input perturbations define the extreme points of a simplex (in red, projected to 2D here) around the statement "great event" that is propagated through a model. At each layer, this shape deforms itself, but can be bounded by axis-parallel bounding boxes, which are propagated similarly. Finally, in logit space, we can compute an upper bound on the worst-case specification violation (e.g., prediction changes).

Common attempts to mitigate the issue are adversarial training (Ebrahimi et al., 2018) and data augmentation (Belinkov and Bisk, 2018; Li et al., 2017), which lead to improved accuracy on adversarial examples. However, this might cause a false sense of security, as there is generally no guarantee that stronger adversaries could not circumvent defenses to find other successful attacks (Carlini and Wagner, 2017; Athalye et al., 2018; Uesato et al., 2018). Rather than continuing the race with adversaries, *formal verification* (Baier and Katoen, 2008; Barrett and Tinelli, 2018; Katz et al., 2017) offers a different approach: it aims at providing provable guarantees to a given model specification. In the case of adversarial robustness, such a specification can be formulated as prediction consistency under

[‡]Work done during an internship at DeepMind.
[§]Equal contribution.

Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing
and the 9th International Joint Conference on Natural Language Processing, pages 4083–4093,
Hong Kong, China, November 3–7, 2019. ©2019 Association for Computational Linguistics

any altered – but semantically invariant – input change.

In this paper, we study verifiable robustness, i.e., providing a certificate that for a given network and test input, no attack or perturbation under the specification can change predictions, using the example of text classification tasks, Stanford Sentiment Treebank (SST) (Socher et al., 2013) and AG News (Zhang et al., 2015). The *specification* against which we verify is that a text classification model should preserve its prediction under character (or synonym) substitutions in a character (or word) based model. We propose modeling these input perturbations as a simplex and then using Interval Bound Propagation (IBP) (Gowal et al., 2018; Mirman et al., 2018; Dvijotham et al., 2018) to compute worst case bounds on specification satisfaction, as illustrated in Figure 1. Since these bounds can be computed efficiently, we can furthermore derive an auxiliary objective for models to *become* verifiable. The resulting classifiers are efficiently verifiable and improve robustness on adversarial examples, while maintaining comparable performance in terms of nominal test accuracy.

The contributions of this paper are twofold:

- To the best of our knowledge, this paper is the first to introduce verification and verifiable training for neural networks in natural language processing (§3).
- Through a series of experiments (§4), we demonstrate (a) the effectiveness of modeling input perturbations as a simplex and using simplex bounds with IBP for training and testing, (b) the weakness of adversarial training under exhaustive verification, (c) the effects of perturbation space on the performance of different methods, and (d) the impact of using GloVe and counter-fitted embeddings on the IBP verification bounds.

2 Related Work

Adversarial Examples in NLP. Creating adversarial examples for NLP systems requires identifying semantically invariant text transformations to define an input perturbation space. In this paper, given our specification, we study word- and character-level *HotFlip* attacks (Ebrahimi et al., 2018) – which consist of character and synonym replacements – on text classification tasks. We compare our verifiable approach to other defenses including adversarial training (Goodfellow et al., 2014) and data augmentation (Li et al., 2017; Belinkov and Bisk, 2018). Note that some existing adversarial perturbations such as syntactically controlled paraphrasing (Iyyer et al., 2018), exploiting backtranslation systems (Ribeiro et al., 2018), or using targeted keyword attack (Cheng et al., 2018) are beyond the specification in this paper.

Formal Verification of Neural Networks. Formal verification provides a provable guarantee that models are consistent with a *specification* for all possible model inputs. Previous work can be categorised into complete methods that use Mixed-Integer Programming (MIP) (Bunel et al., 2017; Cheng et al., 2017) or Satisfiability Modulo Theory (SMT) (Katz et al., 2017; Carlini et al., 2017), and incomplete methods that solve a convex relaxation of the verification problem (Weng et al., 2018; Wong and Kolter, 2018; Wang et al., 2018). Complete methods perform exhaustive enumeration to find the worst case. Hence, complete methods are expensive and difficult to scale, though they provide exact robustness bounds. Incomplete methods provide loose robustness bounds, but can be more scalable and used inside the training loop for training models to be robust and verifiable (Raghunathan et al., 2018; Wong and Kolter, 2018; Dvijotham et al., 2018; Gowal et al., 2018). Our work is the first to extend incomplete verification to text classification, considering input perturbations on a simplex and minimising worst case bounds to adversarial attacks in text classification. We highlight that the verification of neural networks is an extremely challenging task, and that scaling complete and incomplete methods to large models remains an open challenge.

Representations of Combinatorial Spaces. Word lattices and hypergraphs are data structures that have often been used to efficiently represent and process exponentially large numbers of sentences without exhaustively enumerating them. Applications include automatic speech recognition (ASR) output rescoring (Liu et al., 2016), machine translation of ASR outputs (Bertoldi et al., 2007), paraphrase variants (Onishi et al., 2010), and word segmentation alternatives (Dyer et al., 2008). The specifications used to characterise the space of adversarial attacks are likewise a compact representation, and the algorithms discussed below operate on them without exhaustive enumeration.

3 Methodology

We assume a fixed initial vector representation $\mathbf{z}_0$ of a given input sentence z[1] (e.g. the concatenation of pretrained word embeddings) and use a neural network model, i.e. a series of differentiable transformations h_k:

$$\mathbf{z}_k = h_k(\mathbf{z}_{k-1}) \quad k = 1, \ldots, K \tag{1}$$

where $\mathbf{z}_k$ is the vector of activations in the k-th layer and the final output $\mathbf{z}_K$ consists of the logits for each class. Typically each h_k will be an affine transformation followed by an activation function (e.g. ReLU or sigmoid). The affine transformation can be a convolution (with the inputs and outputs having an implied 2D structure) of a vector of activations at each point in a sequence; in what follows these activations will be concatenated along the sequence to form a vector $\mathbf{z}_k$.

3.1 Verification

Verification is the process of examining whether the output of a model satisfies a given specification. Formally, this means establishing whether the following holds true for a given *normal* model input $\mathbf{x}_0$: $\forall \mathbf{z}_0 \in \mathcal{X}_{\text{in}}(\mathbf{x}_0) : \mathbf{z}_K \in \mathcal{X}_{\text{out}}$, where $\mathcal{X}_{\text{out}}$ characterizes a constraint on the outputs, and $\mathcal{X}_{\text{in}}(\mathbf{x}_0)$ defines a neighbourhood of $\mathbf{x}_0$ throughout which the constraint should be satisfied.

In our concrete use case, we consider a specification of robustness against adversarial attacks which are defined by bounded input perturbations (synonym flips up to δ words, or character flips up to δ characters) of the original sentence x. The attack space $\mathcal{X}_{\text{in}}(\mathbf{x}_0)$ is the set of vector representations (embeddings) of all such perturbed sentences. Denoting by $z_{K,y}$ the logit of label y, we formulate the output constraint that for all classes $y : z_{K,y_{\text{true}}} \geq z_{K,y}$. This specification establishes that the prediction of *all* perturbed sentences $\mathbf{z}_0 \in \mathcal{X}_{\text{in}}(\mathbf{x}_0)$ should correspond to the correct label y_{true}. This specification may equivalently be formulated as a set of half-space constraints on the logits: for each class y

$$(\mathbf{e}_y - \mathbf{e}_{y_{\text{true}}})^\top \mathbf{z}_K \leq 0 \quad \forall \mathbf{z}_0 \in \mathcal{X}_{\text{in}}(\mathbf{x}_0) \tag{2}$$

where $\mathbf{e}_i$ is a one-hot vector with 1 in the i-th position. In other words, the true class logit should be greater or equal than those for all other classes y, which means the prediction remains constant.

3.2 Verification as Optimisation

Verifying the specification in Eq. (2) can be done by solving the following constrained optimisation problem to find the input that would most strongly violate it:

$$\begin{aligned} \underset{\mathbf{z}_0 \in \mathcal{X}_{\text{in}}(\mathbf{x}_0)}{\text{maximize}} \quad & \mathbf{c}^\top \mathbf{z}_K \\ \text{subject to} \quad & \mathbf{z}_k = h_k(\mathbf{z}_{k-1}) \quad k = 1, \ldots, K \end{aligned} \tag{3}$$

where $\mathbf{c}$ is a vector with entries $c_y = 1$, $c_{y_{\text{true}}} = -1$ and 0 everywhere else. If the optimal value of the above optimisation problem is smaller than 0, then the specification in Eq. (2) is satisfied, otherwise a counter-example has been found. In our case, this corresponds to a successful adversarial attack.

3.3 Modeling Input Perturbations using Simplices

In the interests of computational feasibility, we will actually attempt to verify the specification on a larger, but more tractable input perturbation space $\bar{\mathcal{X}}_{\text{in}} \supseteq \mathcal{X}_{\text{in}}$. Any data point that is verifiable on this larger input perturbation space is necessarily verifiable with respect to the original specification.

In the domain of image classification, $\mathcal{X}_{\text{in}}$ is often modeled as an L_∞-ball, corresponding to input perturbations in which each pixel may be independently varied within a small interval. However, using such interval bounds is unsuitable for our situation of perturbations consisting of a small number δ of symbol substitutions. Although we could construct an axis-aligned bounding box $\bar{\mathcal{X}}_{\text{in}}$ in embedding space that encompasses all of $\mathcal{X}_{\text{in}}$, it would over-approximate the perturbation space to such an extent that it would contain perturbations where *all* symbols in the sentence have been substituted simultaneously.

To remedy this, we propose a tighter over-approximation in the form of a 'simplex' in embedding space. We first define this for the special case $\delta = 1$, in which $\mathcal{X}_{\text{in}} = \{\mathbf{x}_0\} \cup \{\mathbf{p}_0^{(m)} : 1 \leq m \leq M\}$ consists of the representations of all M sentences $p^{(m)}$ derived from x by performing a *single* synonym (or character) substitution, together with the unperturbed sentence x itself. In this case we define $\bar{\mathcal{X}}_{\text{in}}$ to be the convex hull $\mathcal{S}_1$ of $\mathcal{X}_{\text{in}}$. Note we are not considering contextual embeddings (Peters et al., 2018) here. Each 'vertex' $\mathbf{p}_0^{(m)}$ is a sequence of embedding vectors that differs from $\mathbf{x}_0$ at only one word (or character) position.

For a larger perturbation radius $\delta > 1$, the cardinality of $\mathcal{X}_{\text{in}}$ grows exponentially, so manipulating

[1] For brevity, we will refer both to the original symbol sequence and its corresponding vector representation with the same variable name, distinguishing them by styling.

its convex hull becomes infeasible. However, dilating $\mathcal{S}_1$ centered at $\mathbf{x}_0$, scaling it up by a factor of δ, yields a simplex $\mathcal{S}_\delta$ with $M+1$ vertices that contains $\mathcal{X}_{\text{in}}$.

More formally, we define a region in the input embedding space based on the M 'elementary' perturbations $\{\mathbf{p}_0^{(m)} : m = 1 \ldots M\}$ of $\mathbf{x}_0$ defined earlier for the $\delta = 1$ case. For perturbations of up to δ substitutions, we define $\tilde{\mathcal{X}}_{\text{in}}(\mathbf{x}_0)$ as the convex hull of $\{\mathbf{z}_0^{(m)} : m = 0 \ldots M\}$, where $\mathbf{z}_0^{(0)} = \mathbf{x}_0$ denotes the original (unperturbed) sentence representation and, for $m \geq 1$, $\mathbf{z}_0^{(m)} = \mathbf{x}_0 + \delta \cdot (\mathbf{p}_0^{(m)} - \mathbf{x}_0)$. The convex hull is an over-approximation of $\mathcal{X}_{\text{in}}(\mathbf{x}_0)$: it contains the representations of all sentences derived from x by performing up to δ substitutions at distinct word (or character) positions.

3.4 Interval Bound Propagation

To estimate the optimal value of the problem (3), given an input $\mathbf{z}_0$, we can propagate the upper/lower bounds on the activations $\mathbf{z}_k$ of each layer using interval arithmetic (Gowal et al., 2018).

We begin by computing interval bounds on the first layer's activations. Recall that any input $\mathbf{z}_0 \in \mathcal{X}_{\text{in}}$ will lie within the convex hull of certain vertices $\{\mathbf{z}_0^{(m)} : m = 0 \ldots M\}$. Then, assuming that the first layer h_1 is an affine transformation (e.g. linear or convolutional) followed by a monotonic activation function, the lower and upper bounds on the components $z_{1,i}$ of the first layer's activations $\mathbf{z}_1$ are as follows:

$$\begin{aligned} \underline{z}_{1,i}(\delta) &= \min_{m=0,\ldots,M} \mathbf{e}_i^\top h_1(\mathbf{z}_0^{(m)}) \\ \overline{z}_{1,i}(\delta) &= \max_{m=0,\ldots,M} \mathbf{e}_i^\top h_1(\mathbf{z}_0^{(m)}) \end{aligned} \tag{4}$$

Note that these bounds are efficient to compute (by passing each perturbation $\mathbf{z}_0^{(m)}$ through the first layer); in particular there is no need to compute the convex hull polytope.

For subsequent layers $k > 1$, the bounds on the components $z_{k,i}$ of $\mathbf{z}_k$ are:

$$\begin{aligned} \underline{z}_{k,i}(\delta) &= \min_{\underline{\mathbf{z}}_{k-1}(\delta) \leq \mathbf{z}_{k-1} \leq \overline{\mathbf{z}}_{k-1}(\delta)} \mathbf{e}_i^\top h_k(\mathbf{z}_{k-1}) \\ \overline{z}_{k,i}(\delta) &= \max_{\underline{\mathbf{z}}_{k-1}(\delta) \leq \mathbf{z}_{k-1} \leq \overline{\mathbf{z}}_{k-1}(\delta)} \mathbf{e}_i^\top h_k(\mathbf{z}_{k-1}) \end{aligned} \tag{5}$$

The above optimisation problems can be solved in closed form quickly for affine layers and monotonic activation functions, as illustrated in Gowal et al. (2018). Finally, the lower and upper bounds of the output logits $\mathbf{z}_K$ can be used to construct an upper bound on the solution of (3):

$$\underset{\underline{\mathbf{z}}_K(\delta) \leq \mathbf{z}_K \leq \overline{\mathbf{z}}_K(\delta)}{\text{maximize}} \quad \mathbf{c}^\top \mathbf{z}_K \tag{6}$$

Verifiable Training. The upper bound in (6) is fast to compute (only requires two forward passes for upper and lower bounds through the network). Hence, we can define a loss to optimise models such that the models are trained to be verifiable. Solving (6) is equivalent to finding the worst-case logit difference, and this is achieved when the logit of the true class is equal to its lower bound, and all other logits equal to their upper bounds. Concretely, for each class $y \neq y_{\text{true}}$: $\hat{\mathbf{z}}_{K,y}(\delta) = \overline{\mathbf{z}}_{K,y}(\delta)$, and $\hat{\mathbf{z}}_{K,y_{\text{true}}}(\delta) = \underline{\mathbf{z}}_{K,y_{\text{true}}}(\delta)$. The training loss can then be formulated as

$$L = \kappa \underbrace{\ell(\mathbf{z}_K, y_{\text{true}})}_{L_{\text{normal}}} + (1-\kappa) \underbrace{\ell(\hat{\mathbf{z}}_K(\delta), y_{\text{true}})}_{L_{\text{spec}}} \tag{7}$$

where ℓ is the cross-entropy loss, κ a hyperparameter that controls the relative weights between the classification loss L_{normal} and specification loss L_{spec}. If $\delta = 0$ then $\mathbf{z}_K = \hat{\mathbf{z}}_K(\delta)$, and thus L reduces to a standard classification loss. Empirically, we found that a curriculum-based training, starting with κ=1 and linearly decreasing to 0.25, is effective for verifiable training.

4 Experiments

We conduct verification experiments on two text classification datasets, Stanford Sentiment Treebank (SST) (Socher et al., 2013) and AG News corpus, processed in (Zhang et al., 2015). We focus on word-level and character-level experiments on SST and character-level experiments on AG News. Our specification is that models should preserve their prediction against up to δ synonym substitutions or character typos, respectively.

4.1 A Motivating Example

We provide an example from Table 2 to highlight different evaluation metrics and training methods. Given a sentence, "you ' ve seen them a million times .", that is predicted correctly (called *Nominal Accuracy*[2]) by a classification model, we want to further examine whether the model is robust against character typos (e.g., up to $\delta = 3$ typos) to

[2] We use the term "nominal accuracy" to indicate the accuracy under various adversarial perturbations is much lower.

this example. One way is to use some heuristic to search for a valid example with up to 3 typos that can change the prediction the most (called *adversarial example*). We evaluate the model using this adversarial example and report the performance (called *Adversarial Accuracy*). However, even if the adversarial example is predicted correctly, one can still ask: is the model truly robust against *any* typos (up to 3) to this example? In order to have a certificate that the prediction will not change under any $\delta = 3$ character typos (called *verifiably robust*), we could in theory exhaustively search over all possible cases and check whether any of the predictions is changed (called *Oracle Accuracy*). If we only allow a character to be replaced by another character nearby on the keyboard, already for this short sentence we need to exhaustively search over 2,951 possible perturbations. To avoid this combinatorial growth, we can instead model all possible perturbations using the proposed simplex bounds and propagate the bounds through IBP at the cost of two forward passes. Following Eq. (3), we can check whether this example can be verified to be robust against all perturbations (called *IBP-Verified Accuracy*).

There are also a number of ways in which the training procedure can be enhanced to improve the verifiable robustness of a model against typos to the sentence. The baseline is to train the model with the original/normal sentence directly (called *Normal Training*). Another way is to randomly sample typo sentences among the 2,951 possible perturbations and add these sentences to the training data (called *Data Augmentation Training*). Yet another way is to find, at each training iteration, the adversarial example among the (subset of) 2,951 possible perturbations that can change the prediction the most; we then use the adversarial example alongside the training example (called *Adversarial Training*). Finally, as simplex bounds with IBP is efficient to run, we can train a model to be verifiable by minimising Eq. (7) (called *Verifiable Training*).

4.2 Baselines

In this section we detail our baseline models.

Adversarial Training. In adversarial training (Madry et al., 2018; Goodfellow et al., 2014), the goal is to optimise the following saddle point problem:

$$\min_{\theta} \mathop{\mathbb{E}}_{(\mathbf{x}_0,y)} \left[\max_{\mathbf{z}_0 \in \mathcal{X}_{\text{in}}(\mathbf{x}_0)} \ell_\theta(\mathbf{z}_0, y) \right] \tag{8}$$

where the inner maximisation problem is to find an adversarial perturbation $\mathbf{z}_0 \in \mathcal{X}_{\text{in}}(\mathbf{x}_0)$ that can maximise the loss. In the inner maximisation problem, we use HotFlip (Ebrahimi et al., 2018) with perturbation budget δ to find the adversarial example. The outer minimisation problem aims to update model parameters such that the adversarial risk of (8) is minimised. To balance between the adversarial robustness and nominal accuracy, we use an interpolation weight of 0.5 between the original cross-entropy loss and the adversarial risk.

Data Augmentation Training. In the data augmentation setup, we randomly sample a valid perturbation z with perturbation budget δ from a normal input x, and minimise the cross-entropy loss given the perturbed sample z (denoted as data augmentation loss). We also set the interpolation weight between the data augmentation loss and the original normal cross-entropy loss to 0.5.

Normal Training. In normal training, we use the likelihood-based training using the normal training input x.

4.3 Setup

We use a shallow convolutional network with a small number of fully-connected layers for SST and AG News experiments. The detailed model architectures and hyperparameter details are introduced in the supplementary material. Although we use shallow models for ease of verifiable training, our nominal accuracy is on par with previous work such as Socher et al. (2013) (85.4%) and Shen et al. (2018) (84.3%) in SST and Zhang et al. (2015) (87.18%) in AG News. During training, we set the maximum number of perturbations to $\delta = 3$, and evaluate performance with the maximum number of perturbations from $\delta = 1$ to 6 at test time.

For word-level experiments, we construct the synonym pairs using the PPDB database (Ganitkevitch et al., 2013) and filter the synonyms with fine-grained part-of-speech tags using Spacy (Honnibal and Montani, 2017). For character-level experiments, we use synthetic keyboard typos from Belinkov and Bisk (2018), and allow one possible alteration per character that is adjacent to it on an American keyboard. The allowable input perturbation space is much larger than for word-level synonym substitutions, as shown in Table 3.

	SST-Char-Level			SST-Word-Level			AG-Char-Level		
Training	**Acc.**	**Adv. Acc.**	**Oracle**	**Acc.**	**Adv. Acc.**	**Oracle**	**Acc.**	**Adv. Acc.**	**Oracle**
Normal	79.8	36.5	10.3	84.8	71.3	69.8	89.5	75.4	65.1
Adversarial	79.0	**74.9**	25.8	**85.0**	76.8	74.6	**90.5**	85.5	81.6
Data aug.	79.8	37.8	13.7	85.4	72.7	71.6	88.4	77.5	72.0
Verifiable (IBP)	74.2	73.1	**73.1**	81.7	**77.2**	**76.5**	87.6	**87.1**	**87.1**

Table 1: Experimental results for changes up to δ=3 and δ=2 symbols on SST and AG dataset, respectively. We compare normal training, adversarial training, data augmentation and IBP-verifiable training, using three metrics on the test set: the nominal accuracy, adversarial accuracy, and exhaustively verified accuracy (Oracle) (%).

Prediction	SST word-level examples (by exhaustive verification, not by adversarial attack)
+	it ’ s the kind of pigeonhole-resisting romp that hollywood too rarely provides .
-	it ’ s the kind of pigeonhole-resisting romp that hollywood too rarely **gives** .
-	sets up a nice concept for its fiftysomething leading ladies , but fails loudly in execution .
+	sets up a nice concept for its fiftysomething leading ladies , but fails **aloud** in execution .
Prediction	SST character level examples (by exhaustive verification, not by adversarial attack)
-	you ’ ve seen them a million times .
+	you ’ ve se**r**n them a million times .
+	choose your reaction : a.) that sure is funny !
-	choose **t**our reaction : a.) that sure is funny !

Table 2: Pairs of original inputs and adversarial examples for SST sentiment classification found via an exhaustive verification oracle, but not found by the HotFlip attack (i.e., the HotFlip attack does not change the model prediction). The bold words/characters represent the flips found by the adversary that change the predictions.

4.4 Evaluation Metrics

We use the following four metrics to evaluate our models: i) test set accuracy (called Acc.), ii) adversarial test accuracy (called Adv. Acc.), which uses samples generated by HotFlip attacks on the original test examples, iii) verifiable accuracy under IBP verification (called IBP-verified), that is, the ratio of test samples for which IBP can verify that the specification is not violated, and iv) exhaustively verified accuracy (called Oracle), computed by enumerating all possible perturbations given the perturbation budget δ, where a sample is verifiably robust if the prediction is unchanged under all valid perturbations.

4.5 Results

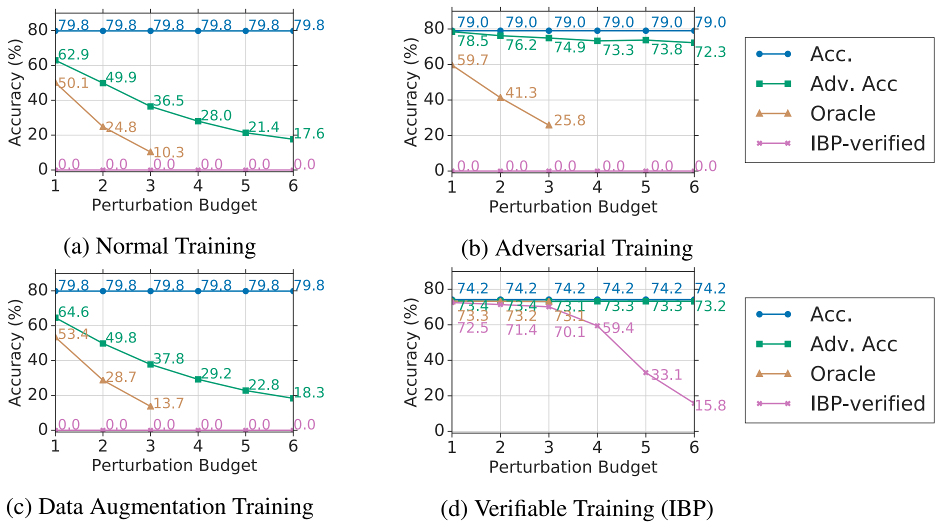
Table 1 shows the results of IBP training and baseline models under $\delta = 3$ and $\delta = 2$[3] perturbations on SST and AG News, respectively. Figures 2 and 3 show the character- and word-level results with δ between 1 and 6 under four metrics on the SST test set; similar figures for SST word-level (adversarial training, data augmentation) models and AG News dataset can be found in the supplementary material.

Oracle Accuracy and Adversarial Accuracy. In Table 1, comparing adversarial accuracy with exhaustive verification accuracy (oracle), we observe that although adversarial training is effective at defending against HotFlip attacks (74.9 / 76.8 / 85.5%), the oracle adversarial accuracy under exhaustive testing (25.8 / 74.6 / 81.6%) is much lower in SST-character / SST-word / AG-character level, respectively. For illustration, we show some concrete adversarial examples from the HotFlip attack in Table 2. For some samples, even though the model is robust with respect to HotFlip attacks, its predictions are incorrect for stronger adversarial examples obtained using the exhaustive verification oracle. This underscores the need for verification, as robustness with respect to suboptimal adversarial attacks alone might give a false sense of security.

Effectiveness of Simplex Bounds with IBP. Rather than sampling individual points from the perturbation space, IBP training covers the full space at once. The resulting models achieve the highest exhaustively verified accuracy at the cost of only moderate deterioration in nominal accuracy (Table 1). At test time, IBP allows for constant-time verification with arbitrary δ, whereas exhaustive verification requires evaluation over an exponentially growing search space.

Perturbation Space Size. In Table 1, when the perturbation space is larger (SST character-level vs. SST word-level), (a) across models, there is a larger gap in adversarial accuracy and true robustness

[3] Note that the exhaustive oracle is not computationally feasible beyond $\delta = 2$ on AG News.



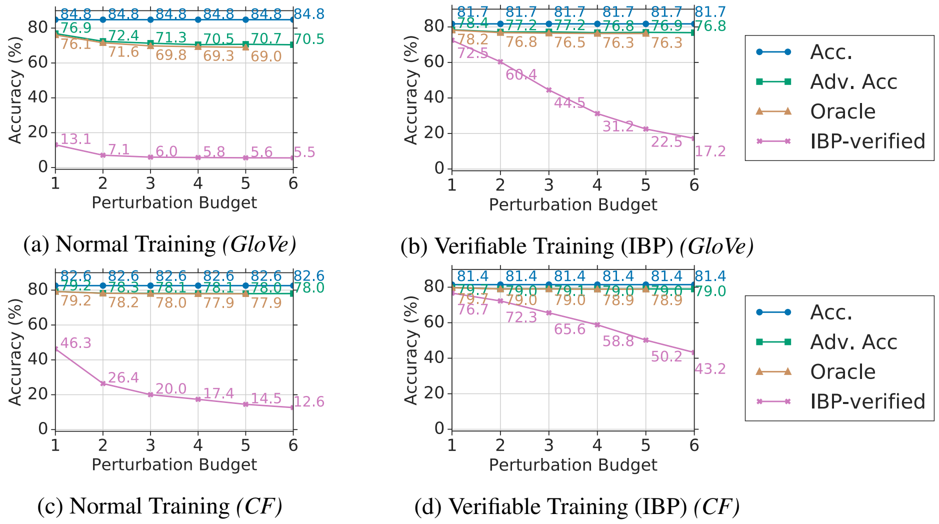
(a) Normal Training

(b) Adversarial Training

(c) Data Augmentation Training

(d) Verifiable Training (IBP)

Figure 2: SST character-level models with different training objectives (trained at δ=3) against different perturbation budgets in nominal accuracy, adversarial accuracy, exhaustively verified accuracy (Oracle), and IBP verified accuracy. Note that exhaustive verification is not scalable to perturbation budget 4 and beyond.



(a) Normal Training *(GloVe)*

(b) Verifiable Training (IBP) *(GloVe)*

(c) Normal Training *(CF)*

(d) Verifiable Training (IBP) *(CF)*

Figure 3: SST word-level models with normal and verifiable training objectives (trained at δ=3) using *GloVe* and *counter-fitted (CF)* embeddings against different perturbation budgets in nominal accuracy, adversarial accuracy, exhaustively verified accuracy (Oracle), and IBP verified accuracy. Note that exhaustive verification is not scalable to perturbation budget 6 and beyond.

(oracle); (b) the difference in oracle robustness between IBP and adversarial training is even larger (73.1% vs. 25.8% and 76.5% vs. 74.6%).

Perturbation Budget. In Figures 2 and 3, we compare normal training, adversarial training, data augmentation, and verifiable training models with four metrics under various perturbation budgets on the SST dataset. Overall, as the perturbation budget increases, the adversarial accuracy, oracle accuracy, and IBP-verified accuracy decrease. We can observe that even for large perturbation budgets, verifiably trained models are still able to verify a sizable number of samples. Again, although adversarial accuracy flattens for larger perturbation budgets in the word level experiments, oracle verification can further find counterexamples to change the prediction. Note that exhaustive verification becomes intractable with large perturbation sizes.

Computational Cost of Exhaustive Verification. The perturbation space in NLP problems is discrete and finite, and a valid option to verify the specification is to exhaustively generate predictions for all $\mathbf{z}_0 \in \mathcal{X}_{\text{in}}(\mathbf{x}_0)$, and then check if at least one does not match the correct label. Conversely, such an

Perturbation radius	$\delta=1$	$\delta=2$	$\delta=3$
SST-word	49	674	5,136
SST-character	206	21,116	1,436,026
AG-character	722	260,282	-

Table 3: Maximum perturbation space size in the SST and AG News test set using word / character substitutions, which is the maximum number of forward passes per sentence to evaluate in the exhaustive verification.

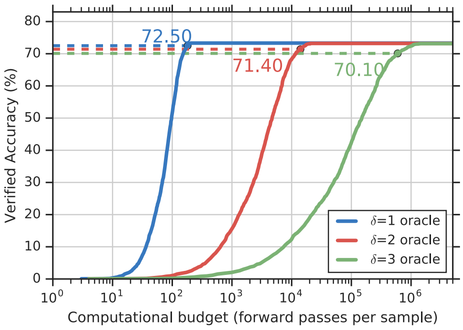


Figure 4: Verified accuracy vs. computation budget for exhaustive verification oracles on the SST character-level test set, for an IBP-trained model trained with δ=3. Solid lines represent the number of forward passes required to verify a given proportion of the test set using exhaustive search. Dashed lines indicate verification levels achievable using IBP verification, which comes at small and constant cost, and is thus orders of magnitude more efficient.

exhaustive (oracle) approach can also identify the strongest possible attack. But the size of $\mathcal{X}_{\text{in}}$ grows exponentially with δ, and exhaustive verification quickly becomes prohibitively expensive.

In Table 3, we show the maximum perturbation space size in the SST and AG News test set for different perturbation radii δ. This number grows exponentially as δ increases. To further illustrate this, Figure 4 shows the number of forward passes required to verify a given proportion of the SST test set for an IBP-trained model using exhaustive verification and IBP verification. IBP reaches verification levels comparable to an exhaustive verification oracle, but requires only two forward passes to verify any sample – one pass for computing the upper, and one for the lower bounds. Exhaustive verification, on the other hand, requires several orders of magnitude more forward passes, and there is a tail of samples with extremely large attack spaces.

4.6 Counter-Fitted Embeddings

As shown in Figures 2 and 3a, although IBP can verify arbitrary networks in theory, the verification bound is very loose except for models trained to be IBP-verifiable. One possible reason is the potentially large volume of the perturbation simplex. Since representations of substitution words/characters are not necessarily close to those of synonyms/typos in embedding space, the vertices of the simplex could be far apart, and thus cover a large area in representation space. Therefore, when propagating the interval bounds through the network, the interval bounds become too loose and fail to verify most of the examples if the models are not specifically trained. To test this hypothesis, we follow Mrkšić et al. (2016) and use fine-tuned GloVe embeddings trained to respect linguistic constraints; these representations (called counter-fitted embeddings) force synonyms to be closer and antonyms to be farther apart using word pairs from the PPDB database (Ganitkevitch et al., 2013) and WordNet (Miller, 1995). We repeat the word level experiments with these counter-fitted embeddings, Figures 3c and 3d show the experimental results. We observe that IBP verified accuracy is now substantially higher across models, especially for $\delta = 1, 2, 3$. The examples which IBP can verify increase by up to 33.2% when using the counter-fitted embeddings (normal training, $\delta = 1$). Moreover, adversarial and exhaustively verified accuracy are also improved, at the cost of a mild deterioration in nominal test accuracy. The IBP-trained model also further improves both its oracle accuracy and IBP verified accuracy. These results validate our hypothesis that reducing the simplex volume via soft linguistic constraints can provide even tighter bounds for IBP, resulting in larger proportions of verifiable samples.

5 Discussion

Our experiments indicate that adversarial attacks are not always the *worst* adversarial inputs, which can only be revealed via verification. On the other hand, exhaustive verification is computationally very expensive. Our results show that using the proposed simplex bounds with IBP can verify a sizable amount of test samples, and can be considered a potent verification method in an NLP context. We note however two limitations within the scope of this work: i) limited model depth: we only investigated models with few layers. IBP bounds are likely to become looser as the number of layers increases. ii) limited model types: we only studied models with CNN and fully connected layers.

We focused on the HotFlip attack to showcase specification verification in the NLP context, with the goal of understanding factors that impact its effectiveness (e.g. the perturbation space volume, see Section 4.6). It is worth noting that symbol substitution is general enough to encompass other threat models such as lexical entailment perturbations (Glockner et al., 2018), and could potentially be extended to the addition of pre/postfixes (Jia and Liang, 2017; Wallace et al., 2019).

Interesting directions of future work include: tightening IBP bounds to allow applicability to deeper models, investigating bound propagation in other types of neural architectures (e.g. those based on recurrent networks or self-attention), and exploring other forms of specifications in NLP.

6 Conclusion

We introduced formal verification of text classification models against synonym and character flip perturbations. Through experiments, we demonstrated the effectiveness of the proposed simplex bounds with IBP both during training and testing, and found weaknesses of adversarial training compared with exhaustive verification. Verifiably trained models achieve the highest exhaustive verification accuracy on SST and AG News. IBP verifies models in constant time, which is exponentially more efficient than naive verification via exhaustive search.

References

Anish Athalye, Nicholas Carlini, and David A. Wagner. 2018. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. In *ICML*, pages 274–283.

Christel Baier and Joost-Pieter Katoen. 2008. *Principles of Model Checking*. MIT press.

Clark Barrett and Cesare Tinelli. 2018. Satisfiability modulo theories. In *Handbook of Model Checking*, pages 305–343. Springer.

Yonatan Belinkov and Yonatan Bisk. 2018. Synthetic and natural noise both break neural machine translation. In *International Conference on Learning Representations*.

Nicola Bertoldi, Richard Zens, and Marcello Federico. 2007. Speech translation by confusion network decoding. In *Proc. ICASSP*.

Rudy Bunel, Ilker Turkaslan, Philip HS Torr, Pushmeet Kohli, and M Pawan Kumar. 2017. Piecewise linear neural network verification: a comparative study. *arXiv preprint arXiv:1711.00455*.

Nicholas Carlini, Guy Katz, Clark Barrett, and David L Dill. 2017. Ground-truth adversarial examples. *arXiv preprint arXiv:1709.10207*.

Nicholas Carlini and David Wagner. 2017. Adversarial examples are not easily detected: Bypassing ten detection methods. In *Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security*, pages 3–14. ACM.

Chih-Hong Cheng, Georg Nührenberg, and Harald Ruess. 2017. Maximum resilience of artificial neural networks. In *International Symposium on Automated Technology for Verification and Analysis*, pages 251–268. Springer.

Minhao Cheng, Jinfeng Yi, Huan Zhang, Pin-Yu Chen, and Cho-Jui Hsieh. 2018. Seq2Sick: Evaluating the robustness of sequence-to-sequence models with adversarial examples. *CoRR*, abs/1803.01128.

Krishnamurthy Dvijotham, Sven Gowal, Robert Stanforth, Relja Arandjelovic, Brendan O'Donoghue, Jonathan Uesato, and Pushmeet Kohli. 2018. Training verified learners with learned verifiers. *arXiv preprint arXiv:1805.10265*.

Christopher Dyer, Smaranda Muresan, and Philip Resnik. 2008. Generalizing word lattice translation. In *Proceedings of ACL-08: HLT*, pages 1012–1020, Columbus, Ohio. Association for Computational Linguistics.

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. Association for Computational Linguistics.

Allyson Ettinger, Sudha Rao, Hal Daumé III, and Emily M. Bender. 2017. Towards linguistically generalizable NLP systems: A workshop and shared task. In *Proceedings of the First Workshop on Building Linguistically Generalizable NLP Systems*.

Juri Ganitkevitch, Benjamin Van Durme, and Chris Callison-Burch. 2013. PPDB: The paraphrase database. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 758–764, Atlanta, Georgia. Association for Computational Linguistics.

Max Glockner, Vered Shwartz, and Yoav Goldberg. 2018. Breaking NLI systems with sentences that require simple lexical inferences. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 650–655, Melbourne, Australia. Association for Computational Linguistics.

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

Sven Gowal, Krishnamurthy Dvijotham, Robert Stanforth, Rudy Bunel, Chongli Qin, Jonathan Uesato, Relja Arandjelovic, Timothy A. Mann, and Pushmeet Kohli. 2018. On the effectiveness of interval bound propagation for training verifiably robust models. *CoRR*, abs/1810.12715.

Matthew Honnibal and Ines Montani. 2017. spaCy 2: Natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing. To appear.

Mohit Iyyer, John Wieting, Kevin Gimpel, and Luke Zettlemoyer. 2018. Adversarial example generation with syntactically controlled paraphrase networks. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1875–1885, New Orleans, Louisiana. Association for Computational Linguistics.

Robin Jia and Percy Liang. 2017. Adversarial examples for evaluating reading comprehension systems. In *Empirical Methods in Natural Language Processing (EMNLP)*.

Guy Katz, Clark Barrett, David L Dill, Kyle Julian, and Mykel J Kochenderfer. 2017. Reluplex: An efficient SMT solver for verifying deep neural networks. In *International Conference on Computer Aided Verification*, pages 97–117. Springer.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. *International Conference on Learning Representations*.

Alexey Kurakin, Ian Goodfellow, and Samy Bengio. 2016. Adversarial examples in the physical world. *arXiv preprint arXiv:1607.02533*.

Yitong Li, Trevor Cohn, and Timothy Baldwin. 2017. Robust training under linguistic adversity. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 21–27.

Xunying Liu, Xie Chen, Yongqiang Wang, Mark J. F. Gales, and Philip C. Woodland. 2016. Two efficient lattice rescoring methods using recurrent neural network language models. *IEEE/ACM Trans. Audio, Speech & Language Processing*, 24(8):1438–1449.

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. 2018. Towards deep learning models resistant to adversarial attacks. In *International Conference on Learning Representations*.

George A. Miller. 1995. WordNet: A lexical database for English. *Commun. ACM*, 38(11):39–41.

Matthew Mirman, Timon Gehr, and Martin Vechev. 2018. Differentiable abstract interpretation for provably robust neural networks. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80, pages 3578–3586.

Nikola Mrkšić, Diarmuid Ó Séaghdha, Blaise Thomson, Milica Gašić, Lina M. Rojas-Barahona, Pei-Hao Su, David Vandyke, Tsung-Hsien Wen, and Steve Young. 2016. Counter-fitting word vectors to linguistic constraints. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 142–148, San Diego, California. Association for Computational Linguistics.

Takashi Onishi, Masao Utiyama, and Eiichiro Sumita. 2010. Paraphrase lattice for statistical machine translation. In *Proceedings of the ACL 2010 Conference Short Papers*, pages 1–5, Uppsala, Sweden. Association for Computational Linguistics.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543. Association for Computational Linguistics.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana. Association for Computational Linguistics.

Aditi Raghunathan, Jacob Steinhardt, and Percy Liang. 2018. Certified defenses against adversarial examples. In *International Conference on Learning Representations*.

Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2018. Semantically equivalent adversarial rules for debugging NLP models. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 856–865. Association for Computational Linguistics.

Dinghan Shen, Guoyin Wang, Wenlin Wang, Martin Renqiang Min, Qinliang Su, Yizhe Zhang, Chunyuan Li, Ricardo Henao, and Lawrence Carin. 2018. Baseline needs more love: On simple word-embedding-based models and associated pooling mechanisms. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 440–450, Melbourne, Australia. Association for Computational Linguistics.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1631–1642. Association for Computational Linguistics.

Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. 2013. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*.

Dimitris Tsipras, Shibani Santurkar, Logan Engstrom, Alexander Turner, and Aleksander Madry. 2019. Robustness may be at odds with accuracy. In *International Conference on Learning Representations*.

Jonathan Uesato, Brendan O'Donoghue, Pushmeet Kohli, and Aron van den Oord. 2018. Adversarial risk and the dangers of evaluating against weak attacks. In *ICML*, pages 5032–5041.

Eric Wallace, Shi Feng, Nikhil Kandpal, Matt Gardner, and Sameer Singh. 2019. Universal trigger sequences for attacking and analyzing NLP. In *Empirical Methods in Natural Language Processing (EMNLP)*.

Shiqi Wang, Kexin Pei, Justin Whitehouse, Junfeng Yang, and Suman Jana. 2018. Formal security analysis of neural networks using symbolic intervals. In *27th USENIX Security Symposium (USENIX Security 18)*, pages 1599–1614, Baltimore, MD. USENIX Association.

Lily Weng, Huan Zhang, Hongge Chen, Zhao Song, Cho-Jui Hsieh, Luca Daniel, Duane Boning, and Inderjit Dhillon. 2018. Towards fast computation of certified robustness for ReLU networks. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 5276–5285, Stockholmsmssan, Stockholm Sweden. PMLR.

Eric Wong and Zico Kolter. 2018. Provable defenses against adversarial examples via the convex outer adversarial polytope. In *International Conference on Machine Learning*, pages 5283–5292.

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. In *Advances in Neural Information Processing Systems*, pages 649–657.