On Adversarial Removal of Hypothesis-only Bias in Natural Language Inference

The Harvard community has made this article openly available. Please share how this access benefits you. Your story matters

<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Citable link</td>
<td><a href="http://nrs.harvard.edu/urn-3:HUL.InstRepos:40827358">http://nrs.harvard.edu/urn-3:HUL.InstRepos:40827358</a></td>
</tr>
<tr>
<td>Terms of Use</td>
<td>This article was downloaded from Harvard University’s DASH repository, and is made available under the terms and conditions applicable to Open Access Policy Articles, as set forth at <a href="http://nrs.harvard.edu/urn-3:HUL.InstRepos:dash.current.terms-of-use#OAP">http://nrs.harvard.edu/urn-3:HUL.InstRepos:dash.current.terms-of-use#OAP</a></td>
</tr>
</tbody>
</table>
Abstract

Popular Natural Language Inference (NLI) datasets have been shown to be tainted by hypothesis-only biases. Adversarial learning may help models ignore sensitive biases and spurious correlations in data. We evaluate whether adversarial learning can be used in NLI to encourage models to learn representations free of hypothesis-only biases. Our analyses indicate that the representations learned via adversarial learning may be less biased, with only small drops in NLI accuracy.

1 Introduction

Popular datasets for Natural Language Inference (NLI) - the task of determining whether one sentence (premise) likely entails another (hypothesis) - contain hypothesis-only biases that allow models to perform the task surprisingly well by only considering hypotheses while ignoring the corresponding premises. For instance, such a method correctly predicted the examples in Table 1 as contradictions. As datasets may always contain biases, it is important to analyze whether, and to what extent, models are immune to or rely on known biases. Furthermore, it is important to build models that can overcome these biases.

Recent work in NLP aims to build more robust systems using adversarial methods (Alzantot et al., 2018; Chen & Cardie, 2018; Belinkov & Bisk, 2018, i.a.). In particular, Elazar & Goldberg (2018) attempted to use adversarial training to remove demographic attributes from text data, with limited success. Inspired by this line of work, we use adversarial learning to add small components to an existing and popular NLI system that has been used to learn general sentence representations (Conneau et al., 2017). The adversarial techniques include (1) using an external adversarial classifier conditioned on hypotheses alone, and (2) creating noisy, perturbed training examples. In our analyses we ask whether hidden, hypothesis-only biases are no longer present in the resulting sentence representations after adversarial learning. The goal is to build models with less bias, ideally while limiting the inevitable degradation in task performance. Our results suggest that progress on this goal may depend on which adversarial learning techniques are used.

Although recent work has applied adversarial learning to NLI (Minervini & Riedel, 2018; Kang et al., 2018), this is the first work to our knowledge that explicitly studies NLI models designed to ignore hypothesis-only biases.

2 Methods

We consider two types of adversarial methods. In the first method, we incorporate an external classifier to force the hypothesis-encoder to ignore hypothesis-only biases. In the second method, we randomly swap premises in the training set to create noisy examples.

Table 1: Examples from SNLI’s development set that Poliak et al. (2018)’s hypothesis-only model correctly predicted as contradictions. The first line in each section is a premise and lines with ▶ are corresponding hypotheses. The italicized words are correlated with the “contradiction” label in SNLI.
2.1 General NLI Model

Let \((P, H)\) denote a premise-hypothesis pair, \(g\) denote an encoder that maps a sentence \(S\) to a vector representation \(v\), and \(c\) a classifier that maps \(v\) to an output label \(y\). A general NLI framework contains the following components:

- A **premise encoder** \(g_P\) that maps the premise \(P\) to a vector representation \(p\).
- A **hypothesis encoder** \(g_H\) that maps the hypothesis \(H\) to a vector representation \(h\).
- A **classifier** \(c_{\text{NLI}}\) that combines and maps \(p\) and \(h\) to an output \(y\).

In this model, the premise and hypothesis are each encoded with separate encoders. The NLI classifier is usually trained to minimize the objective:

\[
L_{\text{NLI}} = L(c_{\text{NLI}}([g_P(P); g_H(H)]; y)) \quad (1)
\]

where \(L(\hat{y}, y)\) is the cross-entropy loss. If \(g_P\) is not used, a model should not be able to successfully perform NLI. However, models without \(g_P\) may achieve non-trivial results, indicating the existence of biases in hypotheses (Gururangan et al., 2018; Poliak et al., 2018; Tsuchiya, 2018).

2.2 AdvCls: Adversarial Classifier

Our first approach, referred to as AdvCls, follows the common adversarial training method (Goodfellow et al., 2015; Ganin & Lempitsky, 2015; Xie et al., 2017; Zhang et al., 2018) by adding an additional adversarial classifier \(c_{\text{Hypoth}}\) to our model. \(c_{\text{Hypoth}}\) maps the hypothesis representation \(h\) to an output \(y\). In domain adversarial learning, the classifier is typically used to predict unwanted features, e.g., protected attributes like race, age, or gender (Elazar & Goldberg, 2018). Here, we do not have explicit protected attributes but rather latent hypothesis-only biases. Therefore, we use \(c_{\text{Hypoth}}\) to predict the NLI label given only the hypothesis. To successfully perform this prediction, \(c_{\text{Hypoth}}\) needs to exploit latent biases in \(h\).

We modify the objective function (1) as

\[
L = L_{\text{NLI}} + \lambda_{\text{Loss}} L_{\text{Adv}}
\]

\[
L_{\text{Adv}} = L(c_{\text{Hypoth}}(\lambda_{\text{Enc}} GRL_{\lambda}(g_H(H)), y))
\]

To control the interplay between \(c_{\text{NLI}}\) and \(c_{\text{Hypoth}}\) we set two hyper-parameters: \(\lambda_{\text{Loss}}\), the importance of the adversarial loss function, and \(\lambda_{\text{Enc}}\), a scaling factor that multiplies the gradients after reversing them. This is implemented by the scaled gradient reversal layer, GRL\(_\lambda\) (Ganin & Lempitsky, 2015). The goal here is to modify the representation \(g_H(H)\) so that it is maximally informative for NLI while simultaneously minimizing the ability of \(c_{\text{Hypoth}}\) to accurately predict the NLI label.

2.3 AdvDat: Adversarial Training Data

For our second approach, which we call AdvDat, we use an unchanged general model, but train it with perturbed training data. For a fraction of example \((P, H)\) pairs in the training data, we replace \(P\) with \(P'\), a premise from another training example, chosen uniformly at random. For these instances, during back-propagation, we similarly reverse the gradient but only back-propagate through \(g_H\). The adversarial loss function \(L_{\text{RandAdv}}\) is defined as:

\[
L(c_{\text{NLI}}([GRL_0(g_P(P'))); \lambda_{\text{Enc}} GRL_{\lambda}(g_H(H))]; y))
\]

where GRL\(_0\) implements gradient blocking on \(g_P\) by using the identity function in the forward step and a zero gradient during the backward step. At the same time, GRL\(_\lambda\) reverses the gradient going into \(g_H\) and scales it by \(\lambda_{\text{Enc}}\), as before.

We set a hyper-parameter \(\lambda_{\text{Rand}} \in [0, 1]\) that controls what fraction \(P\)'s are swapped at random. In turn, the final loss function combines the two losses based on \(\lambda_{\text{Rand}}\) as

\[
L = (1 - \lambda_{\text{Rand}}) L_{\text{NLI}} + \lambda_{\text{Rand}} L_{\text{RandAdv}}
\]

In essence, this method penalizes the model for correctly predicting \(y\) in perturbed examples where the premise is uninformative. This implicitly assumes that the label for \((P, H)\) should be different than the label for \((P', H)\), which in practice does not always hold true.\(^4\)

3 Experiments & Results

Experimental setup Out of 10 NLI datasets, Poliak et al. (2018) found that the Stanford Natural Language Inference dataset (SNLI; Bowman et al., 2015) contained the most (or worst) hypothesis-only biases—their hypothesis-only model outperformed the majority baseline by roughly 100% (going from roughly 34% to 69%). Because of the large magnitude of these biases, confirmed

\(^4\)As pointed out by a reviewer, a pair labeled as neutral might end up remaining neutral after randomly sampling the premise, so adversarial training in this case might weaken the model. Instead, one could limit adversarial training to cases of entailment or contradiction.
by Tsuchiya (2018) and Gururangan et al. (2018), we focus on SNLI. We use the standard SNLI split and report validation and test results. We also test on SNLI-hard, a subset of SNLI that Gururangan et al. (2018) filtered such that it may not contain unwanted artifacts.

We apply both adversarial techniques to InferSent (Conneau et al., 2017), which serves as our general NLI architecture. Following the standard training details used in InferSent, we encode premises and hypotheses separately using bi-directional long short-term memory (BiLSTM) networks (Hochreiter & Schmidhuber, 1997). Premises and hypotheses are initially mapped (token-by-token) to Glove (Pennington et al., 2014) representations. We use max-pooling over the BiLSTM states to extract premise and hypothesis representations and, following Mou et al. (2016), combine the representations by concatenating their vectors, their difference, and their multiplication (element-wise).

We use the default training hyper-parameters in the released InferSent codebase. These include setting the initial learning rate to 0.1 and the decay rate to 0.99, using SGD optimization and dividing the learning rate by 5 at every epoch when the accuracy decreases on the validation set. The default settings also include stopping training either when the learning rate drops below $10^{-5}$ or after 20 epochs. In both adversarial settings, the hyper-parameters are swept through \{0.05, 0.1, 0.2, 0.4, 0.8, 1.0\}.

**Results** Table 2 reports the results on SNLI, with the configurations that performed best on the validation set for each of the adversarial methods.

<table>
<thead>
<tr>
<th>Model</th>
<th>Val</th>
<th>Test</th>
<th>Hard</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baseline</td>
<td>84.25</td>
<td>84.22</td>
<td>68.02</td>
</tr>
<tr>
<td>AdvCls</td>
<td>84.58</td>
<td>83.56</td>
<td>66.27</td>
</tr>
<tr>
<td>AdvDat</td>
<td>78.45</td>
<td>78.30</td>
<td>55.60</td>
</tr>
</tbody>
</table>

Table 2: Accuracies for the approaches. Baseline refers to the unmodified, non-adversarial InferSent.

As expected, both training methods perform worse than our unmodified, non-adversarial InferSent baseline on SNLI’s test set, since they remove biases that may be useful for performing this task. The difference for AdvCls is minimal, and it even slightly outperforms InferSent on the validation set. While AdvDat’s results are noticeably lower than the non-adversarial InferSent, the drops are still less than 6% points.

**4 Analysis**

Our goal is to determine whether adversarial learning can help build NLI models without hypothesis-only biases. We first ask whether the models’ learned sentence representations can be used by a hypothesis-only classifier to perform well. We then explore the effects of increasing the adversarial strength, and end with a discussion of indicator words associated with hypothesis-only biases.

**4.1 Hidden Biases**

Do the learned sentence representations eliminate hypothesis-only biases after adversarial training?

---

2Code developed is available at https://github.com/azpoliak/robust-nli.

3https://github.com/facebookresearch/InferSent
We freeze sentence encoders trained with the studied methods, and retrain a new classifier that only accesses representations from the frozen hypothesis encoder. This helps us determine whether the (frozen) representations have hidden biases.

A few trends can be noticed. First, we confirm that with AdvCls (Figure 1a), the hypothesis-only classifier (c_hypoth) is indeed trained to perform poorly on the task, while the normal NLI classifier (c_NLI) performs much better. However, retraining a classifier on frozen hypothesis representations (c_Hypoth, retrained) boosts performance. In fact, the retrained classifier performs close to the fully trained hypothesis-only baseline, indicating the hypothesis representations still contain biases.

Consistent with this finding, Elazar & Goldberg (2018) found that adversarially-trained text classifiers preserve demographic attributes in hidden representations despite efforts to remove them.

Interestingly, we found that even a frozen random encoder captures biases in the hypothesis, as a classifier trained on it performs fairly well (63.26%), and far above the majority class baseline (34.28%). One reason might be that the word embeddings (which are pre-trained) alone contain significant information that propagates even through a random encoder. Others have also found that random encodings contain non-trivial information (Conneau et al., 2018; Zhang & Bowman, 2018). The fact that the word embeddings were not updated during (adversarial) training could account for the ability to recover performance at the level of the classifier trained on a random encoder. This may indicate that future adversarial efforts should be applied to the word embeddings as well.

Turning to AdvDat, (Figure 1b), as the hyper-parameters increase, the models exhibit fewer biases. Performance even drops below the random encoder results, indicating it may be better at ignoring biases in the hypothesis. However, this comes at the cost of reduced NLI performance.

4.2 Adversarial Strength

Is there a correlation between adversarial strength and drops in SNLI performance? Does increasing adversarial hyper-parameters affect the decrease in results on SNLI?

Figure 2 shows the validation results with various configurations of adversarial hyper-parameters. The AdvCls method is fairly stable across configurations, although combinations of large λ_loss and λ_enc hurt the performance on SNLI a bit more (Figure 2a). Nevertheless, all the drops are moderate. Increasing the hyper-parameters further (up to values of 5), did not lead to substantial drops, although the results are slightly less stable across configurations (Appendix A). On the other hand, the AdvDat method is very sensitive to large hyper-parameters (Figure 2b). For every value of λ_enc, increasing λ_rand leads to significant performance drops. These drops happen sooner for larger λ_enc values. Therefore, the effect of stronger hyper-parameters on SNLI performance seems to be specific to each adversarial method.

4.3 Indicator Words

Certain words in SNLI are more correlated with specific entailment labels than others, e.g., negation words (“not”, “nobody”, “no”) correlated with CONTRADICTION (Gururangan et al., 2018; Poliak et al., 2018). These words have been referred to as “give-away” words (Poliak et al., 2018). Do the adversarial methods encourage models to make predictions that are less affected by these biased indicator words?
Table 3: Indicator words and how correlated they are with CONTRADICTION predictions. The parentheses indicate hyper-parameter values: $(\lambda_{\text{Loss}}, \lambda_{\text{Enc}})$ for AdvCls and $(\lambda_{\text{Rand}}, \lambda_{\text{Enc}})$ for AdvDat. Baseline refers to the unmodified InferSent.

| Word  | Count | $\hat{p}(l|w)$ | Baseline | AdvCls (1,1) | AdvDat (0.4,1) | AdvDat (1,1) |
|-------|-------|----------------|-----------|--------------|----------------|--------------|
| sleeping | 108   | 0.88           | 0.24      | 15.63        | 53.13          | -81.25       |
| driving | 53    | 0.81           | 0.32      | -8.33        | 50             | -66.67       |
| Nobody  | 52    | 1              | 0.42      | 14.29        | 42.86          | 14.29        |
| alone   | 50    | 0.9            | 0.32      | 0            | 83.33          | 0            |
| cat     | 49    | 0.84           | 0.31      | 7.14         | 57.14          | -85.71       |
| asleep  | 43    | 0.91           | 0.39      | -18.75       | 50             | 12.5         |
| no      | 31    | 0.84           | 0.36      | 0            | 52.94          | -52.94       |
| empty   | 28    | 0.93           | 0.3       | -16.67       | 83.33          | -16.67       |
| eats    | 24    | 0.83           | 0.3       | 37.5         | 87.5           | -25          |
| naked   | 20    | 0.95           | 0.46      | 0            | 83.33          | -33.33       |

5 Conclusion

We employed two adversarial learning techniques to a general NLI model by adding an external adversarial hypothesis-only classifier and perturbing training examples. Our experiments and analyses suggest that these techniques may help models exhibit fewer hypothesis-only biases. We hope this work will encourage the development and analysis of models that include components that ignore hypothesis-only biases, as well as similar biases discovered in other natural language understanding tasks (Schwartz et al., 2017), including visual question answering, where recent work has considered similar adversarial techniques for removing language biases (Ramakrishnan et al., 2018; Grand & Belinkov, 2019).

6 Acknowledgements

This work was supported by JHU-HLTCOE, DARPA LORELEI, and the Harvard Mind, Brain, and Behavior Initiative. We thank the anonymous reviewers for their comments. Views and conclusions contained in this publication are those of the authors and should not be interpreted as representing official policies or endorsements of DARPA or the U.S. Government.

References


A Stronger hyper-parameters for AdvCls

Figure 3 provides validation results using AdvCls with stronger hyper-parameters to complement the discussion in §4.2. While it is difficult to notice trends, all configurations perform similarly and slightly below the baseline. These models seem to be less stable compared to using smaller hyper-parameters, as discussed in §4.2.

Figure 3: Validation results using AdvCls with stronger hyper-parameters.