Does Data Augmentation Improve Generalization in NLP?

Rohan Jha
Brown University

Charles Lovering
Brown University

Ellie Pavlick
Brown University

{first}_{last}@brown.edu

Abstract

Neural models often exploit superficial features to achieve good performance, rather than deriving more general features. Overcoming this tendency is a central challenge in areas such as representation learning and ML fairness. Recent work has proposed using data augmentation, i.e., generating training examples where the superficial features fail, as a means of encouraging models to prefer the stronger features. We design a series of toy learning problems to test the hypothesis that data augmentation leads models to unlearn weaker heuristics, but not to learn stronger features in their place. We find partial support for this hypothesis: Data augmentation often hurts before it helps, and it is less effective when the preferred strong feature is much more difficult to extract than the competing weak feature.

1 Introduction

Neural models often perform well on tasks by using superficial ("weak") features, rather than the more general ("strong") features that we'd prefer them to use. As a result, models often fail in systematic ways. For example, in visual question answering, models failed when tested on rare color descriptions (*"green bananas"*) (Agrawal et al., 2018); in coreference, models failed when tested on infrequent profession-gender pairings (*"the nurse cared for his patients"*) (Rudinger et al., 2018); in natural language inference (NLI), models failed on sentence pairs with high lexical overlap but different meanings (*"man bites dog"*/*"dog bites man"*) (McCoy et al., 2019). One proposed solution has been to augment training data with "counterexamples" to the model's adopted heuristics, i.e., training examples on which the weak features fail. This technique has shown positive initial results for POS tagging (Elkahky et al., 2018), NLI (Min et al., 2020), and reducing gender bias (Zhao et al., 2018; Zmigrod et al., 2019). However, it is not yet known whether this strategy is a feasible way of improving systems in general.

Our hypothesis is that augmenting data with such counterexamples will lead models to unlearn weaker features as intended (e.g., reducing the correlation between gender and profession) but will not necessarily lead models to adopt the "better" stronger features we intend them to use instead (e.g., syntax, common sense inference). To test this, we design a set of toy learning problems that contain two competing features which differ both in how well they predict the label and in how easy they are for the model to extract. We find that:

- Data augmentation is less effective when the preferred ("strong") feature is harder for the model to extract from its input (§4.1).
- Adding counterexamples often hurts before it helps, and at low levels ($\sim\leq 1\%$ of training data), data augmentation leads models to shift to new weak features rather than to the stronger feature (§4.2). This suggests apparent gains from data augmentation could be misleading, especially if we only evaluate on the targeted phenomenon.
- Data augmentation only works when the strong feature is already sufficiently present in training. When training data is pathologically skewed such that the strong feature very rarely or never occurs without the weak feature, data augmentation has no effect (§4.3).

Note on Terminology: The goal of this study is to isolate the phenomenon of interest (data augmentation in NLP) and observe patterns without the confounds that exist in applied settings. By homing in on specific empirical trends, we hope to lay the groundwork for more formal subsequent analysis. As such, our choice of terminology

(*"strong"*, *"weak"*, *"hard"*, *"counterexample"*) is meant to be informal. We deliberately avoid more widely-used terms (*"adversarial examples"*, *"spurious correlations"*, *"counterfactual"*, etc.) since we do not yet wish to invoke specific connotations or assumptions about modeling approach, causality, or downstream application.

2 Experimental Design

2.1 Setup

We use a binary sequence classification task. We assume there exists some **strong feature** which perfectly predicts the label (i.e., the label is 1 *iff* the strong feature holds), but which is non-trivial to extract given the raw input. Additionally, there exists a **weak feature** which is easy for the model to extract from the input and which frequently co-occurs with the strong feature in training. Thus, a model which only represents the weak feature can make correct predictions much of the time. We can vary the features' co-occurence by adding **counterexamples** to training in which either the strong or the weak feature is present, but not both (see Fig. 1). Intuitively, we intend the weak feature to be representative of features such as lexical priors (e.g., *"not"* being indicative of contradiction in NLI) and the strong feature to be representative of syntactic and compositional semantic features.

2.2 Task and Model

We use a synthetic sentence classification task with k-length sequences of numbers as input and binary labels as output. We use a symbolic vocabulary V with the integers $0 \dots |V|$. We fix $k = 10$ and $|V| = 50$K. We use an initial training size of 200K examples, though the total training size varies as we add counterexamples (§2.5). Test and validation sizes are 40K each. Our classifier is a simple network with an embedding layer, a 1-layer LSTM, and an MLP with 1 hidden layer and ReLU activation (about 13M parameters). We use Intel Cascade CPUs, and models are trained until convergence using early-stopping. Various ablations are included in Supplementary Material (§A). The test and validation vocabularies are disjoint from the training; for the model to test well, it has to learn the intended strong feature (i.e., the notion of a duplicated symbol) vs. finding an easier workaround (e.g., memorizing bigrams). Models converge on average by 8 epochs (500 seconds each). Our code is available for reproducibility.[1]

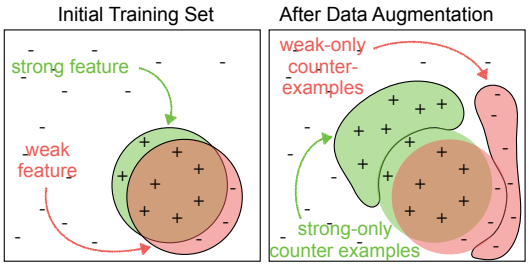

Figure 1: Schematic of experimental setup.

2.3 Strong and Weak Features

In all experiments, we set the weak feature to be the presence of the symbol 2 anywhere in the input. We consider several different strong features (Table 1), intended to vary in how "hard" they are for the classifier to detect. To quantify the notion of **hardness**, we use minimum description length (MDL), a recently introduced information-theoretic metric designed to capture the "extractability" of a feature from a representation (Voita and Titov, 2020). Intuitively, MDL computes the minimum number of bits that would be needed to encode the labels given the representations: The more directly the representations encode the labels, the fewer the bits needed. To compute MDL[2], we train a model to predict directly whether each feature holds, using a set of 200K training examples evenly split between cases when the feature does and does not hold. Table 1 contains the MDL metrics for each feature (averaged over 3 re-runs). We see the desired gradation of feature hardness. The validation error of the converged classifier is $< 7.5\%$ in all cases.

2.4 Performance Metrics

We partition test accuracy into four cases, since we are interested in measuring accuracy in relation to the presence or absence of the weak feature.

***weak-only* acc:**	$P(pred = 0 \mid wk, \neg str)$
***strong-only* acc:**	$P(pred = 1 \mid \neg wk, str)$
***both* acc:**	$P(pred = 1 \mid wk, str)$
***neither* acc:**	$P(pred = 0 \mid \neg wk, \neg str)$

Our test and validation sets have 10K examples for each of these cases.[3] Intuitively, low *weak-only* accuracy indicates the model associates the weak feature with the positive label, whereas low

[1] Code at `http://bit.ly/bb-data-aug`.

[2] We compute the online MDL; see (Voita and Titov, 2020) for implementation details.

[3] Validation is always within 1% of average test error.

Feature Nickname	Description	Data MDL	Model MDL	Example
`contains-1`	`1` occurs somewhere in the sequence	1.2×10^1	2.8×10^2	`2 4 11 1 4`
`prefix-dup`	Sequence begins with a duplicate	3.0×10^2	8.0×10^4	`2 2 11 12 4`
`first-last`	First number equals last number	3.3×10^2	1.2×10^5	`2 4 11 12 2`
`adjacent-dupl`	Adjacent duplicate anywhere in seq.	5.8×10^2	2.5×10^5	`11 12 2 2 4`

Table 1: Features used to instantiate the strong feature in our experiments. Features are intended to differ in how hard they are for an LSTM to detect given sequential input. The MDL statistics are measured in bits (§2.3).

strong-only accuracy indicates the model either fails to detect the strong feature altogether, or detects it but fails to associate it with positive label. In practice, we might prioritize these accuracies differently in different settings. For example, in work on fairness (Hall Maudslay et al., 2019), we target *weak-only* accuracy, since the primary goal is to prevent the model from falsely associating protected attributes with specific labels or outcomes. In contrast, when aiming to improve the overall robustness of NLP models (Min et al., 2020), we presumably target *strong-only* accuracy, since the hope is that, by lessening the effectiveness of shallow heuristics, we will encourage models to learn deeper, more robust features instead.

2.5 Counterexamples

As with error categories, we distinguish between types of counterexamples: ***weak-only* counterexamples** where the weak feature occurs without the strong feature and the label is 0, and ***strong-only* counterexamples** where the strong feature occurs without the weak feature and the label is 1. These two types are meaningfully different in practical settings, as often *weak-only* counterexamples are easy to obtain, whereas *strong-only* counterexamples are difficult or impossible to obtain. For example, in the case of NLI and the lexical overlap heuristic from McCoy et al. (2019), it is easy to generate *weak-only* counterexamples (sentence pairs with high lexical overlap but which are not in an entailment relation) using a set of well-designed syntactic templates. However, generating *strong-only* examples (entailment pairs without lexical overlap) likely requires human effort (Williams et al., 2018). This difference is exacerbated in realistic problems where there are many weak features and/or it is impossible to fully isolate the strong features from all weak features (e.g., ultimately, it may not be possible to decouple the "underlying semantics" of a sentence from all lexical priors). We characterize most recent applied work on data augmentation (Elkahky et al., 2018; Min et al., 2020; Zhao et al., 2018; Zmigrod et al., 2019) as using *weak-only* examples.

We begin with an initial training set of 200K, evenly split between *both* and *neither* examples. We then add all combinations of i *weak-only* counterexamples and j *strong-only* counterexamples for $i, j \in \{0, 10, 100, 500, 1\text{K}, 2.5\text{K}, 5\text{K}, 25\text{K}, 50\text{K}, 100\text{K}\}$. This results in 1,191 total models trained: 3 random restarts[4] for 100 combinations of i, j over the four features in Table 1.

3 Initial Results

Using this terminology, our hypothesis is: Adding *weak-only* counterexamples will improve *weak-only* accuracy but will not improve *strong-only* accuracy. As a first-pass analysis, we use a standard multiple regression.[5] This allows us to observe the first-order effect of the number and type of counterexample on accuracy, controlling for the effects that additional counterexamples have on total training size and on training label skew. For the moment, we focus only on highlighting general relationships between the number and type of counterexample added and the model's *weak-only* and *strong-only* accuracies. We focus in this way for simplicity, to give a starting point of our analysis, and don't claim that this tells the entire story. In Section 4, we look more deeply at the trends observed in relation to *both* and *neither* accuracy.

We run four regressions, one per accuracy metric. We include variables to control for the confounding effects of adding counterexamples, e.g., the label skew and the total number of examples. These aren't shown because of space constraints.[6] As preprocessing, we take the log of number of counterexamples and of total training size, and we normalize all explanatory variables to have a mean of 0 and standard deviation of 1. Table 2 shows the

[4] It isn't always exactly 3 restarts due to some job failures.
[5] `linear_model.OLS` from `statsmodels`
[6] Supplementary has these metrics and *both* and *neither*.

regression results. There are two main takeaways.

First, we see a significant, positive relationship both between the number of *weak-only* examples and *weak-only* accuracy ($\beta = 0.37$, $p = 0.00$) but no significant relationship between the number of *weak-only* examples and *strong-only* accuracy ($\beta = 0.03$, $p = 0.25$). This is consistent with our hypothesis: All else being equal, *weak-only* examples have a much larger effect on *weak-only* accuracy than on *strong-only* accuracy. When looking at the effects associated with adding *strong-only* examples, we see a strong positive effect on *strong-only* accuracy ($\beta = 0.46$, $p = 0.00$), but a modest negative effect on *weak-only* accuracy ($\beta = -0.14$, $p = 0.00$). This seems counterintuitive; we investigate further in Section 4.2.

The second trend that stands out from the regression results is the clear association between the hardness of the strong feature, as measured by MDL, and the model's accuracy. That is, harder features have more negative impact on accuracy. To understand this trend better, we run a second regression that includes interaction terms between the number of counterexamples and each specific strong feature. We find that our hardest feature (`adjacent-dupl`) behaves differently than all the others: *strong-only* examples hurt *weak-only* accuracy ($\beta = -0.09, p = 0.00$), and *weak-only* examples fail to help *strong-only* accuracy ($\beta = -0.05, p = 0.17$). For all other features, counterexamples have a positive (albeit asymmetric) effect on both metrics: *weak-only* examples have a roughly $3\times$ to $5\times$ larger effect on *weak-only* accuracy than on *strong-only* accuracy, and vice-versa for *strong-only* examples.[7]

		weak-only acc. $R^2 = 0.74$		*strong-only* acc. $R^2 = 0.70$	
	σ	β	p	β	p
log weak-only ex.	1.5	0.37	0.00*	0.03	0.25
log strong-only ex.	1.5	−0.14	0.00*	0.46	0.00*
contains-1		0.89	0.00*	0.90	0.00*
prefix-dupl		−0.07	0.03*	−0.01	0.71
first-last		−0.26	0.00*	−0.43	0.00*
adjacent-dupl		−0.58	0.00*	−0.48	0.00*

Table 2: Regression results for predicting *weak-only* and *strong-only* accuracy. All explanatory variables are z-normalized, so coefficients should be interpreted in units of σ (variable's st. dev.). First section shows effects of number and type of counterexamples. Second section shows effects of dummy variables for each individual feature type.

4 In-Depth Analyses

Regression analysis cannot tell the complete story. We thus take a closer look at the effect of hardness (§4.1) and at the sometimes-negative effects associated with data augmentation (§4.2 and §4.3).

4.1 Effect of Strong Feature's Hardness

In Section 3, we observed a relationship between the hardness of the strong feature and both the overall accuracy of the model and the effectiveness of counterexamples. To better understand this relationship, we look at the mutual information $\mathcal{I}$ between the model's prediction at test time and the presence of each feature. In a perfect model (i.e., one basing its decisions entirely on the strong feature), the presence of the weak feature would have no effect on the prediction, so $\mathcal{I}(weak, pred) \approx 0$ and $\mathcal{I}(strong, pred) \approx 1$. In Figure 2, for each feature, we look at $\mathcal{I}$ as a function of the relative frequency of *weak-only* vs. *strong-only* examples in the training data. Specifically, we plot $\mathcal{I}(weak, pred)$ and $\mathcal{I}(strong, pred)$ as a function of the ratio $m : n$ where m is the number of *both* examples and n is the number of *weak-only* examples. We also vary the number of *strong-only* examples (§2.5), and Figure 2 shows the aggregated trends (across all numbers of *strong-only* examples) as well as the deaggregated runs.

There is a clear association between the hardness of the strong feature and the sensitivity of the model to the dataset distribution (the $m : n$ ratio). For example, for `contains-1`, even when the training data is pathologically skewed (10^4 *both* examples for every 1 *weak-only* example), the model learns the desired associations: the strong feature alone determines the prediction. In contrast, for `adjacent-dupl`, even when the data is perfectly balanced with an equal number of *both* and *weak-only* examples, the model still does not exhibit the desired behavior. Rather, the model appears to unlearn the heuristic based on the weak feature ($I(weak, pred) \approx 0$) but does not use the *strong* feature as it should ($I(strong, pred) << 1$). Note, in some settings–when the number of *strong-only* examples in the training data is very high–the model appears to learn the correct associations. We note that, in practice, this is likely to be a property of the training data/task that is outside of our control as practitioners (§2.5).

[7] See Supplementary Material (Table 4).

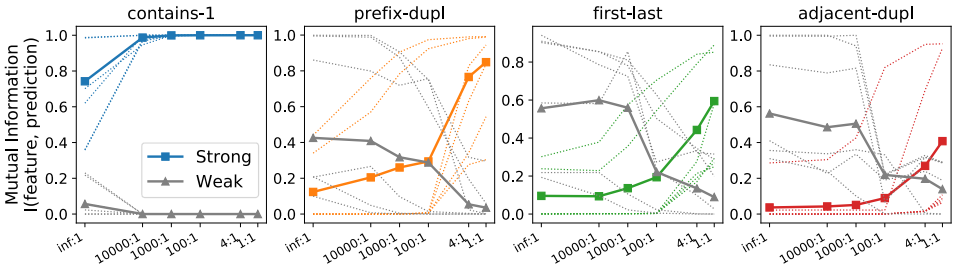

Figure 2: Mutual information (y-axis) between presence of feature and model's prediction, as a function of dataset skew (x-axis). Skew $m : n$ means that the features occur together m times for every n times the weak feature occurs alone. Ideally, $\mathcal{I}(weak, pred)$ would be near 0 and $\mathcal{I}(strong, pred)$ near 1. Dark lines are aggregated over multiple simulations (in which we alter the frequency of *strong-only* as discussed in §2.5). Deaggregated runs shown as thin lines; when there is variation, runs achieving high $\mathcal{I}(strong, pred)$ and low $\mathcal{I}(weak, pred)$ are those with higher numbers of *strong-only* examples (discussed more in §4.3).

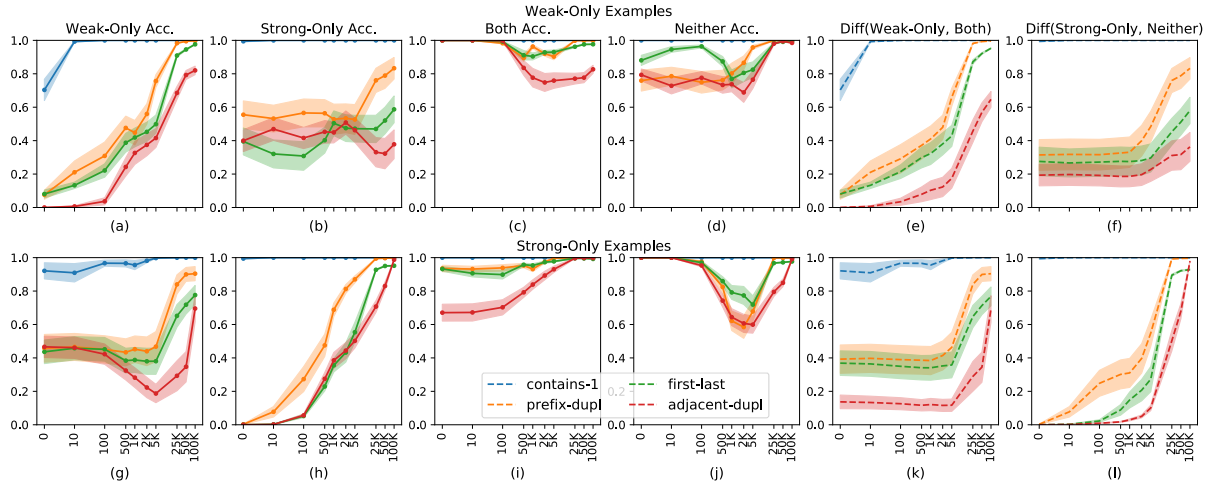

Figure 3: **Left (a–d and g–j):** Accuracies of models for a given number of counterexamples of one type, averaged over all values of counterexamples of the other type. Graphs should be compared in terms of trajectories (increasing vs. decreasing) rather than absolute numbers along the y-axis (0 vs. 0.5). This is because starting points on the y-axis are not always comparable. E.g., the difference in the starting points of (a) vs. (b) reflects that runs with high numbers of *strong-only* examples favor *strong-only* acc (b) more than they favor *weak-only* acc (a). **Right (e–f and k–l):** Differences between classification rates. See text for details.

4.2 Nonlinear Trends

Our regression analysis suggests that the effect of adding counterexamples is sometimes negative. To further investigate how accuracy changes as a function of the number and type of counterexample, we focus on the effect of adding one type of counterexample (*weak-only* or *strong-only*), averaging over all levels of the other type of counterexample (*strong-only* or *weak-only*, respectively). In Section 4.3, we look at the deaggregated trends.

General Observations. Figure 3 shows aggregated trends over each feature and accuracy metric. We see the expected positive linear relationship between *weak-only* examples and *weak-only* accuracy, and between *strong-only* examples and *strong-only* accuracy. The other relationships are more nuanced: e.g., the effect of *strong-only* examples on *weak-only* accuracy (Fig. 3(g)), and the effect of either example type on *neither* accuracy (Figs. 3(d,j)), appear to produce U-shaped curves.

Relationship to Decision Rules. To understand these trends better, we attempt to connect them to the "rules" a model could use. For example, if a model uses the heuristic $weak \rightarrow 1$ and $\neg weak \rightarrow 0$, we'd expect low *strong-only* and *weak-only* accuracy and high *both* and *neither* accuracy. Table 3 summarizes the rules the model might use if it represents the weak feature w or the strong feature s. This discussion is intended to be qualitative and informal, to summarize our inter-

pretation of the quantitiative trends in Fig. 3.

Adding *strong-only* examples. We look first at the effects of adding *strong-only examples*. Before adding any counterexamples, the model achieves low *strong-only* and *weak-only* accuracies, and high *both* and *neither* accuracies. This behavior is consistent with the rule $w \rightarrow 1, \neg w \rightarrow 0$. Once we begin adding counterexamples, we see that learning can be divided into two clear phases. First, at low levels of data augmentation (up to 2K examples, around 1% of the training data), *weak-only* and *neither* accuracies are either flat or decreasing while *strong-only* and *both* accuracies are increasing. This behavior would be consistent with a shift from using $\neg w \rightarrow 0$ to using $\neg w \rightarrow 1$. Second, at higher levels of augmentation, the trajectory shifts, and all accuracy metrics begin increasing in tandem. This behavior is not consistent with any combination of rules that rely only on w, and thus suggests that the model is beginning to represent and to use the strong feature. This story is summarized in Table 3.

Figures 3(k,l) tell this same story a bit differently, by looking at the (absolute) difference between $\hat{P}(1|\textit{weak-only})$ and $\hat{P}(1|\textit{both})$ and at the difference between $\hat{P}(1|\textit{strong-only})$ and $\hat{P}(1|\textit{neither})$. Here, $\hat{P}(1)$ is the rate at which the model predicts 1. When the model is only using the weak feature in its decisions, these metrics necessarily move against each other: e.g., it is not possible to differentiate between *both* and *weak-only* unless the strong feature is somehow represented. Thus, during the first phase (low levels of data augmentation), we see that these differences are flat, suggesting that changes in these metrics exactly offset one another and thus the role of the strong feature in the models' predictions is not changing. At higher levels of data augmentation, these metrics stop competing, and we see the differences going to one, suggesting the model cannot be relying on the weak feature alone.

Adding *weak-only* examples. The trends for *weak-only* examples (Figs. 3(a–f)) are similar to those for *strong-only* examples. However, curves show higher variance in general, and when the model shifts from the first phase (where it moves from one weak-feature-based rule to another) to the second (where it begins using the strong feature) the *strong-only* accuracy learning curve is much flatter. This is exacerbated when the strong feature is harder to extract (i.e., `first-last` and `adjacent-dupl`). As a result, at the levels of data augmentation we explore, we never see the model achieve high *strong-only* accuracy on these features. However, we do see the model reach perfect performance on all other accuracy metrics. This is consistent with the model learning to use the strong feature in some cases, but not yet abandoning the $\neg w \rightarrow 0$ heuristic.

4.3 Interaction Between Example Types

Above, we focused on the effect of adding one type of counterexample, largely ignoring how the number counterexamples of other type already present in the training data impact results. The two counterexample types are likely to complement one another, and different learning problems warrant different assumptions about how the initial training set is distributed. Thus, it's important to understand how the effectiveness of data augmentation changes as we vary assumptions about the initial distribution of the training data.

Figure 4 shows each accuracy metric across all combinations of # *strong-only* examples $\times$ # *weak-only* examples for the `adjacent-dupl` feature.[8] We see that when the number of *strong-only* examples is extremely low (0 or 10), no number of *weak-only* examples affects *strong-only* error, and vice-versa when then number of *weak-only* examples is extremely low. Perhaps more interestingly, we see that when the number of *strong-only* examples is low but not negligible (around 1% of training data), adding *weak-only* examples can hurt rather than help. We hypothesize that around this level of data skew, the model learns something other than just the weak feature but has insufficient data to learn the strong feature. Thus, it uses features that are stronger during training but fail to generalize to test (e.g. character bigrams).

5 Discussion and Future Work

Our analysis reveals a few trends that warrant further investigation, as they may have implications for the effectiveness of data augmentation and the behavior of models trained via these methods. First, the U-shaped learning curves observed suggest that low levels of data augmentation are likely only to improve some metrics at the expense of others. There are two notable properties of the

[8] Other features given in Supplementary Material (§C).

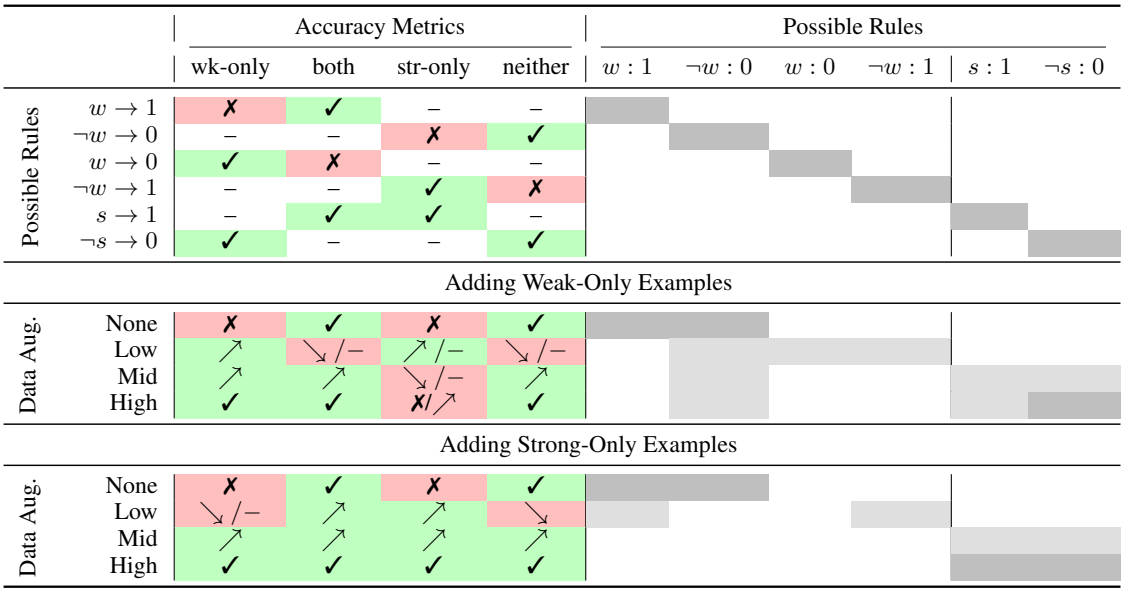

Table 3: Summary of rule-like behavior of model at different levels of data augmentation. Symbols (✓, ✗) denote stable points, at the beginning or when the model seems to have converged. Arrows (↗, ↘) denote trends, when the model is shifting behavior. Slashes indicate ambiguity, when different features behave notably differently. Dark gray means that the model is behaving consistently with a given rule. Lighter gray means error patterns are partially consistent with and/or trending toward the rule's error patterns.

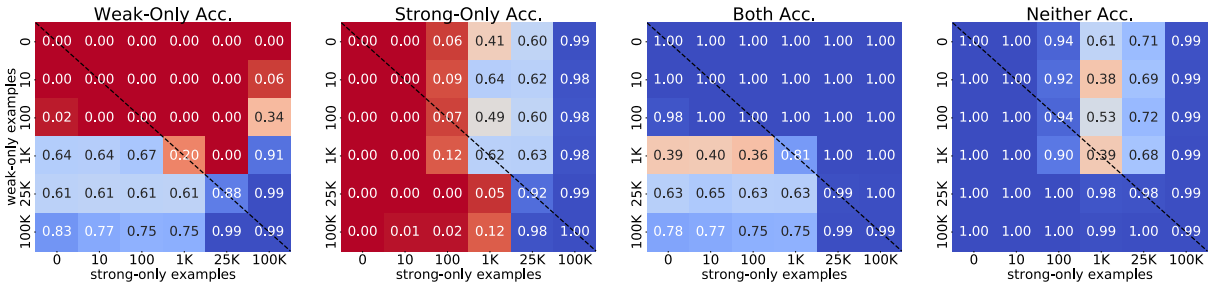

Figure 4: Accuracies for `adjacent-dupl` with # *weak-only* × # *strong-only* examples.

learning problem that appear to make this problem more pronounced: a) when only *weak-only* examples are added and b) when the strong feature is significantly more difficult to extract from the input than the targeted weak feature. These two properties are significant as they correspond to the typical setting in which we'd expect to use data augmentation in practice. One straightforward takeaway from this is the importance of thoroughly evaluating models trained via data augmentation. Evaluating only on the targeted phenomenon (*weak-only* error) is likely to over-state the gains; evaluating only on aggregated metrics (accuracy on standard test sets) is likely to hide interesting win-loss patterns.

Second, the relationship between the strong feature's "hardness" and the model's sensitivity to counterexamples (of both types) suggests promising directions for future work. In particular, our experiments suggested that current data augmentation practices–i.e. adding small numbers of *weak-only* examples–can be effective as long as the strong feature is not too much harder than the weak feature to extract from the input. This suggests that further investing in representation learning, and specifically prioritizing the *extractability* of a feature in the input representation, could yield better outcomes than investing in data augmentation alone. Such a direction could be interesting from both a practical and a theoretical perspective.

Finally, there are limitations in our setup. We focus on a simple model and task, and it remains to be seen whether similar patterns hold for e.g., large pretrained language models. Also, we use a fixed training size (200K) and cannot say whether the trends we observed across "low" vs. "high"

levels of data augmentation are better understood as absolute numbers (low=2K counterexamples) or as a proportion of training data (low=1%) (or rather both). Finally, we focused only on the case in which there is a single strong feature and a single weak feature in the data. In real language tasks, this assumption would never hold.

6 Related Work

Data Augmentation. A wave of recent work has constructed evaluation sets composed of "adversarial examples" (or "challenge examples" or "probing sets") to analyze weaknesses in the decision procedures of neural NLP models (Jia and Liang, 2017; Glockner et al., 2018; Dasgupta et al., 2018; Gururangan et al., 2018; Poliak et al., 2018b, and others). Our work is motivated by the subsequent research that asks if adding such examples to training data help improve robustness. Liu et al. (2019) show that fine-tuning on small challenge sets can sometimes (but not always) help model performance. Similar approaches have been explored for improving syntactic parsing (Elkahky et al., 2018), improving NLI models' handling of syntactic (Min et al., 2020) and semantic (Poliak et al., 2018a) phenomena, and mitigating gender biases (Zmigrod et al., 2019; Zhao et al., 2018, 2019; Hall Maudslay et al., 2019; Lu et al., 2018). Nie et al. (2020) and Kaushik et al. (2020) also explore augmenting training sets with a human-in-the-loop.

Feature Representations and Robustness. A related body of recent work asks which features are extracted by neural language models, in particular whether SOTA models represent "deeper" syntactic and semantic features. Work in this vein has shown that pretrained language models encode knowledge of syntax, using a range of techniques including supervised "diagnostic classifiers" (Tenney et al., 2019; Conneau et al., 2018; Hewitt and Manning, 2019), classification performance on targeted stimuli (Linzen et al., 2016; Goldberg, 2019), attention maps/visualizations (Voita et al., 2019; Serrano and Smith, 2019), and relational similarity analyses (Chrupała and Alishahi, 2019). Geiger et al. (2019) attempts to quantify how much data a model should need to learn a given deeper feature. We contribute to this literature by asking under what conditions we might expect deeper features to be extracted and used.

Adversarial robustness examines how small perturbations in inputs can cause models to make wrong predictions (Ribeiro et al., 2018; Iyyer et al., 2018; Hsieh et al., 2019; Jia et al., 2019), or to change their output (Alzantot et al., 2018) or internal representations (Hsieh et al., 2019). In NLP, such perturbations often involve word-level (Alzantot et al., 2018; Hsieh et al., 2019; Jia et al., 2019) or sentence-level (Ribeiro et al., 2018; Iyyer et al., 2018) paraphrasing. Ilyas et al. (2019) make a distinction between useful features (that generalize well) and those that are robustly-useful (that generalize well, even if an example is adversarially perturbed). Madry et al. (2017); Athalye et al. (2018) investigate robustness by giving adversaries access to model gradients.

Generalization of Neural Networks A still larger body of work studies feature representation and generalization in neural networks. Mangalam and Prabhu (2019) show that neural networks learn "easy" examples (as defined by shallow ML model performance) before they learn "hard" examples. Zhang et al. (2016) and Arpit et al. (2017) show that neural networks with good generalization performance can memorize noise, suggesting that such models might have an inherent preference to learn more general features. Finally, there is ongoing theoretical work that characterizes the ability of over-parameterized networks to generalize in terms of complexity (Neyshabur et al., 2019) and implicit regularization (Blanc et al., 2019).

7 Conclusion

We propose a framework for simulating the effects of data augmentation in NLP and use it to explore how training on counterexamples impacts model generalization. Our results suggest that adding counterexamples in order to encourage a model to "unlearn" weak features is likely to have the immediately desired effect (the model will perform better on examples that look similar to the generated counterexamples), but the model is unlikely to shift toward relying on stronger features in general. Specifically, in our experiments, the models trained on data augmented with a small number of counterexamples (< 100K) still fail to correctly classify examples which contain only the strong feature. We see also that data augmentation may become less effective as the underlying strong features become more difficult to extract.

References

Aishwarya Agrawal, Dhruv Batra, Devi Parikh, and Aniruddha Kembhavi. 2018. Don't just assume; look and answer: Overcoming priors for visual question answering. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Moustafa Alzantot, Yash Sharma, Ahmed Elgohary, Bo-Jhang Ho, Mani Srivastava, and Kai-Wei Chang. 2018. Generating natural language adversarial examples. *arXiv preprint arXiv:1804.07998*.

Devansh Arpit, Stanislaw Jastrzkebski, Nicolas Ballas, David Krueger, Emmanuel Bengio, Maxinder S. Kanwal, Tegan Maharaj, Asja Fischer, Aaron Courville, Yoshua Bengio, and Simon Lacoste-Julien. 2017. A closer look at memorization in deep networks. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ICML'17, pages 233–242. JMLR.org.

Anish Athalye, Nicholas Carlini, and David Wagner. 2018. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. *arXiv preprint arXiv:1802.00420*.

Guy Blanc, Neha Gupta, Gregory Valiant, and Paul Valiant. 2019. Implicit regularization for deep neural networks driven by an ornstein-uhlenbeck like process. *arXiv preprint arXiv:1904.09080*.

Grzegorz Chrupała and Afra Alishahi. 2019. Correlating neural and symbolic representations of language. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2952–2962, Florence, Italy. Association for Computational Linguistics.

Alexis Conneau, German Kruszewski, Guillaume Lample, Loïc Barrault, and Marco Baroni. 2018. What you can cram into a single $&!#* vector: Probing sentence embeddings for linguistic properties. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2126–2136, Melbourne, Australia. Association for Computational Linguistics.

Ishita Dasgupta, Demi Guo, Andreas Stuhlmüller, Samuel J Gershman, and Noah D Goodman. 2018. Evaluating compositionality in sentence embeddings. *arXiv preprint arXiv:1802.04302*.

Ali Elkahky, Kellie Webster, Daniel Andor, and Emily Pitler. 2018. A challenge set and methods for noun-verb ambiguity. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2562–2572, Brussels, Belgium. Association for Computational Linguistics.

Atticus Geiger, Ignacio Cases, Lauri Karttunen, and Christopher Potts. 2019. Posing fair generalization tasks for natural language inference. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4484–4494, Hong Kong, China. 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. Association for Computational Linguistics.

Yoav Goldberg. 2019. Assessing bert's syntactic abilities. *arXiv preprint arXiv:1901.05287*.

Suchin Gururangan, Swabha Swayamdipta, Omer Levy, Roy Schwartz, Samuel R Bowman, and Noah A Smith. 2018. Annotation artifacts in natural language inference data. *arXiv preprint arXiv:1803.02324*.

Rowan Hall Maudslay, Hila Gonen, Ryan Cotterell, and Simone Teufel. 2019. It's all in the name: Mitigating gender bias with name-based counterfactual data substitution. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5266–5274, Hong Kong, China. Association for Computational Linguistics.

John Hewitt and Christopher D. Manning. 2019. A structural probe for finding syntax in word representations. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4129–4138, Minneapolis, Minnesota. Association for Computational Linguistics.

Yu-Lun Hsieh, Minhao Cheng, Da-Cheng Juan, Wei Wei, Wen-Lian Hsu, and Cho-Jui Hsieh. 2019. On the robustness of self-attentive models. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1520–1529.

Andrew Ilyas, Shibani Santurkar, Dimitris Tsipras, Logan Engstrom, Brandon Tran, and Aleksander Madry. 2019. Adversarial examples are not bugs, they are features. In *Advances in Neural Information Processing Systems*, pages 125–136.

Mohit Iyyer, John Wieting, Kevin Gimpel, and Luke Zettlemoyer. 2018. Adversarial example generation with syntactically controlled paraphrase networks. *arXiv preprint arXiv:1804.06059*.

Robin Jia and Percy Liang. 2017. Adversarial examples for evaluating reading comprehension systems. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2021–2031. Association for Computational Linguistics.

Robin Jia, Aditi Raghunathan, Kerem Göksel, and Percy Liang. 2019. Certified robustness to adversarial word substitutions. *arXiv preprint arXiv:1909.00986*.

Divyansh Kaushik, Eduard Hovy, and Zachary Lipton. 2020. Learning the difference that makes a difference with counterfactually-augmented data. In *International Conference on Learning Representations*.

Tal Linzen, Emmanuel Dupoux, and Yoav Goldberg. 2016. Assessing the ability of LSTMs to learn syntax-sensitive dependencies. *Transactions of the Association for Computational Linguistics*, 4:521–535.

Nelson F. Liu, Roy Schwartz, and Noah A. Smith. 2019. Inoculation by fine-tuning: A method for analyzing challenge datasets. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2171–2179, Minneapolis, Minnesota. Association for Computational Linguistics.

Kaiji Lu, Piotr Mardziel, Fangjing Wu, Preetam Amancharla, and Anupam Datta. 2018. Gender bias in neural natural language processing. *arXiv preprint arXiv:1807.11714*.

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. 2017. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*.

Karttikeya Mangalam and Vinay Uday Prabhu. 2019. Do deep neural networks learn shallow learnable examples first?

R Thomas McCoy, Ellie Pavlick, and Tal Linzen. 2019. Right for the wrong reasons: Diagnosing syntactic heuristics in natural language inference. *arXiv preprint arXiv:1902.01007*.

Junghyun Min, R. Thomas McCoy, Dipanjan Das, Emily Pitler, and Tal Linzen. 2020. Syntactic data augmentation increases robustness to inference heuristics. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, Seattle, Washington. Association for Computational Linguistics.

Behnam Neyshabur, Zhiyuan Li, Srinadh Bhojanapalli, Yann LeCun, and Nathan Srebro. 2019. The role of over-parametrization in generalization of neural networks. In *International Conference on Learning Representations*.

Yixin Nie, Adina Williams, Emily Dinan, Mohit Bansal, Jason Weston, and Douwe Kiela. 2020. Adversarial NLI: A new benchmark for natural language understanding. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4885–4901, Online. Association for Computational Linguistics.

Adam Poliak, Aparajita Haldar, Rachel Rudinger, J. Edward Hu, Ellie Pavlick, Aaron Steven White, and Benjamin Van Durme. 2018a. Collecting diverse natural language inference problems for sentence representation evaluation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 67–81, Brussels, Belgium. Association for Computational Linguistics.

Adam Poliak, Jason Naradowsky, Aparajita Haldar, Rachel Rudinger, and Benjamin Van Durme. 2018b. Hypothesis only baselines in natural language inference. *arXiv preprint arXiv:1805.01042*.

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.

Rachel Rudinger, Jason Naradowsky, Brian Leonard, and Benjamin Van Durme. 2018. Gender bias in coreference resolution. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 8–14, New Orleans, Louisiana. Association for Computational Linguistics.

Sofia Serrano and Noah A. Smith. 2019. Is attention interpretable? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2931–2951, Florence, Italy. Association for Computational Linguistics.

Ian Tenney, Patrick Xia, Berlin Chen, Alex Wang, Adam Poliak, R Thomas McCoy, Najoung Kim, Benjamin Van Durme, Sam Bowman, Dipanjan Das, and Ellie Pavlick. 2019. What do you learn from context? probing for sentence structure in contextualized word representations. In *International Conference on Learning Representations*.

Elena Voita, David Talbot, Fedor Moiseev, Rico Sennrich, and Ivan Titov. 2019. Analyzing multi-head self-attention: Specialized heads do the heavy lifting, the rest can be pruned. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5797–5808, Florence, Italy. Association for Computational Linguistics.

Elena Voita and Ivan Titov. 2020. Information-theoretic probing with minimum description length. *arXiv preprint arXiv:2003.12298*.

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. 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 1112–1122. Association for Computational Linguistics.

Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. 2016. Understanding deep learning requires rethinking generalization. *CoRR*, abs/1611.03530.

Jieyu Zhao, Tianlu Wang, Mark Yatskar, Ryan Cotterell, Vicente Ordonez, and Kai-Wei Chang. 2019. Gender bias in contextualized word embeddings. *arXiv preprint arXiv:1904.03310*.

Jieyu Zhao, Tianlu Wang, Mark Yatskar, Vicente Ordonez, and Kai-Wei Chang. 2018. Gender bias in coreference resolution: Evaluation and debiasing methods. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 15–20, New Orleans, Louisiana. Association for Computational Linguistics.

Ran Zmigrod, Sebastian J. Mielke, Hanna Wallach, and Ryan Cotterell. 2019. Counterfactual data augmentation for mitigating gender stereotypes in languages with rich morphology. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1651–1661, Florence, Italy. Association for Computational Linguistics.

A Ablations

We fix most system parameters in our experiments; Here we show that the trends hold under other parameters. We only use one random seed and one feature (`adjacent-dupl`). See Figures 5-8. Like Figures 2 and 3, accuracies are for a given number of counterexamples of one type, averaged over all values of counterexamples of the other type.

B De-Aggregated Results

We de-aggregate the results over the number of counterexamples and inspect them for each feature. See Figures 9-12.

C More Accuracies

Figures 13-15 show each accuracy metric across all combinations of # *strong-only* examples $\times$ # *weak-only* examples for the various features.

D More Regression Results

See the additional regression results in Tables 4 and 5.

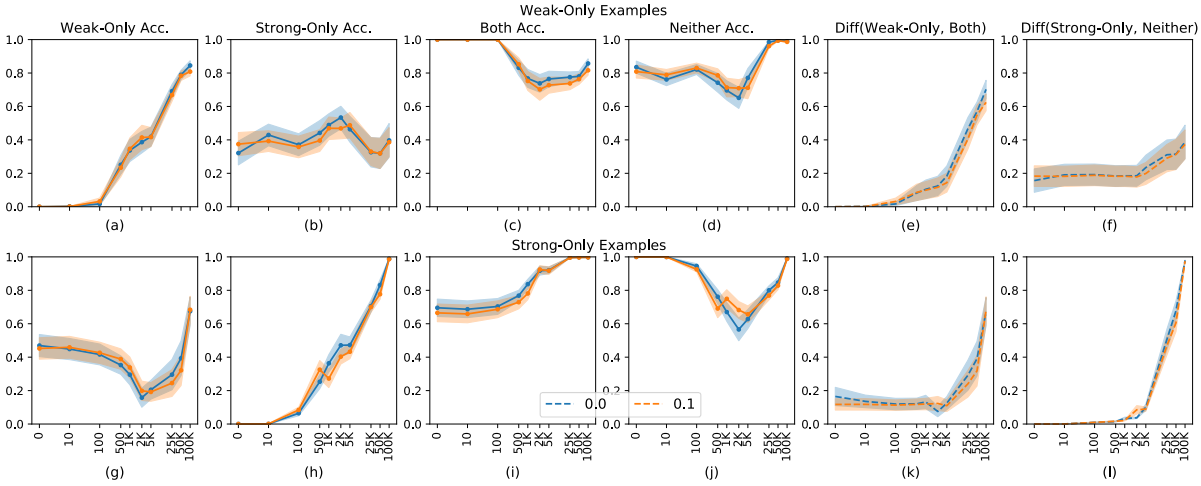

Figure 5: Dropout. We test dropout of 0 (original) and 0.1. Dropout is added after the first linear layer in the decoder.

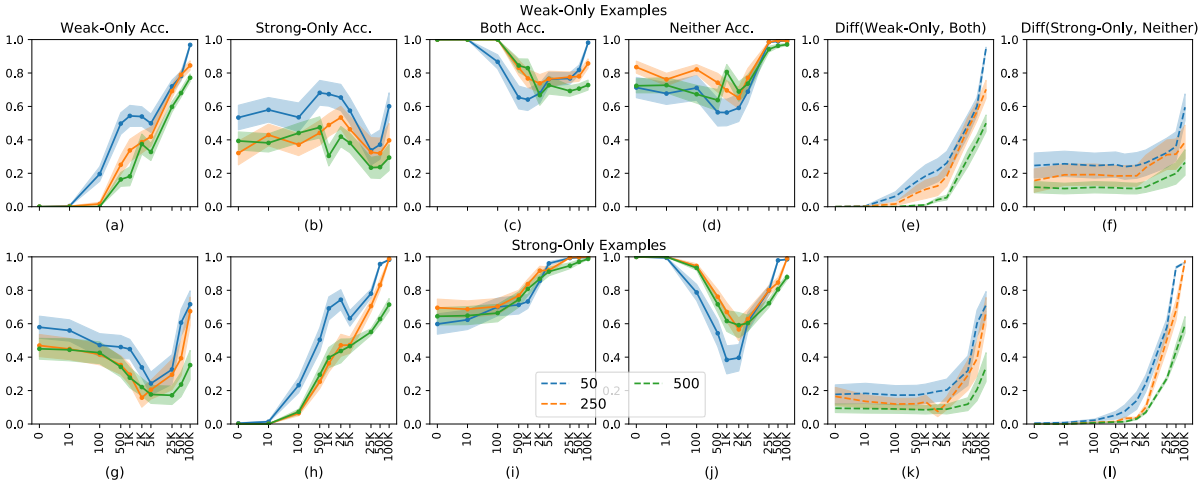

Figure 6: Embedding and hidden size. We test embedding and hidden sizes of 50, 250 (original), and 500.

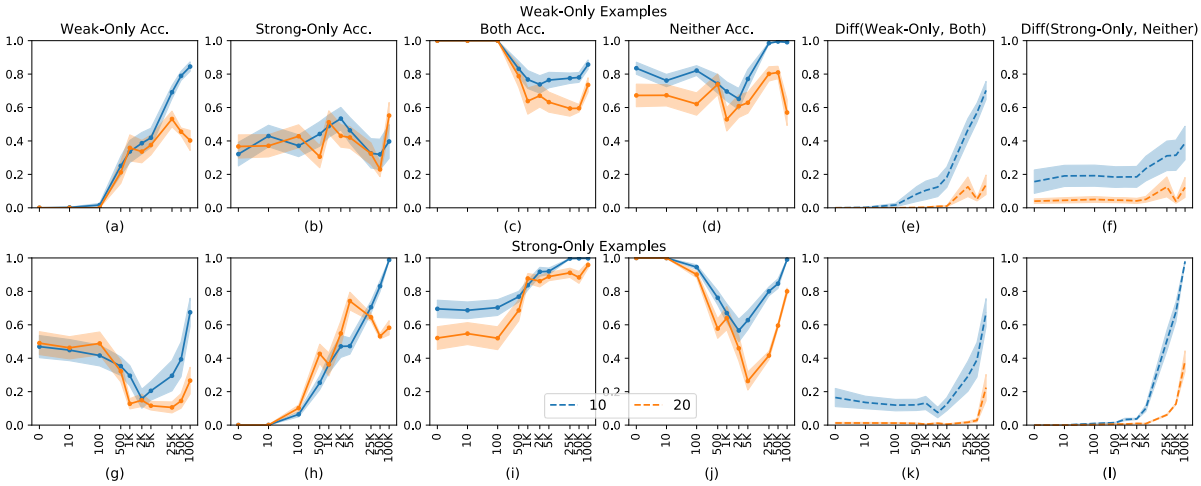

Figure 7: Sequence length. We test sequence lengths of 10 (original) and 20.

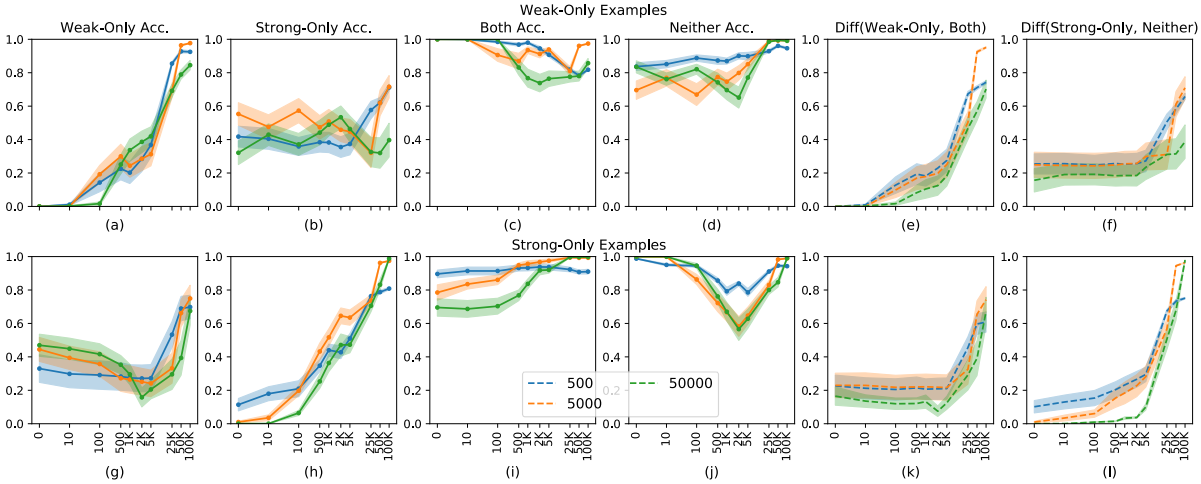

Figure 8: Vocab size. We test vocabulary sizes of 500, 5K, and 50K (original).

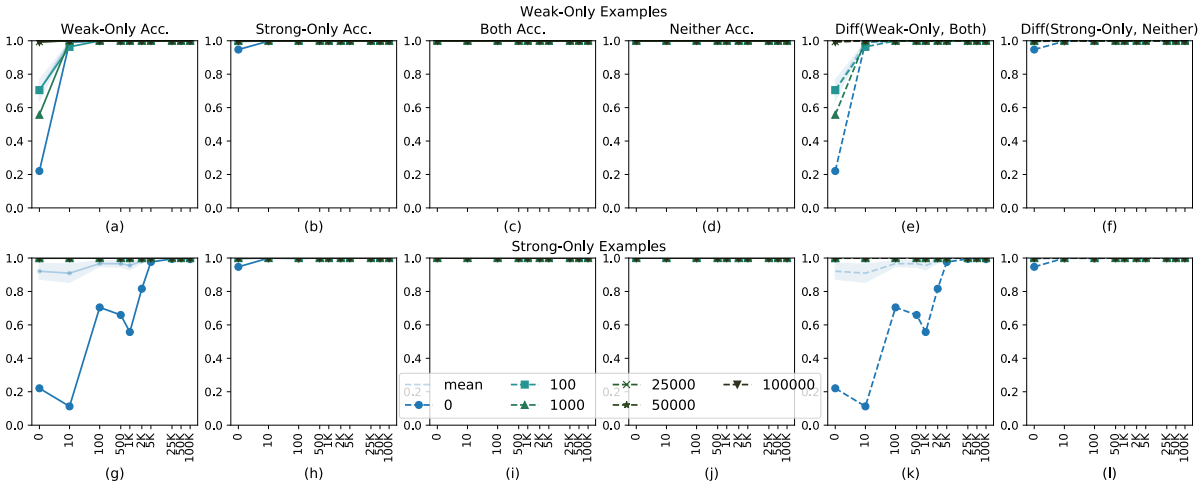

Figure 9: Curves for `contains-1` where the number *weak-only* examples varies for different settings of *strong-only* examples (top row) and vice versa (bottom row).

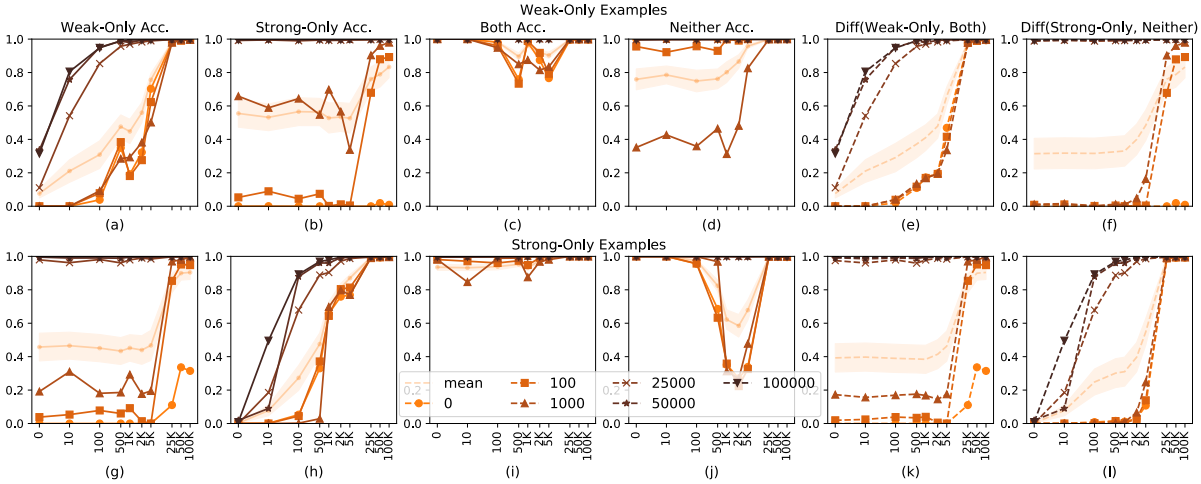

Figure 10: Curves for `prefix-dupl` where the number *weak-only* examples varies for different settings of *strong-only* examples (top row) and vice versa (bottom row).

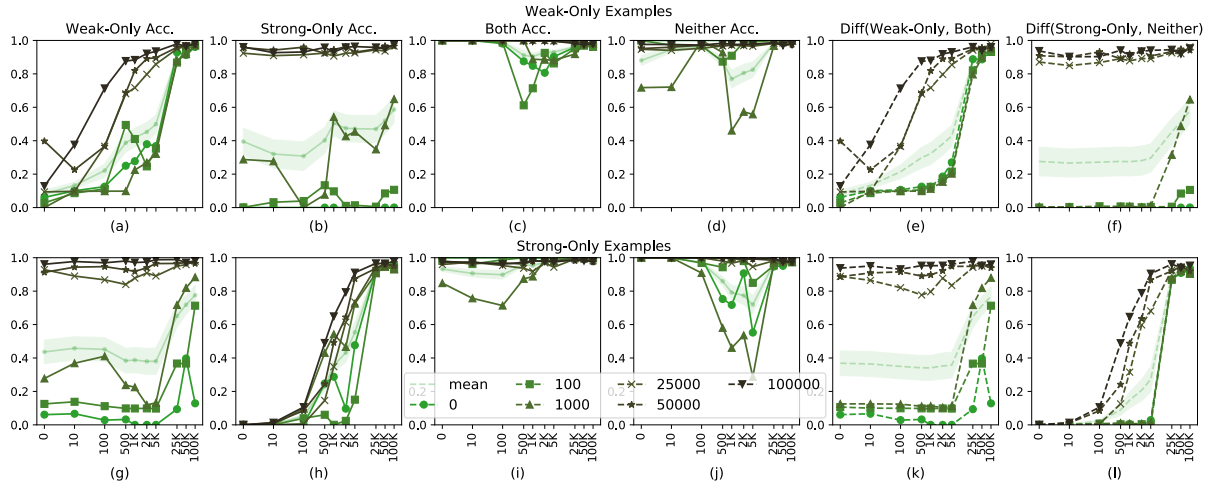

Figure 11: Curves for `first-last` where the number *weak-only* examples varies for different settings of *strong-only* examples (top row) and vice versa (bottom row).

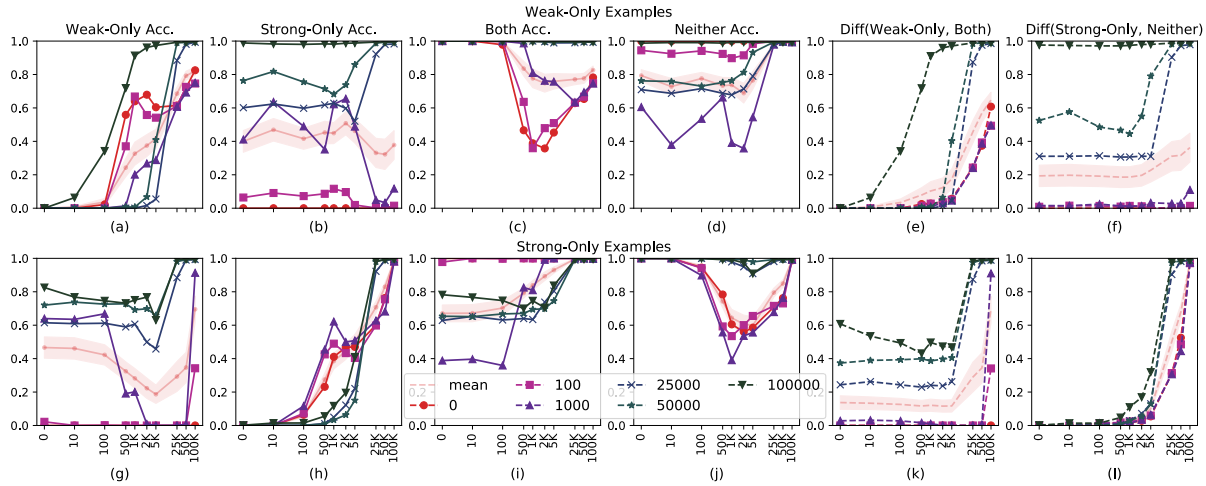

Figure 12: Curves for `adjacent-dupl` where the number *weak-only* examples varies for different settings of *strong-only* examples (top row) and vice versa (bottom row).

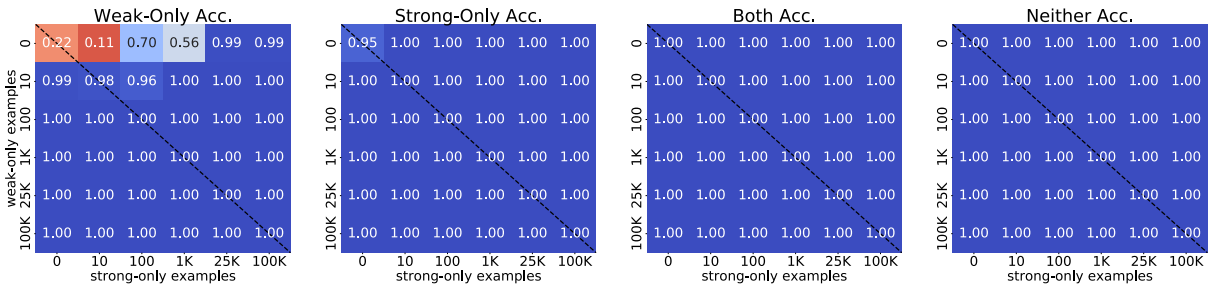

Figure 13: Accuracies for `contains-1` with # *weak-only* $\times$ # *strong-only* examples.

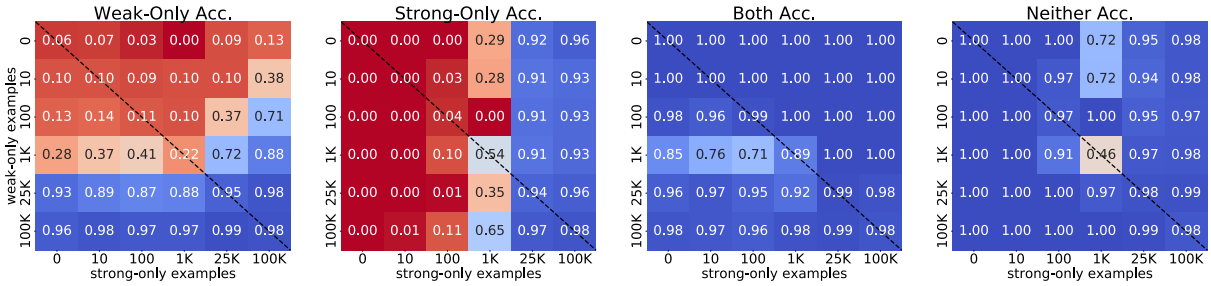

Figure 14: Accuracies for `first-last` with # *weak-only* × # *strong-only* examples.

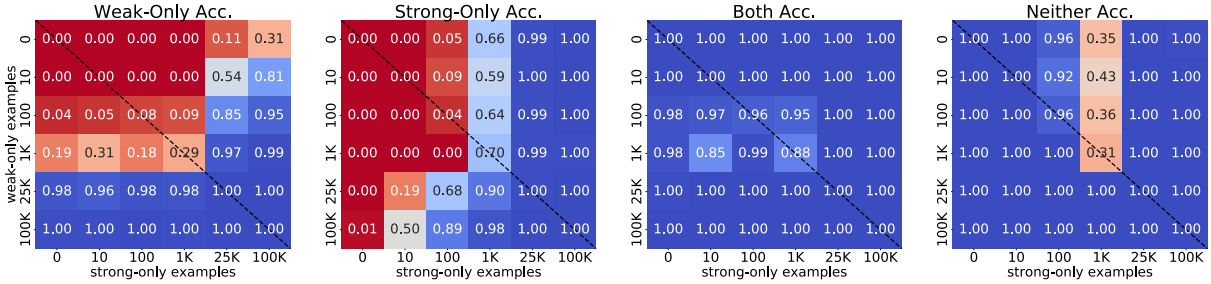

Figure 15: Accuracies for `prefix-dupl` with # *weak-only* × # *strong-only* examples.

		weak-only acc. $R^2 = 0.82$		*strong-only* acc. $R^2 = 0.84$	
	σ	β	p	β	p
contains-1		0.89	0.00*	0.90	0.00*
prefix-dupl		−0.07	0.01*	−0.01	0.60
first-last		−0.26	0.00*	−0.43	0.00*
adjacent-dupl		−0.58	0.00*	−0.48	0.00*
log strong-only ex.	1.5	−0.22	0.00*	−0.15	0.00*
pfx-dup		0.28	0.00*	0.86	0.00*
first-last		0.15	0.00*	0.83	0.00*
adj-dup		−0.10	0.00*	0.75	0.00*
log weak-only ex.	1.5	−0.06	0.04*	−0.05	0.11
pfx-dup		0.62	0.00*	0.19	0.00*
first-last		0.58	0.00*	0.16	0.00*
adj-dup		0.54	0.00*	−0.05	0.17
log total ex.	0.1	−0.13	0.13	−0.09	0.26
I(label, weak)	0.3	−0.61	0.00*	−0.25	0.00*
$P(0 \mid \neg weak)$	0.2	−0.28	0.06	0.64	0.00*
$P(1 \mid weak)$	0.2	0.21	0.16	−0.61	0.00*
$P(0)$	0.1	−0.44	0.01*	0.81	0.00*

Table 4: Regression results with terms for the interaction between feature type and number of examples added.

		weak-only acc. $R^2 = 0.74$		*strong-only* acc. $R^2 = 0.70$		*both* acc. $R^2 = 0.39$		*neither* acc. $R^2 = 0.40$	
	σ	β	p	β	p	β	p	β	p
contains-1		0.89	0.00*	0.90	0.00*	0.43	0.00*	0.49	0.00*
prefix-dupl		−0.07	0.03*	−0.01	0.71	0.17	0.00*	−0.14	0.00*
first-last		−0.26	0.00*	−0.43	0.00*	0.10	0.02*	0.03	0.52
adjacent-dupl		−0.58	0.00*	−0.48	0.00*	−0.73	0.00*	−0.39	0.00*
log strong-only ex.	1.5	−0.14	0.00*	0.46	0.00*	0.28	0.00*	−0.72	0.00*
log weak-only ex.	1.5	0.37	0.00*	0.03	0.25	−0.44	0.00*	−0.11	0.00*
log total ex.	0.1	−0.15	0.16	−0.08	0.48	−0.32	0.05	−0.77	0.00*
I(label, weak)	0.3	−0.61	0.00*	−0.25	0.00*	−0.22	0.04*	−0.83	0.00*
$P(0\|\neg weak)$	0.2	−0.27	0.13	0.63	0.00*	0.22	0.42	0.79	0.00*
$P(1\|weak)$	0.2	0.22	0.22	−0.63	0.00*	0.29	0.28	0.22	0.40
$P(0)$	0.1	−0.45	0.03*	0.82	0.00*	0.13	0.70	0.14	0.65

Table 5: Regression results for predicting all accuracy metrics. Fourth section are variables meant to control for confounding effects of adding counterexamples. $I()$ is mutual information between the presence of the weak feature and label according to training data; $P()$ are label skews according to training.