Minimal Edit Counterfactual Data Augmentation for Model Robustness

He He, Sam Bowman, Nitish Joshi, Johnny Ma

New York University CILVR

11-4-20

Introduction

- Large-scale LMs for NLU have known, systematic gaps in data understanding (SNLI never-contradiction).
- Minimal Edit expert contrast sets can "fill" these gaps near local decision boundaries at **evaluation** time.
- Not clear that crowd sourcing counterfactuals can improve model generalizability at **training** time.

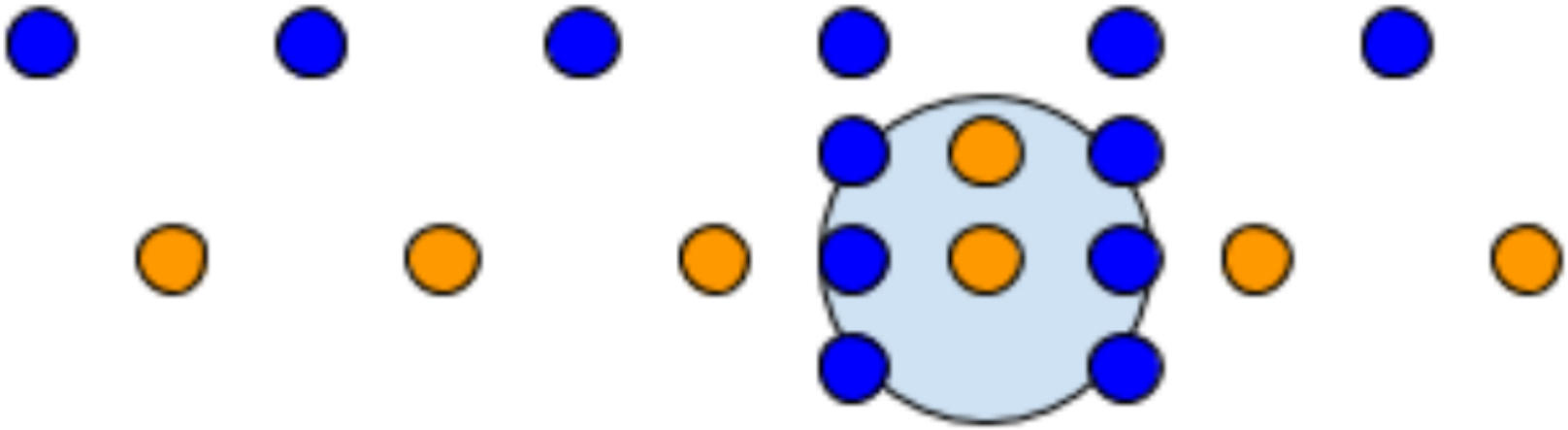


Figure 1: Local ball around test instance.

Structured Contrast Set Generation

- CLARE (Li *et al.* 2020) generates MP candidates with USE, then evaluates success of label attack.
  - Replacing and merging nouns are most successful attack types.
- LIT (Liu *et al.* 2020) use ERG/ACE parser as backbone to transform syntax and semantics.
  - Transforms 20% of SNLI instances, English only.
- BAE (Garg *et al.* 2020) generate more fluent and coherent adversarial attacks. They use pre-trained BERT-MLM to replace and insert a masked token. USE+POS for similarity, powerful model attack and high human rated fluency.
  - They identify "token importance" by computing decrease in model same class probability when deleting a given token.

Conditioned Language Model Contrast Set Generation

- MiCE (Ross *et al.* 2020) assert that contrastive edits are "effective explanations" of model behavior. They use a two step approach:
  1. Fine-tune T5 using masked span infilling task, with additional pre-pended gold/predicted label.
  2. Generate over masked consecutive spans of tokens with *desired contrast* label prepended. Continue infilling until label flips.
- Uses T5 generic text-to-text pre-trained model. Measure/compare MiCE vs. human success with flip rate, minimality, and fluency. **Gradient Masking** important tokens in Stage 1 is significant, conditioning on **target labels** also helps flips.

Conditioned Language Model Contrast Set Generation

- GYC (Madaan *et al.* 2020) draws counterfactuals from $p(y|x, \textit{condition})$ from latent space perturbations of GPT-2 key-value layer history matrix $\tilde{H}_t = H_t + \Delta H_t$, maximizing $\log(p(c|y)) - \sum_t D_{KL}[p(y_t|y_{t-1}, \tilde{H}_{t-1})||p(y_t|y_{<t})]$.
- Captures proximity (reconstruction), reward for high probability on desired class, and diversity (entropy) with multiple losses $\mathcal{L} = \lambda_r \mathcal{L}_r + \lambda_p \mathcal{L}_p + \lambda_H \mathcal{L}_H$
- Use various metrics for diversity, tree-edit distance, content preservation, combined with human evaluation.

Automatic Counterfactual Generation

- Polyjuice (Wu *et al. 2020*) produces high-quality, controllable, and minimal counterfactual sets with low annotation cost while improving model generalization.
- Fine-tune GPT-2 using ILM framework 6 existing counterfactual pairs dataset. Extract "control code" from masked data using parse tree/POS, sample 10,000 per code (8 in total), concat x + code $\rightarrow \hat{x}$ (blanked CF) as task. They release model for possible comparison!
- Mask data slices $P(Y|s_i, X)$ of "interest," generate using code + masked x. For multiple $\tilde{x}$ generations, take within-generated distanced CFs, ensure one flips label.
- Important to mask **known error/high attention/SHAP importance** spans and target specific codes to improve model (RoBERTA) generalizability.

Tables

Dataset	negation	quantifier	leixcal	resemantic	insert	delete	restructure	shuffle	global
CAD	3,456	457	10,650	4,634	2,169	2,162	234	84	3,756
Contrast	336	436	1,607	1,291	589	586	275	149	877
HANS	50	0	0	0	3,926	3,926	494	1,602	2
ParaNMT	2,797	825	10,000	10000	6,442	6,205	5,136	1,417	10,000
PAWS	81	1,815	10,000	10000	3,630	3,403	4,551	10,000	10,000
WinoGrande	3,011	94	10,000	6,927	120	124	453	65	3184
Crawled	0	0	5,000	0	5,000	5,000	0	108	5,000
Total	9,731	3,627	47,257	32,852	21,876	21,406	11,143	13,425	32,819

Table 1: Control Codes extracted from counterfactual datasets.

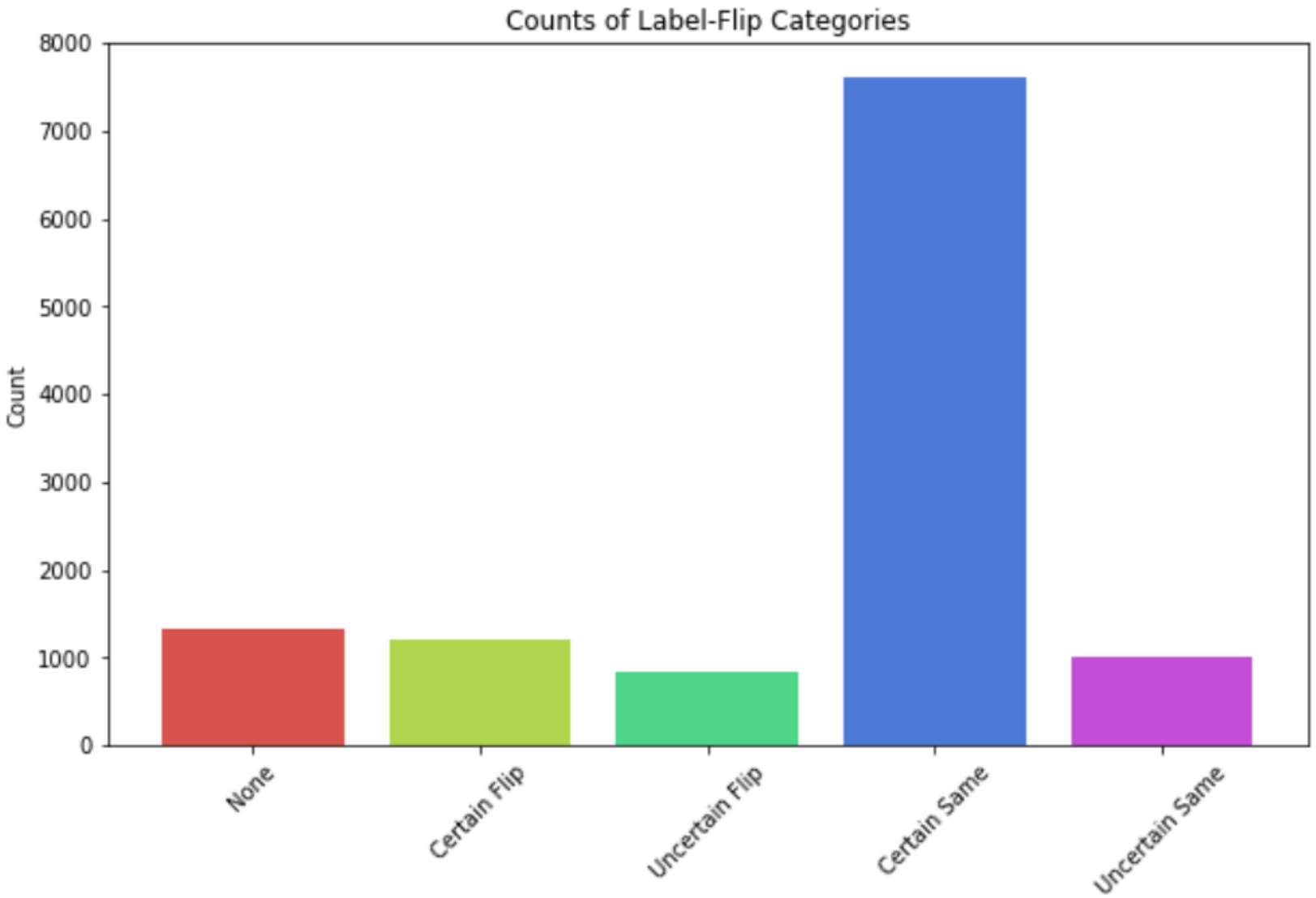
Task	Dev.	Orig. set	Contrast set	Consistency
Sentiment	94.3	93.8	84.9(-8.9)	76.1
NLI	86.5	91.6	72.3(-19.3)	56.4
QQP	91.7	87.5	75.3(-12.2)	61.1

Table 2: Contrast Set Model performance for model evaluation.

MNLI and BART

- **Editor**: Fine-Tune Fairseq BART for span mask-infill of hypothesis (Denoising, $\lambda = 3$) on concatenated MNLI sentence pairs.
  - Prem + [SEP] + masked Hypo $\rightarrow$ Prem + [SEP] + Hypo
- **Generator**: Mask N-Grams around K POS content words, infill with beam search sampling. Filter out $\tilde{x} = x$ and repeat $\tilde{x}$, generates $[0, K * N]$ examples per x.
- **Metrics**: For each x $\tilde{x}$ pair, calculate similarity metrics (BERT-Score, W2V) for replaced tokens and entire sequence.
- **Classification**: Using MNLI fine-tuned RoBERTA, calculate class probabilities for x and $\tilde{x}$ pair. Assign categories based on **label flip** $\times$ **uncertainty**. Manually assess model errors.

Label Flip Counts



Results Summary

Model is Right % / 25	Certain Flip	Uncertain Flip	Certain Same	Uncertain Same
Johnny sampling from high BERT-Score	88%	44% (40%)	96%	72% (20%)
Nitish Random samples	84%	56%	84%	60%

Table 3: Manual labeling of generated examples.

Original ->	Contradiction	Neutral	Entailment
Contradiction	2126	216	160
Neutral	173	3435	188
Entailment	162	230	3008

Table 4: Distribution of label flips.

Editor Improvements

- Use a more flexible span mask-infill model.
  - T5 (MiCE) or ILM (PJ) generic source-to-target pre-training.
  - Can move away from *span* requirement and create multiple masks. Better for longer sequences.
- Condition on gold/predicted class label (MiCE) or masking behavior (PJ control code) for more guided control during generation.
- Evaluate success of fine-tuning editors using PPL, GPT-2 log-probs (PJ), or downstream manual categories counts.

Generator Improvements

- More targeted masking protocol:
  - Tokens/spans with high gradient w.r.t class and embeddings from predictor (MiCE).
  - Remove tokens and observe class probability change, sort by "important" tokens (BAE). Good for black box narrative.
  - Use data slice protocol from Chen *et al.* 2019.
- Guide generation to desired behavior with prompted control codes or desired class label.
  - Potentially disrupts our "shotgun approach" narrative.
- Fix number of $\tilde{x}$ per x, define specific strategies and compare with human generated or PJ generated.
- Explore sampling strategies: top-p, top-k, temperature, for diversity and quality trade-off.

Metrics Improvements

- **Minimality** measurements:
  - Levenshtein Distance (MiCE).
  - Semantic Tree-Edit Distance (GYC, PJ).
  - Other embeddings or LM based measurements.
- **Fluency** measurements:
  - Masked-LM loss.
  - Mostly human evaluation.
- **Diversity** measurements:
  - Self-BLEU (PJ, GYC) over all $\tilde{x}$ for a x.
  - Entropy of logits, distribution comparisons.
- Can find patterns in model shifting
- Asking humans to evaluate these is generally accepted.

Classification Improvements

- Readjust our pre-defined **label flip** $\times$ **uncertainty** definitions.
- Focus more on "fragile" x along decision boundaries.
- Try different data sources and compare model behavior *within* datasets, *between* editor/generator set-ups.

Questions

- Are we interested in generating counterfactuals, or minimally edited new training data?
- Is augmenting data with *contrasting* examples or *model error* examples more helpful for *training* generalization?
- Can we make improvements on the human-in-the-loop methodology?
- How much do we want to guide the generation process?
- Worth it to re-frame the problem as learning a generator with loss constraints, such as in GYC?