# Scale-invariant-Fine-Tuning for Improved Model Generalization

Luheng Wang, Yash Bharti, Yilun Kuang

New York University

May 18, 2021

# Overview

# Topic

- What is SiFT?
- Perturbation (SMART by [Jia+20])
- Layer Normalization

# Data

- CoLA [1](NYU)
- Twitter Sentiment Dataset [2] (Kaggle)
- UCI Sentiment Dataset [3] (UCI)




[1] https://nyu-mll.github.io/CoLA/

[2] https://www.kaggle.com/arkhoshghalb/twitter-sentiment-analysis-hatred-speech

[3] http://archive.ics.uci.edu/ml/datasets

# Smoothness-inducing Adversarial Regularizer

Consider the optimization objective with a standard cross entropy loss $\mathcal{L}(\theta)$ and a regularizer:

$$\hat{\theta} = \underset{\theta}{\operatorname{argmin}}\, \mathcal{L}(\theta) + \lambda_s \mathcal{R}_s(\theta). \tag{1}$$

The Smoothness-inducing Adversarial Regularizer $\mathcal{R}_s(\theta)$ is defined as

$$\mathcal{R}_s(\theta) = \frac{1}{n} \sum_{i=1}^{n} \max_{||\tilde{x}_i - x_i|| \leq \epsilon} l_s(f(\tilde{x}_i; \theta), f(x_i; \theta)). \tag{2}$$

where $l_s$ is the symmetrized KL-divergence [**p1**]

$$l_s(P, Q) = \mathcal{D}_{\mathrm{KL}}(P||Q) + \mathcal{D}_{\mathrm{KL}}(Q||P). \tag{3}$$

# Code Implementation

```
loss_fct = CrossEntropyLoss()
regular_loss = loss_fct(preds,labels)
loss_list = [regular_loss]
```

Listing 1: cross entropy loss $\mathcal{L}(\theta)$

```
MODE="SIFT"
normalise = True if MODE == "SIFT" else False
noised_embeddings = noise(embed,normalize=normalise)
adv_logits = model.predict(noised_embeddings)
adv_loss = stable_kl(preds, adv_logits)
```

Listing 2: calculate $\mathcal{R}_s(\theta)$

```
loss_list.append(adv_loss)
loss = sum(loss_list)
total_train_loss += loss.item()
loss.backward()
```

Listing 3: minimize $\mathcal{L}(\theta) + \mathcal{R}_s(\theta)$

# Cola Dataset

| | sentence_source | label | label_notes | sentence |
|---|---|---|---|---|
| 4689 | ks08 | 1 | NaN | Loren was relied on by Pavarotti and Hepburn b... |
| 404 | bc01 | 1 | NaN | His book is nice. |
| 7363 | sks13 | 1 | NaN | I put the book on the desk on Sunday. |
| 2291 | l-93 | 0 | * | David constructed the bricks into a house. |
| 8043 | ad03 | 1 | NaN | I haven't left yet |
| 7859 | ad03 | 1 | NaN | He'll no can do it, will he? |
| 3380 | l-93 | 1 | NaN | The horse jumped into the stream. |
| 2812 | l-93 | 1 | NaN | Paula swatted at the fly. |
| 2285 | l-93 | 0 | * | The pasture is herding with cattle. |
| 8229 | ad03 | 1 | NaN | I have been flying helicopters for years. |

# UCI Dataset and Twitter Dataset

| | Sentence | Label |
|---|---|---|
| 0 | A very, very, very slow-moving, aimless movie ... | 0 |
| 1 | Not sure who was more lost - the flat characte... | 0 |
| 2 | Attempting artiness with black & white and cle... | 0 |
| 3 | Very little music or anything to speak of. | 0 |
| 4 | The best scene in the movie was when Gerardo i... | 1 |

# Testing Methodology

## CoLA Baselines

All the baselines were first trained on a cola dataset. Then the Matthews Correlation Coefficient was calculated on the test set due to CoLAs skewed nature

## UCI Dataset

Our technique to check on generalisation. The in domain training set was on Twitter. We achieved high results on the out of domain set of UCI Sentiment Analysis: IMDB, AMAZON, YELP reviews, with very little training.

# Results with BERT

| **Model** | **CoLA** | **UCI 10** |
|---|---|---|
| Plain BERT | 0.52 | 0.55 |
| BERT w SMART | 0.51 | 0.70 |
| BERT w SIFT | 0.53 | 0.80 |

Table: Results with the BERT Model

# Results with DeBERTA

| **Model** | **CoLA** | **UCI 10** |
|---|---|---|
| Plain DeBERTA | 0.54 | 0.56 |
| DeBERTA w SMART | 0.57 | 0.80 |
| DeBERTA w SIFT | 0.58 | 0.91 |

Table: Results with the DeBERTA Model

# Experiment Results

- Successfully achieved higher results for SiFT compared to SMART and plan deberta, and bert
- Higher Matthews Correlation Coefficient for Cola on both BERT and DeBERTA.
- Higher accuracy on the UCI Sentiment Analysis dataset after training for 10 percent dataset.
- Effectively proved that SiFT technique is better for generalization than the SMART algorithm and the plain models.

# References

SMART: Robust and Efficient Fine-Tuning for Pre-trained Natural Language Models through Principled Regularized Optimization

*Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*