Meta-Learning for Sample Reweighting using Sequence Models: A Natural Language Inference Case Study

Ethan A. Chi
Stanford NLP
ethanchi@cs.stanford.edu

Shikhar Murty
Stanford NLP
smurty@cs.stanford.edu

Sidd Karamcheti
Stanford NLP / ILIAD
skaramcheti@cs.stanford.edu

Extended Abstract

Natural Language Inference (NLI) (Bowman et al., 2015; Williams et al., 2018) is a sequence modeling task on the critical path to natural language understanding (NLU). The task itself can be expressed simply, reducing down to a simple 2 or 3-way classification problem: given a *premise* (e.g., "the doctor near the actor danced") and a *hypothesis* (e.g., "the doctor danced"), predicted whether or not the premise *1) entails, 2) contradicts, or 3) is neutral to* the given hypothesis — in this case, the answer is (1), "the doctor near the actor danced" *entails* that "the doctor danced."

However, since its inception, NLI has been plagued with problems (Gururangan et al., 2018; McCoy et al., 2019; Belinkov et al., 2019); critically, models trained on various NLI datasets often learn brittle, spurious correlations, or overfit to artifacts of their training datasets, hurting generalization to standard validations sets, let alone more challenging out-of-distribution challenge sets. This problem is partially due to poor quality of crowdsourced data, as workers often generate examples that are simple or that are heavily biased, a result of using simple heuristics to produce label examples (e.g., adding the word "not" to a premise to build a "contradiction" example).

Despite this, NLI remains to be a critical testbed for evaluating sequence modeling architectures and existing natural language understanding models, and solving (or even slightly improving the robustness of models trained on NLI) would not only bring us one step closer to better natural language understanding, but would give us a better means of calibrating existing general-purpose language models — like BERT, GPT-2/3, or T5 (Devlin et al., 2019; Radford et al., 2019; Raffel et al., 2019). To this end, *we present an approach for building more robust sequence models for natural language inference by leveraging meta-learning for sample reweighting.*

Related work (Ren et al., 2018) uses inner-loop meta-learning with simple convolutional-neural network architectures to leverage a *clean validation set* that they backprogagate through to learn weights for different examples in the training set; intuitively, the idea is to upweight training examples that improve performance relative to this validation set, and downweight all other examples. On simple experiments on the MNIST and CIFAR-100 datasets with artificially generated label noise and label imbalance, they show that their inner-loop meta-learning framework can lift out the "good" examples, and improve validation performance relative to models trained with a standard maximum likelihood objective.

We scale up their approach in two ways: first, by looking at meta-learning for sample reweighting as applied to more complex classes of models – namely recurrent neural networks (RNNs) (Hochreiter and Schmidhuber, 1997; Cho et al., 2014) and large, pretrained language models such as BERT (Devlin et al., 2019) — the latter a large-scale model of a scale that has to our knowledge, not been used in the context of inner-loop (2nd order) meta-learning. Second, in addition to evaluating on NLI in the context of artifically injected noise in the form of label noise and label imbalance, we look to realistic challenge sets like HANs (McCoy et al., 2019) to serve as the validation set for our reweighting NLI training set. The hope is by using these "natural" challenge sets, we can prevent our NLI models from learning these spurious correlations or fitting to these artifacts, thereby showing robust generalization and better performance overall.

Our results show that not only does meta-learning for sample reweighting boost NLI performance with bag-of-word models, RNNs, and BERT for NLI in the artifical label noise and imbalance setting, but also when using the more "natural" and targeted HANS validation set. Yet, our investigation in applying 2nd-order methods to RNNs and BERT using the `higher` library (Grefenstette et al., 2019) discovered a rather insidious memory-leak bug, limiting our initial results with RNNs and BERT; namely, we were only able to run inner-loop-based sample reweighting by fine-tuning the last few layers. However, by re-deriving the math behind the approach used by Ren et al. (2018), we found a way to express the inner loop objective as a simple dot product between gradients. This allowed us to scale sample-reweighting to the full training of our RNN and BERT-based models, reinforcing our earlier results.

Meta-Learning for Sample Reweighting using Sequence Models: A Natural Language Inference Case Study

Ethan A. Chi
Stanford NLP
ethanchi@cs.stanford.edu

Shikhar Murty
Stanford NLP
smurty@cs.stanford.edu

Sidd Karamcheti
Stanford NLP / ILIAD
skaramcheti@cs.stanford.edu

Abstract

Natural Language Inference (NLI) is a sequence modeling task on the critical path towards natural language understanding. However, NLI datasets and models are plagued by a series of problems; critically most datasets reflect significant annotator bias and as such, models pick up on spurious correlations and overfit to artifacts, hurting generalization to standard validation sets, let alone hard out-of-domain generalization challenge sets. In this work, we present an approach for guarding our sequence models against these spurious correlations and artifacts, and generally improving the generalization ability of NLI models through the use of *meta-learning for sample reweighting*. Given a standard NLI training dataset, and a clean, or representative challenge set (capturing some desired phenomena of NLI models), we use meta-learning to reweight training samples based on those that improve performance on this held-out set; intuitively, we learn to upweight examples that allow for better performance relative to the validation set, and downweight others. Our results across a wide variety of data corruption schemes and models show sample reweighting is a viable approach, boosting performance compared to models trained with a standard maximum likelihood objective. In addition, we demonstrate a theoretical result that the previously described meta-learning reweighting method we use reduces to an influence function-like dot product on gradients, leading to significant improvements in train compute use. [1]

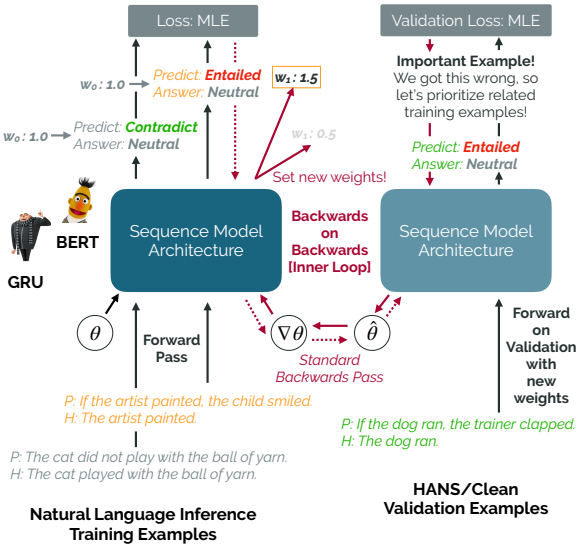

Figure 1: Proposed Approach. We explore *meta-learning for sample reweighting* with multiple sequence models (GRU-Recurrent Neural Networks, BERT) for the task of Natural Language Inference (NLI). We first perform a full forward-backward pass with a batch of training data to produce an updated set of weights (after gradient descent). We use these new weights to perform a forward pass with a "challenge" or "clean" validation batch that captures some phenomena we want of our model — in this case we use the HANS (McCoy et al., 2019) dataset to prevent our models from learning spurious correlations. We backpropagate through the update (backwards-on-backwards pass — inner loop style meta-learning) to *reweight* examples in our training set, improving generalization ability.

1 Introduction

At the core of evaluating progress in natural language understanding is the task of Natural Language Inference (NLI) (Bowman et al., 2015; Williams et al., 2018; Nie et al., 2019). Expressed simply, NLI is a classification task; given two sentences — a *premise* and a *hypothesis* — predict whether the premise *entails*, contradicts, or *is neutral to* the given hypothesis. While simple in framing, NLI is versatile and expressive; broad tasks across natural language processing (NLP) can be framed as NLI problems (Dagan et al., 2006). For example, general question-answering can be expressed as NLI (Demszky et al., 2018). NLI is so powerful and general a task that it is represented as a staple in multiple NLP benchmarks, such as GLUE (Wang et al., 2019b), SuperGLUE (Wang et al., 2019a), and DecaNLP (McCann et al., 2018).

Despite its ubiquity, NLI is plagued with problems, stemming from *annotator bias* (Gururangan et al., 2018; McCoy et al., 2019; Belinkov et al., 2019); due to the way NLI datasets are collected—often leveraging crowdworkers with minimal quality control or

[1] We release our code here: https://github.com/ethanachi/reweighting/tree/distilbert-inner-lr.

other forms of verification (Bowman et al., 2015)—many training examples are full of artifacts and other forms of bias (Poliak et al., 2018); bias that models pick up on, when training in a standard maximum likelihood regime. As a result, models trained on NLI datasets often learn brittle, spurious correlations and overfit to the oddities of their training sets, significantly hurting their ability to generalize to standard validation sets, let alone challenging out-of-domain NLI splits. Surprisingly, this problem seems endemic to these datasets, and is actually exacerbated by the strength of the model. Large, pretrained language models like BERT or GPT-2 (Devlin et al., 2019; Radford et al., 2019) are especially susceptible to these artifacts, possibly due to their capacity to memorize and quickly overfit; recent work shows that BERT fine-tuned 100 times on a standard NLI dataset exhibits generalization performance in the range of 0 (!) to 66% on an out-of-distribution (OOD) challenge set, even though all runs showed consistent performance in-domain (McCoy et al., 2020).

How do we solve this problem? While there is work interested in debiasing or collecting better NLI datasets, these miss the forest for the trees; as long as artifacts are present in the underlying datasets (or any future data these models are trained on), models *will* fit to them, and exhibit spurious correlations. The right way forward, then, is to develop a training procedure that *adaptively allows models to learn the "right" thing*; in other words, downweight the examples that hurt their generalization performance — examples we equate with those riddled with artifacts or other biases — and upweight others, that are more correlated with good generalization. As a result, in this work, we focus on *sample reweighting* as a means of building robust natural language inference models.

Building off of work by Ren et al. (2018), we leverage a (MAML-style) inner loop *meta-learning* approach to sample reweighting; by assuming access to a small validation set of clean examples that exhibit phenomena we want to encourage in our models (e.g., a certain type of generalization, robust performance in spite of labeling errors, etc.), we learn how to assign weights to examples in our training set that correlate with strong performance in our validation set. A more concrete visualization of our approach can be found in Figure 1. However, the original work by Ren et al. (2018) only proved their approach on simple, feed-forward style models with limited datasets (e.g., MNIST and CIFAR-100), and artifical sources of dataset noise. In this work, we build on their core meta-learning algorithm for reweighting by extending it to work with sequence models like Recurrent Neural Networks (Hochreiter and Schmidhuber, 1997; Cho et al., 2014) and large, pretrained transformer language models like BERT (Vaswani et al., 2017; Devlin et al., 2019). More importantly, while we show some initial experiments evaluating our approach against artificial dataset biases (label noise, label imbalance), we also show that we can use a subset of an NLI challenge set (McCoy et al., 2019) that tries to guard NLI models against learning a certain type of spurious correlation, and show that in tandem with our reweighting procedure, we can improve out-of-distribution generalization.

2 Related Work

We build our approach on a long body of work on training and evaluating models on various NLI datasets (Bowman et al., 2015; Williams et al., 2018; Nie et al., 2020), as well as recent work in sample reweighting (Chawla et al., 2004; Ren et al., 2018; Shu et al., 2019).

2.1 Natural Language Inference

Natural Language Inference (NLI) is a task on the critical path towards natural language understanding (NLU); many standard NLP tasks (such as question answering) can be expressed as NLI problems, and the simple setup of the task (classification given pairs of sentences) lends itself well to evaluating general-purpose language models. NLI is concerned with evaluating the relationship between sentences; Given a *premise* like "The doctor near the actor danced," and a *hypothesis* such as "The doctor danced," the goal is to predict whether the premise *entails*, *contradicts*, or *is neutral to* the hypothesis based solely off the language present in the two sentences (in this case, the premise *entails* the hypothesis)[2]. Such a task can be used to express a wide variety of linguistic and super-linguistic (e.g., commonsense) phenomena, and truly test a model's ability to perform language understanding and incorporate prior knowledge.

Datasets. The first large scale NLI dataset was the Stanford NLI dataset introduced by Bowman et al. (2015). It contains 570K sentence pairs, and was collected by leveraging captions from the Flickr30k corpus (Young et al., 2014) as *premises*; given captions for an image (without access to the image itself), crowd-

[2]Note that some work conflates the last two categories, effectively making NLI a binary classification problem between entails, and does not entail.

workers were asked to generate alternative captions — *hypotheses* — that were definitely true of the unseen image (entailed), could be true of the image (neutral), and that were definitely not true of the image (contradiction).

One of the limitations of the original SNLI dataset is the variety of text present — because the premises were all taken from Flickr30, the resulting dataset was fairly representative of the types of topics present in that dataset; namely everyday and outdoor activities, mostly with descriptions of physical scenes. To move past this, and to generally introduce a more diverse NLI dataset, Williams et al. (2018) introduced Multi-NLI (MNLI), the Multi-Genre Natural Language Inference Corpus. Instead of focusing on physical scenes, MNLI looks at several different genres to identify their premises, ranging from voice transcripts, to travel guides, to government reports, to excerpts from works of speculative fiction. MNLI significantly changed the landscape of NLI evaluation, as the multiple genres each require different types of language understanding. We use MNLI as our training set for all of the models in this work.

Separately, recent years have shown an explosion of different NLI datasets, including adversarial (**?**), cross-lingual (Conneau et al., 2018), and domain-specific NLI datasets (Romanov and Shivade, 2018).

Challenges. With the proliferation of various new NLI datasets, a parallel body of work investigated model performance, and specifically model errors by identifying structured ways of evaluating NLI models. Crucially, Gururangan et al. (2018) performed a detailed study of SNLI and MNLI and found that without access to the premise, over 60% of SNLI and 50% of MNLI can be correctly classified (compared to a 33% random baseline), indicating a deep-seated bias in the dataset. Probing deeper, they found annotator "shortcuts" — artifacts present in the datasets that were highly correlated with the output label. To be concrete, one such example they found is that the presence of the word "not" is highly indicative of *contradiction*, as one way crowdworkers could come up with contradictory hypotheses is to just add "not" somewhere in the original premise.

Expanding on this work, McCoy et al. (2019) identify three types of heuristics that are present in existing NLI systems, and develop a *challenge set* called HANS (Heuristic Analysis for NLI Systems) that specifically includes examples that counter these heuristics. To make this concrete, one heuristic present in existing NLI models trained on SNLI or MNLI is the "constitutent heuristic"; this is a feature of NLI models that assumes a label of *entails* anytime the hypothesis contains any complete substring of the original premise. For example, given a premise like "If the artist slept, the actor ran," and the hypothesis "The artist slept," models are likely to predict *entails*, even though this is not true. HANS is an interesting challenge set in that its examples specifically guard against spurious correlations or bad heuristics that NLI models might pick up naively when training with a maximum likelihood objective on MNLI. As such, we use HANS as one of our *validation sets* in our experiments, to see if we can use some limited knowledge about the heuristics to avoid to downweight the corresponding samples in our training set, and thus show better out-of-distribution generalization.

2.2 Sample Reweighting

We base our approach off of the principle of Sample Reweighting, which seeks to learn a weighting over examples in our training set such that when training via a maximum likelihood objective, we learn to prioritize examples that correlate with strong performance on our validation set, and downweight any other examples.

Sample reweighting is not a new idea; arguably, it can be traced back to Adaboost (Freund and Schapire, 1995). However, recent work has used the idea of sample reweighting to improve robustness by downweighting high-variance, imbalanced, or otherwise problematic examples (Chawla et al., 2004; Chang et al., 2017; Jiang et al., 2018). A focused portion of this work looks into meta-learning approaches for "learning-to-learn" sample reweighting, specifically those introduce by Ren et al. (2018) and Shu et al. (2019).

In this work, we build off the approach introduced by Ren et al. (2018) that uses inner-loop (MAML-style) meta-learning to learn-to-learn what examples are important, by leveraging an external (labeled) validation set. A broad overview of the approach can be found in Figure 1. Given a uniform set of weights, we first perform a forward-backward pass on a batch of training data to produce a new set of weights (preserving all gradients). Given these new weights, we perform a forward pass on a batch of validation data, and take the loss. We then perform a backwards-on-backwards pass (inner-loop/MAML-style) to optimize the weights over examples and minimize the validation loss. This is all enabled by standard automatic differentiation libraries that support taking second-order gradients.

In their original work, Ren et al. (2018) look at artificial sources of noise (label noise, label imbalance) on relatively small datasets with small architectures—specifically, MNIST, CIFAR-10, and CIFAR-100, with architectures like the standard LeNet (LeCun et al., 2015) and small variants of the ResNet architecture. In our work, we extend their approach by scaling up to sequence models like Recurrent Neural Networks (Hochreiter and Schmidhuber, 1997; Cho et al., 2014) as well as large pretrained transformer language models such as BERT (Vaswani et al., 2017; Devlin et al., 2019). More importantly, outside of artificial noise, we experiment with using subsets of challenge sets such as HANS (McCoy et al., 2019) to guide learning, guarding our NLI models from picking up on bad heuristics and learning spurious correlations.

3 Problem Statement

We work within (and adapt) the *reweighting* framework proposed by Ren et al. (2018). Assume that we have access to a large, noisy dataset $\mathcal{D}$ and a small, clean validation set $\mathcal{D}_V$. Under standard expected risk-minimization loss (ERM), we aim to minimize a loss equally weighted over the input examples:

$$\frac{1}{N}\sum_{i=1}^{N} C(\hat{y}_i, y_i) = \frac{1}{N}\sum_{i=1}^{N} f_i(\theta) \tag{1}$$

However, due to the noisiness of the training set, some examples may be more valuable to learn from than others. To ameliorate this, we aim to learn some weighting w_i of the input pairs (x_i, y_i) such that the loss $\mathcal{L}(\theta; \mathcal{D}_V)$ is minimized.

Given some set of N_t training examples, at every timestep t, we initialize a set of N free weights $\epsilon_i = 0$. We can construct a function $\theta'(\epsilon)$ which represents the updated parameters after a single inner optimization step with weights ϵ:

$$\theta'(\epsilon) = \theta - \lambda \nabla_\theta \sum_{i=1}^{n} \epsilon_i \mathcal{L}(\theta, y^i, \hat{y}^i) \tag{2}$$

We then sample a minibatch of clean validation examples and calculate the loss of the updated parameters θ' with respect to the validation step:

$$\mathcal{L}^{\text{val}}(\theta') = \sum_j \mathcal{L}(\theta', y^j_{\text{val}}, \hat{y}^j_{\text{val}}) \tag{3}$$

We then take the gradient of each ϵ_i with respect to the validation loss. In effect, this acts as a first-order approximation such that the new parameters $\hat{\theta}_{t+1}(\epsilon)$ generalize maximally well to the clean validation set:

$$u_i = -\eta \frac{1}{m}\sum_{j=1}^{m} \nabla_{\epsilon_i} \mathcal{L}^{\text{val}}(\theta'(\epsilon), y^j_{\text{val}}, \hat{y}^j_{\text{val}})\Big|_{\epsilon_i=0} \tag{4}$$

$$\tilde{w}_i = \max(u_i, 0) \tag{5}$$

where η is an inner-loop learning rate. Intuitively, $w_{i,t}$ is now an approximation of the "applicability" of the training example (x_i, y_i) to the validation set. In practice, we then rectify and rescale the w's to keep a constant and positive step size:

$$w_i = \frac{\tilde{w}_i}{(\sum_j \tilde{w}_j) + \mathbb{1}[\sum_j \tilde{w}_j = 0]}, \tag{6}$$

where the bottom right term accounts for the case in which every example has a negative weight. Finally, we take a weighted step on the train set with the calculated weights:

$$\theta'(\epsilon) = \theta - \lambda \nabla \sum_{i=1}^{n} w_i \mathcal{L}(\theta, y^i, \hat{y}^i) \tag{7}$$

4 Meta-Learning for Reweighting

We explored reweighting three sequential models, described below. All models were implemented in PyTorch (Paszke et al., 2017).

4.1 Bag-of-Words Models

As a simple baseline, we explored reweighting a *bag-of-words* model. Given a premise $\mathbf{p}$ and hypothesis $\mathbf{h}$, the bag-of-words model takes the mean over each sequence's embeddings:

$$p = \frac{1}{|\mathbf{p}|}\sum_i \text{Emb}(\mathbf{p}_i) \tag{8}$$

$$h = \frac{1}{|\mathbf{h}|}\sum_i \text{Emb}(\mathbf{h}_i) \tag{9}$$

A MLP is then trained to perform two- or three-way classification given p, h. Following Williams et al. (2018), we use difference and product features to leverage the features of the embedding space:

$$\hat{y} = \text{MLP}(p, h, p - h, p \odot h) \tag{10}$$

In practice, we embed sentences using GloVe-840B (Pennington et al., 2014). We use a four-layer MLP with 200 hidden units and ReLU activations, following Williams et al. (2018); however, for efficiency reasons, we do not use trainable embeddings, keeping them fixed.

4.2 Recurrent Neural Networks and BERT

Although this is an effective and computationally efficient baseline, taking the mean over the input sequences means that bag-of-words models are unable to model sequential context. Consequently, we investigated reweighting of two *sequential* models: RNN-GRU and fine-tuned BERT.

GRU-RNN Models A gated recurrent unit (GRU; Cho et al. 2014) is a recurrent neural network capable of modelling long-term dependencies. Unlike a standard RNN, a GRU models an additional update term z_t that allows for varying the degree of parameter update at each timestep; this allows for the modelling of complex long-term relationships without vanishing gradients.

BERT BERT (Devlin et al., 2019) is a pre-trained language model trained on a *masked language modelling* objective: given some input $\mathbf{x}$, random positions $x_{1,2,\ldots,i}$ are masked. The model is then trained on a reconstruction loss, through which it learns expressive representations which can then be fine-tuned for downstream tasks. As is standard, we fine-tuned the representations of the `[CLS]` token using a final linear classifier to make predictions.

Problem: Blocked by a `HIGHER` Power. We re-implemented Ren et al. (2018)'s reweighting algorithm, with adaptations appropriate for a NLP setting (e.g. gradient accumulation) in the PyTorch `higher` library (Grefenstette et al., 2019), which allows natural calculation of second-order gradients in an idiomatic way. Although this enabled reweighting a simple bag-of-words model, we rapidly encountered out-of-memory errors, often within a single epoch. Investigation revealed that the `higher` library caused memory leaks when applied to sequence models, due to incorrect freeing of free weights when exiting the meta-learning context. This would often increase memory needs by nearly $50\times$, making it impossible to complete the meta-training process.

We experimented with two strategies to address this:

- **Freezing:** First, both recurrent model (GRU or BERT) and classifier layer are trained using ERM. We then freeze these representations, re-initialize the final classification layers and performed reweighting solely on the last layer, using the ERM-trained recurrent representations. This approach, which is denoted ReweightLastLayer in our results table, achieved better performance than baselines but still did not achieve strong results.
- **Patching:** We manually patched the GRU in PyTorch to be stateless. We then performed manual inner gradient step descent as normal, updating the fast weights as necessary.

Re-Deriving the Reweighting Update Rule. In an effort to more concretely address the memory and computational cost of inner-loop meta-learning, we more closely examine the mathematical form of the meta-learning update rule. We find a simpler formulation that not only avoids the need for an inner update step, but also reveals parallels with work in *influence functions* (van der Vaart, 1998).

Recall that the gradient term is modelled as follows:

$$u_k = -\eta \frac{1}{m} \sum_{j=1}^{m} \nabla_{\epsilon_k} \mathcal{L}^{\text{val}}(\theta'(\epsilon), y^j_{\text{val}}, \hat{y}^j_{\text{val}}) \Big|_{\epsilon_k = 0} \tag{11}$$

Since this is a composed function of ϵ, we can apply the chain rule:

$$= -\eta \frac{1}{m} \sum_{j=1}^{m} \left[\nabla_{\theta'} \mathcal{L}^{\text{val}}(\theta', y^j_{\text{val}}, \hat{y}^j_{\text{val}}) \right] \left[\nabla_{\epsilon_k} \theta'(\epsilon) \right] \Big|_{\epsilon_k = 0} \tag{12}$$

$$= -\eta \frac{1}{m} \sum_{j=1}^{m} \left[\nabla_{\theta'} \mathcal{L}^{\text{val}}(\theta', y^j_{\text{val}}, \hat{y}^j_{\text{val}}) \right] \tag{13}$$
$$\left[\nabla_{\epsilon_k} \left(\theta - \lambda \nabla_\theta \sum_{i=1}^{n} \epsilon_i \mathcal{L}(\theta, y^i, \hat{y}^i) \right) \right] \Big|_{\epsilon_k = 0}$$

Simplifying:

$$= \eta \frac{1}{m} \sum_{j=1}^{m} \left[\nabla_{\theta'} \mathcal{L}^{\text{val}}(\theta', y^j_{\text{val}}, \hat{y}^j_{\text{val}}) \right] \tag{14}$$
$$\left[\nabla_{\epsilon_k} \nabla_\theta \sum_{i=1}^{n} \epsilon_i \mathcal{L}(\theta, y^i, \hat{y}^i) \right] \Big|_{\epsilon_k = 0}$$

$$= \eta \frac{1}{m} \sum_{j=1}^{m} \nabla_{\theta'} \mathcal{L}^{\text{val}}(\theta', y^j_{\text{val}}, \hat{y}^j_{\text{val}}) \cdot \nabla_\theta \mathcal{L}(\theta, y^i, \hat{y}^i) \tag{15}$$

where $\cdot$ is a dot product.

In summary, we demonstrate that the inner-loop meta-learning procedure reduces to a dot product between the gradient with respect to the train loss and gradient with respect to the validation loss. Rather than taking an inner loop step over ϵ, we can now explicitly estimate the influence term analytically, allowing us to **reweight entire sequential models** while drastically speeding up computation.

Model	Label Noise			
	Baseline	0.5	0.9	Imbalance
Bag-of-Words				
ERM	0.536	0.471	0.389	0.524
REWEIGHT	0.535	0.522	0.422	0.588
GRU				
ERM	0.660	0.480	0.440	0.488
REWEIGHTLASTLAYER	OOM†	OOM†	OOM†	0.543
REWEIGHT	0.631	0.530	*0.361**	0.555

Table 1: Performance on the MultiNLI `dev-mismatched` set, which we use as an evaluation set, under various training conditions. For all conditions, a random baseline gives 0.333. *Did not converge at time of writing. †Out-of-memory error encountered due to memory leaks in the `higher` library; see Section 4.2 for details of the error and steps taken to address this.

Connections with Influence Functions. The goal of the influence functions framework is to estimate the change in the loss at a given test example by upweighting the loss at a given training example. In the classic derivation (c.f. Van der Vaart (1998)), to compute the influence of a training example $z = (x, y)$ on a test example $z_t = (x_t, y_t)$, we start by changing its weight from 1 to $1 + \epsilon$. This leads to an updated empirical loss, which we *minimize* to obtain updated model parameters θ'. This is then used to compute the updated loss on the test example. Mathematically, the influence score is given by

$$\mathcal{I}_{z,z_t} = -\nabla_\theta \mathcal{L}(\theta, y, \hat{y}) H_\theta^{-1} \nabla_\theta \mathcal{L}(\theta, y_t, \hat{y}_t), \quad (16)$$

where H_θ^{-1} is the inverse hessian of the loss function at θ. We note that this has a similar functional form as the reweighting computed in Equation 15, except for the hessian term in the middle. Intuitively, we expect reweighting to assign high weights to examples that would *influence* the validation loss most; the derivation presented above validates this intuition.

5 Experiments

5.1 MNIST

To verify our implementation, we evaluate on the image-recognition setting from Ren et al. (2018).

Dataset. We investigate the ability of reweighting to learn to classify an imbalanced subset of MNIST. We select a total of 5,000 images from classes **4** and **9**, where **9** makes up 99.5% of the training distribution. The testing distribution is evenly split between **4** and **9**. To inform the reweighting algorithm, we also construct a balanced validation set of 10 examples.

Results. We train LeNet-5 (LeCun et al., 2015) with both the reweighting objective and the standard ERM[3] objective. Our results (Table 2) demonstrate that reweighting significantly outperforms the baseline.

Model	Performance
ERM	0.790
REWEIGHT	0.950

Table 2: Performance of a LeNet-5 model on the Imbalanced-MNIST dataset with standard training and reweighting. Our PyTorch reimplementation replicates the performance achieved by Ren et al. (2018).

5.2 MultiNLI: Label Noise and Imbalance

Similar to MNIST experiments, we evaluate NLI models under both label noise and label imbalance. For these experiments, we use *Bag-of-Words* and *GRU* models; for the GRU, we experiment with both reweighting the top layer only (with `higher`; **REWEIGHTLL**) and reweighting the entire model (with our custom-patched GRU; **REWEIGHT**).

Dataset. Under the *label noise* condition, we flip a fraction p of labels to a random label. We experiment with $p \in \{0.5, 0.9\}$. Under the *label imbalance* condition, almost all of the data points come from one class; this thus tests our algorithm's ability to upweight relevant examples from the minority class. We use 120K entailment examples, 13K non-entailment examples, and 13K contradiction examples. In all cases, we use the MultiNLI `dev-matched` set as a validation set for reweighting, and the `dev-mismatched` set for evaluation; the two sets have no genre overlap.

[3] Empirical risk minimization

Results. Results can be found in Table 1. We find that for the bag-of-words baseline, reweighting improves performance significantly, with improvements of +3.3 and +5.1 points on our two noising conditions and a +6 point improvement on the imbalance condition. For GRUs, we observe inconclusive results on the noise conditions, with a +5.0 point improvement on $p = 0.5$ but a +7.9 point deterioriation on $p = 0.9$ (we note, however, that the $p = 0.9$ model converged extremely slowly and had not converged within 10 iterations). On the imbalance condition, we observe a +5.5 point improvement over ERM when fine-tuning a single linear layer, and a +6.7 point improvement when fine-tuning the entire GRU. We did find that GRU underperformed the bag-of-words model on all noisy conditions, possibly due to the heuristic-laden nature of the MultiNLI dataset.

5.3 MultiNLI-HANS

We now evaluate models on the MultiNLI-HANS setting which we describe below. For this evaluation, we experiment with a GRU based model as well as BERT, a large scale pre-trained transformer model.

Data. We train on a binary version of the MultiNLI dataset, by collapsing both `neutral` and `contradiction` labels to a `not-entailment` label. As our clean validation set, we sample 5000 example from the `subsequence` subset of HANS examples. We use the `lexical-overlap` subset of HANS examples as our test set. Notably, these two subsets target completely different heuristics, so transfer from one subset to another is challenging.

Results. From Table 3, we observe that reweighting leads to improved performance for all settings. Compared to the baseline **ERM** model, we observe an improvement of +13 points for GRU encoders. For the more powerful BERT encoder, we observe that *end-to-end* finetuning of BERT with reweighting leads to an improvement of +2.2 points. Interestingly, we observe that finetuning only the final layer of BERT leads to a slight improvement of +0.9 points.

5.4 Analysis

In this experiment, we seek to understand if reweighting can recover examples in the training set that come from the same distribution as the test set. To do this, we randomly remove 100 examples from the test set of MultiNLI-HANS and add it to the training set. From results in Table 4, we observe that the reweighting based BERT model is able to reach a perfect accuracy in this setting while an ERM based model achieves an underwhelming accuracy of 0.538. This indicates that the reweighting based model is able to accurately upweight leaked examples, and simultaneously downweight everything else. Unlike fine-tuning, our method does not require us to know which examples are important to achieve good performance.

Model	Performance
GRU	
ERM	0.490
REWEIGHT	0.561
BERT Finetuning	
ERM	0.527
REWEIGHTLASTLAYER	0.558
REWEIGHT	0.549

Table 3: Performance of GRUs and BERT finetuning on the MultiNLI-HANS setting. Random baseline: 0.5.

Model	Performance
ERM	0.538
REWEIGHT	1.0

Table 4: Comparing ERM with REWEIGHT by training on a combination of MultiNLI-HANS training data with 100 leaked test set examples. We observe that REWEIGHT obtains perfect accuracy suggesting that it is able to recover the leaked examples.

6 Conclusion

For several NLP problems, data-driven neural network models are approaching super-human performance. However, a closer inspection reveals that these state-of-the-art models learn superficial correlations that fail to generalize beyond the training distribution (Jia and Liang, 2017; Gururangan et al., 2018; McCoy et al., 2019). On the other hand, human understanding of language is remarkably robust to distribution shifts—we are able to read books set in fictional worlds, systematically interpret new combinations of known words, and effortlessly use newly acquired words in conversations. How can we make progress towards building models with similar generalization properties?

In this work, we apply a reweighting based approach from Ren et al. (2018) to make progress towards this goal, for the task of NLI. The key idea is to dynamically modify the training distribution by

upweighting examples that lead to improved generalization and downweighting examples that do not. Crucially, to scale this approach to large NLP models, we reformulate the meta-learning approach as a simple dot product between gradients. From experiments on several challenging domain shifts, we establish the effectiveness of this approach.

Acknowledgements

We would like to thank the CS 330 CAs (especially Dilip Arumugam, our project TA), as well as our amazing instructors Chelsea Finn and Karol Hausman for an amazing class. Not only were the lectures amazing, but the assignments allowed us to grasp concepts that would have otherwise just been a bunch of vague math in the back of our heads. This final project also allowed us hands-on experience applying meta-learning techniques in practice to areas that we are passionate about. Thank you all for this opportunity, and for such a great class!

References

Y. Belinkov, A. Poliak, S. Shieber, B. V. Durme, and A. M. Rush. 2019. Don't take the premise for granted: Mitigating artifacts in natural language inference. In *Association for Computational Linguistics (ACL)*.

S. Bowman, G. Angeli, C. Potts, and C. D. Manning. 2015. A large annotated corpus for learning natural language inference. In *Empirical Methods in Natural Language Processing (EMNLP)*.

H. Chang, E. Learned-Miller, and A. McCallum. 2017. Active bias: Training more accurate neural networks by emphasizing high variance samples. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 1002–1012.

N. V. Chawla, N. Japkowicz, and A. R. Kolcz. 2004. Editorial: Special issue on learning from imbalanced data sets. *ACM SIGKDD Explorations Newsletter*, 6(1).

K. Cho, B. van Merriënboer, D. Bahdanau, and Y. Bengio. 2014. On the properties of neural machine translation: Encoder-decoder approaches. *arXiv preprint arXiv:1409.1259*.

A. Conneau, R. Rinott, G. Lample, A. Williams, S. Bowman, H. Schwenk, and V. Stoyanov. 2018. Xnli: Evaluating cross-lingual sentence representations. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 2475–2485.

I. Dagan, O. Glickman, and B. Magnini. 2006. The PASCAL recognising textual entailment challenge. In *Machine learning challenges. evaluating predictive uncertainty, visual object classification, and recognising tectual entailment*, pages 177–190.

D. Demszky, K. Guu, and P. Liang. 2018. Transforming question answering datasets into natural language inference datasets. *arXiv preprint arXiv:1809.02922*.

J. Devlin, M. Chang, K. Lee, and K. Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Association for Computational Linguistics (ACL)*, pages 4171–4186.

Y. Freund and R. Schapire. 1995. A decision-theoretic generalization of on-line learning and an application to boosting. In *Conference on Learning Theory (COLT)*.

E. Grefenstette, B. Amos, D. Yarats, P. M. Htut, A. Molchanov, F. Meier, D. Kiela, K. Cho, and S. Chintala. 2019. Generalized inner loop meta-learning. *arXiv preprint arXiv:1910.01727*.

S. Gururangan, S. Swayamdipta, O. Levy, R. Schwartz, S. Bowman, and N. A. Smith. 2018. Annotation artifacts in natural language inference data. In *Association for Computational Linguistics (ACL)*, pages 107–112.

S. Hochreiter and J. Schmidhuber. 1997. Long short-term memory. *Neural Computation*, 9(8):1735–1780.

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

L. Jiang, Z. Zhou, T. Leung, L. Li, and L. Fei-Fei. 2018. MentorNet: Learning data-driven curriculum for very deep neural networks on corrupted labels. In *International Conference on Machine Learning (ICML)*, pages 2304–2313.

Yann LeCun et al. 2015. Lenet-5, convolutional neural networks. *URL: http://yann. lecun. com/exdb/lenet*, 20(5):14.

B. McCann, N. S. Keskar, C. Xiong, and R. Socher. 2018. The natural language decathlon: Multitask learning as question answering. *arXiv preprint arXiv:1806.08730*.

R. T. McCoy, J. Min, and T. Linzen. 2020. Berts of a feather do not generalize together: Large variability in generalization across models with similar test set performance. In *Proceedings of the Third BlackBoxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP @ EMNLP*.

R. T. McCoy, E. Pavlick, and T. Linzen. 2019. Right for the wrong reasons: Diagnosing syntactic heuristics in natural language inference. In *Association for Computational Linguistics (ACL)*.

Y. Nie, A. Williams, E. Dinan, M. Bansal, J. Weston, and D. Kiela. 2020. Adversarial nli: A new benchmark for natural language understanding. In *Association for Computational Linguistics (ACL)*.

Yixin Nie, Adina Williams, Emily Dinan, Mohit Bansal, Jason Weston, and Douwe Kiela. 2019. Adversarial nli: A new benchmark for natural language understanding. *arXiv preprint arXiv:1910.14599*.

A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer. 2017. Automatic differentiation in pytorch.

J. Pennington, R. Socher, and C. D. Manning. 2014. GloVe: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543.

A. Poliak, J. Naradowsky, A. Haldar, R. Rudinger, and B. V. Durme. 2018. Hypothesis only baselines in natural language inference. In *Joint Conference on Lexical and Computational Semantics*.

A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever. 2019. Language models are unsupervised multitask learners. *OpenAI Blog*, 1(8).

M. Ren, W. Zeng, B. Yang, and R. Urtasun. 2018. Learning to reweight examples for robust deep learning. In *International Conference on Machine Learning (ICML)*.

A. Romanov and C. Shivade. 2018. Lessons from natural language inference in the clinical domain. In *Empirical Methods in Natural Language Processing (EMNLP)*.

J. Shu, Q. Xie, L. Yi, Q. Zhao, S. Zhou, Z. Xu, and D. Meng. 2019. Meta-Weight-Net: Learning an explicit mapping for sample weighting. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 1919–1930.

A. W. van der Vaart. 1998. *Asymptotic statistics*. Cambridge University Press.

AW Van der Vaart. 1998. Asymptotic statistics.

A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin. 2017. Attention is all you need. *arXiv preprint arXiv:1706.03762*.

A. Wang, Y. Pruksachatkun, N. Nangia, A. Singh, J. Michael, F. Hill, O. Levy, and S. R. Bowman. 2019a. SuperGLUE: A stickier benchmark for general-purpose language understanding systems. In *Advances in Neural Information Processing Systems (NeurIPS)*.

A. Wang, A. Singh, J. Michael, F. Hill, O. Levy, and S. R. Bowman. 2019b. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *International Conference on Learning Representations (ICLR)*.

A. Williams, N. Nangia, and S. Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Association for Computational Linguistics (ACL)*, pages 1112–1122.

P. Young, A. Lai, M. Hodosh, and J. Hockenmaier. 2014. From image descriptions to visual denotations: New similarity metrics for semantic inference over event descriptions. *Transactions of the Association for Computational Linguistics (TACL)*, 2:67–78.