What if I don't have in-domain unlabeled data for Semi-Supervised Learning? Well, generate some!

Xuanli He [1] **Islam Nasser** [1] **Jamie Kiros** [2] **Gholamreza Haffari** [1] **Mohammad Norouzi** [2]

Abstract

Semi-Supervised Learning (SSL) has seen success in many application domains, but this success often hinges on the availability of in-domain unlabeled data. We present Generative Self-Training (GeST), a simple refinement of SSL algorithms, in particular self-training, that alleviates the need for in-domain unlabeled data. The key idea is to train an unconditional domain-specific generative model, and use it to generate synthetic unlabeled data for SSL. To train strong domain-specific generative models, one fine-tunes generic generative models (trained on open-domain data) on specific domains. GeST enables combining the benefits of large language models and large self-supervised representations; when GPT-2-large is fine-tuned separately on inputs of each GLUE task and used as the generative model of GeST to self-train RoBERTa-large, we achieve an average improvement of 1.3% over fine-tuned RoBERTa-large, yielding state-of-the-art performance of 90.1% on GLUE dev sets. Moreover, GeST achieves significant gains on CIFAR-10 and two UCI tabular datasets: connect-4 and Drug Review. Finally, we show that knowledge distillation using generated unlabeled data can help bridge the gap between 12- and 6-layer transformers on GLUE tasks.

1. Introduction

Unlabeled data is abundant in the real world, but domain-specific unlabeled data within the scope of a given machine learning problem is challenging to find. For instance, one cannot easily find in-domain unlabeled data conforming to the input distribution of a specific Natural Language Processing (NLP) task from the GLUE benchmark (Wang et al., 2019b). Some NLP tasks require an input comprising a sentence pair with a particular relationship between them or a question-paragraph pair. Moreover, supervised datasets are tailored toward a certain text or image distribution and only include a limited number of class labels. If domain-specific unlabeled data were available, one could adopt self-training (Yarowsky, 1995) to automatically annotate unlabeled data with pseudo labels to help improve accuracy and robustness of machine learning models. This paper aims to make self-training more universally applicable by leveraging *generated* unlabeled data within self-training.

There has been a surge of interest in improving accuracy and label efficiency of machine learning models either via:

1. *Self-Supervised pretraining* on open-domain unlabeled data in a task-agnostic way (*e.g.,* Peters et al. (2018); Devlin et al. (2019); Chen et al. (2020b)), or,
2. *Self-Training* using domain-specific unlabeled data in a task-specific way (*e.g.,* Rosenberg et al. (2005); McClosky et al. (2006); Xie et al. (2020)).

While self-supervised learning can be applied to a broad distribution of unlabeled data, self-training requires unlabeled data that at least can be annotated using the same set of class labels available for the downstream task (Oliver et al., 2018). For instance, if one is interested in training a classifier to distinguish images of cats and dogs, self-training with images of aircraft is likely not helpful, but it is conceivable that self-supervised learning with images of aircraft can still help. A growing body of recent work suggests that perhaps self-supervised pretraining and self-training are compatible and can be combined to achieve the best semi-supervised learning performance (Chen et al., 2020c; Du et al., 2020). We corroborate the existing evidence by showing gains from our *generative* self-training on top of BERT on GLUE tasks.

The dependence of self-training on in-domain unlabeled data has made it hardly applicable to realistic problems without in-domain unlabeled data. To address this challenge, Du et al. (2020) have used nearest neighbor retrieval to harvest in-domain unlabeled data from a large corpus of open-domain text, leading to a successful application of self-training to certain NLP tasks. While retrieval can indeed help find in-domain data for problems with simple inputs, it is not practical for problems with complex input schemes, *e.g.,* sentence pairs with certain relations and tabu-

[1]Monash University, Australia [2]Google Research, Toronto. Correspondence to: Xuanli He <xuanli.he1@monash.edu>, G. Haffari <gholamreza.haffari@monash.edu>, M. Norouzi <mnorouzi@google.com>.

lar data. Accordingly, to our knowledge, no prior work has successfully applied self-training to tasks from the GLUE benchmark that often involve mulit-sentence inputs.

We present Generative Self-Training (GeST), a simple refinement of self-training that alleviates the need for in-domain unlabeled data. The key idea of GeST is to train an unconditional domain-specific generative model, and use it to generate lots of synthetic unlabeled data, useful for self-training. Thus, the difference between self-training and GeST is that self-training uses existing in-domain unlabeled data, annotated with synthetic labels, whereas GeST uses both synthetic unlabeled data and synthetic labels. Building on recent advances in text and image generation (Radford et al., 2019; Karras et al., 2020), we train strong domain-specific generative model for GeST, by fine-tuning an existing generative model that has been pretrained on open-domain data on specific domains. This works particularly well for NLP applications, where large language models such as GPT-2 are publicly available.

Summary of the results. When GPT-2-large is separately fine-tuned on inputs of each GLUE task and used as the generative model of GeST to self-train RoBERTa-large, we achieve an average improvement of 1.3% over fine-tuned RoBERTa-large and SOTA score of 90.1% on GLUE dev sets. In this setting, open-domain text is used twice; once for training RoBERTa, and again for training GPT-2.

In addition, GeST built on FixMatch (Sohn et al., 2020) achieves consistent gains on CIFAR-10 using a score-based unconditional generative model solely trained on CIFAR-10 (Song & Ermon, 2019). Using a synthetic dataset $10\times$ bigger than the training set, we achieve an improvement of up to 1% on CIFAR-10 test set on various model architectures.

We apply GeST to two generic tabular tasks from the UCI dataset repository, connect-4 (Burton & Kelly, 2006) and Drug Review (Gräßer et al., 2018), and achieve an improvement of 2.5% and 1.2% respectively. Finally, we show that using synthetic data for knowledge distillation (KD) can help bridge the gap between 12- and 6-layer transformers on the GLUE benchmark, outperforming several existing KD baselines. Our main contributions are summarized as:

- We propose GeST: a novel wrapper around SSL and KD that advocates the use of unconditional generative models to synthesize in-domain unlabeled data for SSL and KD.
- We demonstrate the efficacy of GeST on common NLP, computer vision, and tabular tasks.
- We link GeST to empirical risk minimization and vicinal risk minimization, which helps explain why GeST works and why using class-conditional generative models to obtain synthetic data is often not as effective.
- We systematically dissect GeST and study the key components leading to its success.

2. Related Work

Semi-supervised learning (SSL) has received considerable attention over the last few decades (Cooper & Freeman, 1970; McLachlan & Ganesalingam, 1982; Riloff, 1996; Chapelle et al., 2009; Van Engelen & Hoos, 2020). One of the oldest family of SSL algorithms is known as *self-training*, *a.k.a.* self-learning or self-labeling (Scudder, 1965; Fralick, 1967; Agrawala, 1970; Yarowsky, 1995). The main intuition of self-training is to encourage knowledge transfer between a *teacher* and a *student* model in such a way that the student can outperform the teacher. Specifically, one leverages the teacher's knowledge to annotate unlabeled data with so-called *pseudo labels*, and the student learns from a mixture of pseudo- and human-labeled data. Self-training has seen a surge of recent interest across vision and NLP applications (Yalniz et al., 2019; Xie et al., 2020; Zoph et al., 2020; Du et al., 2020).

Recent work aims to combine self-training and *consistency regularization* to develop powerful SSL algorithms. The key idea is to ensure that the predictions of a classifier on unlabeled examples are robust to strong augmentations (Berthelot et al., 2019a; Sohn et al., 2020; Xie et al., 2019). We build on prior work and investigate the use of synthetic data within the broad family of self-training methods.

Recent theoretical work analyzes self-training for linear models, often under the assumption that the data distribution is (nearly) Gaussian (Carmon et al., 2019; Raghunathan et al., 2020; Chen et al., 2020d; Kumar et al., 2020a; Oymak & Gulcu, 2020). Wei et al. (2021) prove that, under "expansion" and "class separation" assumptions, self-training can lead to more accurate neural network classifiers. We link GeST to empirical and vicinal risk minimization (Vapnik, 1992; Chapelle et al., 2001), but leave deeper theoretical understanding of GeST to future work.

An important family of related work uses generative models for SSL by learning features that are useful for both generation and discrimination (*e.g.,* Chen et al. (2020a); Odena (2016); Dai et al. (2017)). For instance, Kingma et al. (2014) approach SSL by viewing missing class labels as a set of latent variables and use variational inference to impute missing labels as well as other factors of variation. By contrast, our work does not make use of features learned by deep generative models and keeps the generative and discriminative processes separate. This gives us more flexibility and allows GeST to use self-supervised approaches that are not fully generative to pretrain discriminative models.

Knowledge Distillation (KD) (Buciluǎ et al., 2006; Hinton et al., 2015) uses a procedure similar to self-training to distill knowledge of an expressive teacher model into a smaller student model. In contrast, self-distillation (Furlanello et al., 2018; Zhang et al., 2019a; Mobahi et al., 2020) uses teacher

and student models of equal size, hoping to iteratively refine class labels. Previous work uses unlabeled data (Buciluǎ et al., 2006) and adversarial training (Wang et al., 2018) to improve KD. We demonstrate that synthetic data generated by unconditional generative models can improve KD on NLP, outperforming strong baselines (*e.g.,* Xu et al. (2020)).

Advanced generative models are able to generate realistic images and text (Karras et al., 2017; Brock et al., 2019; Karras et al., 2019; Radford et al., 2019; Brown et al., 2020). The quality of synthetic samples has improved to the extent that deep fake detection has become an important research topic itself (Zellers et al., 2019; Dolhansky et al., 2019). Recent work has aimed to utilize class-conditional generative models to help improve supervised learning (Antoniou et al., 2017; Bowles et al., 2018; Zhang et al., 2019b; Kumar et al., 2020b; Gao et al., 2020). However, Ravuri & Vinyals (2019) have shown that images generated by state-of-the-art class-conditional generative models fall short of improving ImageNet classification accuracy, despite strong sample quality scores (Salimans et al., 2016; Heusel et al., 2017). Similarly, Kumar et al. (2020b) find that it is difficult for sentences generated by label-conditioned GPT-2 (Radford et al., 2019) to retain the semantics or pragmatics of a specified category, which leads to poor performance on downstream tasks. We discuss why class-conditional generative models are hardly effective for supervised learning, and instead, focus on unconditional generative models.

3. Self-Training

Given a labeled dataset $L = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^N$ and an unlabeled dataset $U = \{\boldsymbol{x}_j\}_{j=1}^M$, we summarize the general family of SSL algorithms known as self-training (Algorithm 1) as:

1. First, an initial model denoted f_1 is trained using supervised learning on the labeled dataset L.
2. Then, at iteration t, one adopts f_t as the teacher model to annotate the unlabeled dataset U using *pseudo labels*.
3. Optionally, one uses a selection method to pick a subset $S_t \subseteq \{(\boldsymbol{x}_j, f_t(\boldsymbol{x}_j))\}_{j=1}^M$ of pseudo labeled examples.
4. A student model f_{t+1} is trained to optimize a classification loss on the combination of L and S_t:

$$\ell_{t+1} = \mathbb{E}_{(\boldsymbol{x},y)\sim(L \cup S_t)} H(y, f_{t+1}(\boldsymbol{x})) \ , \qquad (1)$$

   where $H(q, p) = q^\top \log p$ is the softmax cross entropy loss, and y is assumed to be a one-hot vector (original labels) or a vector of class probabilities (pseudo labels).
5. Self-training iterations are repeated T times or until performance plateaus.

Many different variants of the basic self-training algorithm discussed above exist in the literature. These variants differ in the type of pseudo labels used, the selection strategy to filter pseudo labeled examples, the speed at which f_t is replaced with f_{t+1}, the choice of data augmentation strategy in the teacher and student models, and the weighting between original and pseudo labeled datasets in the objective (Berthelot et al., 2019b;a; Xie et al., 2020; Sohn et al., 2020; Du et al., 2020).

Algorithm 1 SelfTraining(L, U, f_0, T)

Input: Labeled dataset $L = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^N$
Unlabeled dataset $U = \{\boldsymbol{x}_j\}_{j=1}^M$
Initial parameters of a classifier f_0
Output: A better classifier f_{T+1} after T self-training steps
1: train a base model f_1 by fine-tuning f_0 on L
2: **for** $t = 1$ to T do:
3: apply f_t to unlabeled instances of U
4: select a subset $S_t \subseteq \{(x, f_t(x)) \mid x \in U\}$
5: train a new model f_{t+1} by either fine-tuning f_0 on $L \cup S_t$ or gradient descend on a minibatch from $L \cup S_t$
6: **return** f_{T+1}

Algorithm 2 GeST(L, g_0, f_0, k, T)

Input: Labeled dataset $L = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^N$
Initial parameters of a generative model g_0
Initial parameters of a classifier f_0
Output: A better classifier f_{T+1} after T GeST steps
1: train a generative model g by fine-tuning g_0 on L_x where $L_x = \{\boldsymbol{x} \mid (\boldsymbol{x}, y) \in L\}$
2: generate $U = \{\widetilde{\boldsymbol{x}}_j\}_{j=1}^{kN}$ by drawing kN random samples *i.i.d.* from $g(\boldsymbol{x})$, *i.e.,* $\widetilde{\boldsymbol{x}}_j \sim g(\boldsymbol{x})$ for $j = 1$ to kN.
3: **return** SelfTraining(L, U, f_0, T)

An important design choice is the type of pseudo labels used. One can simply use soft class probabilities predicted by a teacher f_t (Du et al., 2020), sharpened class probabilities (Berthelot et al., 2019b), or hard labels (a one-hot vector that is zero except at $\text{argmax} f_t(\boldsymbol{x})$) (Lee et al., 2013). Alternatively, one can use meta-learning to derive pseudo labels (Pham et al., 2020). Another important consideration is the selection strategy to retain a subset of pseudo-labeled examples. FixMatch (Sohn et al., 2020) uses a hyper-parameter τ to select examples on which the teacher model has a certain level of confidence, *i.e.,*

$$S_t = \{(\boldsymbol{x}, f_t(\boldsymbol{x})) \mid \boldsymbol{x} \in U \ \& \ \max(f_t(\boldsymbol{x})) \geq \tau\} \ . \qquad (2)$$

NoisyStudent (Xie et al., 2020) also uses a form of confidence filtering but ensures that the class labels in the selected subset are balanced. In principle, any method for out-of-distribution detection (Hendrycks & Gimpel, 2016) can be adopted for filtering pseudo-labeled examples.

Our contribution in this paper is orthogonal to the specific design choices of self-training. We adopt relatively simple variants of self-training and limit hyper-parameter tuning to a bare minimum. That said, more advanced self-training algorithms will likely improve the results further. For NLP and tabular tasks we adopt soft pseudo labels without sharpening, and we do not adopt any subset selection method. For vision tasks we adopt FixMatch (*i.e.,* hard pseduo labels and confidence-based filtering in (2)).

4. Generative Self-Training (GeST)

Given a labeled dataset $L = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^N$, we first train an unconditional domain-specific generative model $g(\boldsymbol{x})$ on $L_x = \{\boldsymbol{x}_i\}_{i=1}^N$, and then use it to synthesize unlabeled data. Such synthetic unlabeled data is used to enable the adoption of self-training even without in-domain unlabeled data. We call this general framework Generative Self-Training (GeST) because it uses generated data within self-training. The objective function of GeST during iteration t of self-training, provided a teacher model f_t, is expressed as:

$$\begin{aligned}\ell_{t+1} \; &= \lambda \, \mathbb{E}_{(\boldsymbol{x},y)\sim L} H(y, f_{t+1}(\boldsymbol{x})) \\ &\quad + (1-\lambda) \, \mathbb{E}_{\widetilde{\boldsymbol{x}}\sim g(\boldsymbol{x})} H(f_t(\widetilde{\boldsymbol{x}}), f_{t+1}(\widetilde{\boldsymbol{x}})) \; .\end{aligned} \tag{3}$$

Here, we assume the use of soft pseudo labels within self-training. In practice to improve computational efficiency of GeST, we do not generate unlabeled data on the fly as implied by (3). Rather, we generate as many unconditional samples as practically feasible and store them in a synthetic unlabeled dataset U. For most experiments we use $\lambda = 0.5$.

Not surprisingly, the size of the gains from GeST depends on the fidelity and diversity of synthetic examples. We find that recent methods for text generation via autoregressive modeling and image generation via iterative refinement are particularly effective when used as the generative engine of GeST to empower self-training. Semi-supervised learning methods other than self-training can potentially benefit from our unconditional generative models as well, but we restrict our attention to self-training and its variants. Section 5.4 discusses the use of generated data within knowledge distillation to help effectively compress neural networks.

The GeST framework entails a wrapper around self-training (or other SSL algorithms) as shown in Algorithm 2 and Figure 1. To obtain the best domain-specific generative model $g(\boldsymbol{x})$, one can pretrain a generic generative model g_0 on lots of open-domain data first, and then fine-tune on a specific domain as manifested in L_x. This is particularly helpful if the size of L_x is small (Hernandez et al., 2021). In what follows, we first discuss how GeST is connected to empirical and vicinal risk minimization, which helps motivate the approach. Then, we discuss practical considerations around building domain-specific generative models in Section 4.2.

4.1. An Empirical Risk Minimization Perspective

In supervised learning, one seeks to learn a mapping f that given an input $\boldsymbol{x}$, predicts a reasonable output y. To define the supervised learning problem formally, one assumes that input-output pairs are drawn from a joint distribution P, *i.e.,* $(\boldsymbol{x}, y) \sim P(\boldsymbol{x}, y)$, and a loss function $H(y, f(\boldsymbol{x}))$ is used to assess the quality of a mapping f. This loss is used to define a notion of *expected risk*:

$$R(f) = \mathbb{E}_{P(\boldsymbol{x},y)} H(y, f(\boldsymbol{x})) \; . \tag{4}$$

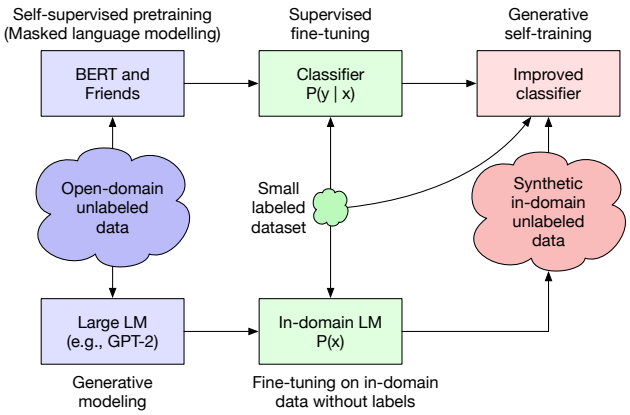

Figure 1. An illustration of the proposed Generative Self-Training (GeST) approach for NLP applications. We use open-domain data once for self-supervised pretraining (*e.g.,* BERT) and once for training a large LM (*e.g.,* GPT-2). Then, BERT is fine-tuned on in-domain data to yield a classifier for the task of interest. GPT-2 is fine-tuned on in-domain data without labels to obtain an unconditional domain-specific LM, which is then used to generate a lot of synthetic in-domain unlabeled data. Finally, we perform a few iterations of self-training by leveraging the small amount of labeled data and the large amount of synthetic unlabeled data.

In almost all practical applications $P(\boldsymbol{x}, y)$ is unknown. Hence, a labeled dataset of examples $L = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^N$ is used to approximate $R(f)$ as

$$\widehat{R}(f) = \frac{1}{N} \sum\nolimits_{i=1}^{N} H(y_i, f(\boldsymbol{x}_i)) \; . \tag{5}$$

This objective function is known as *empirical risk*, and learning f through minimizing $\widehat{R}(f)$ is known as the *empirical risk minimization* principle (Vapnik, 1992). To compensate for the finite sample size in (5), one typically combines $\widehat{R}(f)$ with a regularizer to improve generalization.

Beyond empirical risk minimization. Empirical risk minimization (5) is motivated as a way to approximate $P(\boldsymbol{x}, y)$ through a set of Dirac delta functions on labeled examples: $P_\delta(\boldsymbol{x}, y) = \sum_i \delta(\boldsymbol{x} = \boldsymbol{x}_i, y = y_i)/N$. However, this approximation is far from perfect, hence one uses a heldout validation set for early stopping and hyper parameter tuning.

Vicinal risk minimization (Chapelle et al., 2001) approximates expected risk as $\mathbb{E}_{P_\nu(\boldsymbol{x},y)} H(y, f(\boldsymbol{x}))$, using a *vicinity distribution*, *e.g.,* $\nu(\tilde{\boldsymbol{x}}, \tilde{y} \mid \boldsymbol{x}, y) = \mathcal{N}(\tilde{\boldsymbol{x}} - \boldsymbol{x}, \sigma^2)\delta(\tilde{y} = y)$ to approximate $P(\boldsymbol{x}, y)$ as

$$P_\nu(\boldsymbol{x}, y) = \frac{1}{N} \sum\nolimits_{i=1}^{N} \nu(\tilde{\boldsymbol{x}}, \tilde{y} \mid \boldsymbol{x}_i, y_i) \; . \tag{6}$$

The goal is to increase the support of each labeled data point and improve the quality and robustness of the risk function.

Recent work on mixup regularization (Zhang et al., 2018) proposes an effective way to construct another vicinity distribution by interpolating between two data points and their

labels. While these smoothing techniques tend to improve matters, it is hard to believe that simple kernels or interpolation techniques can fully capture the complexity of $P(\boldsymbol{x}, y)$. Moreover, designing $\nu(\tilde{\boldsymbol{x}}, \tilde{y} \mid \boldsymbol{x}, y)$

Generative models for risk minimization. One can factorize the joint distribution of input-output pairs as $P(\boldsymbol{x}, y) = P(\boldsymbol{x})P(y \mid \boldsymbol{x})$. Accordingly, if one is able to learn a reasonable unconditional generative model of $\boldsymbol{x}$ denoted $g(\boldsymbol{x})$, then one can draw a pair $(\boldsymbol{x}, y)$ by first drawing $\boldsymbol{x} \sim g(\boldsymbol{x})$ and then using the current instance of f_t to draw $y \sim f_t(\boldsymbol{x})$. Then, one can use f_t and g to approximate expected risk as

$$R_t(f_{t+1}) = \mathbb{E}_{\boldsymbol{x} \sim g(\boldsymbol{x})} \mathbb{E}_{y \sim f_t(\boldsymbol{x})} H(y, f_{t+1}(\boldsymbol{x})) \; . \quad (7)$$

The quality of this approximation highly depends on the quality of f_t and g. If f_t is far from an optimal classifier f^* or $g(\boldsymbol{x})$ is far from $P(\boldsymbol{x})$, (7) yields a poor approximation.

The expected risk in (7) smoothens the risk landscape in complex ways beyond simple Gaussian smoothing and interpolation. This smoothing is applicable to any continuous, discrete, or structured domain as long as expressive generative models of $P(\boldsymbol{x})$ are available. That said, for almost all reasonable loss functions H (*e.g.,* softmax cross entropy and squared error), (7) is minimized when $f_{t+1} = f_t$, which is not ideal, especially when f_t is far from f^*. On the other hand, empirical risk (5) anchors the problem in real labeled examples that are provided as ground truth.

GeST aims to combine the benefits of (5) and (7) via:

$$\begin{aligned} R_t(f_{t+1}) \; = \; & \frac{\lambda}{N} \sum_{i=1}^{N} H(y_i, f_{t+1}(\boldsymbol{x}_i)) \\ & (1-\lambda) \mathbb{E}_{\boldsymbol{x} \sim g(\boldsymbol{x})} \mathbb{E}_{y \sim f_t(\boldsymbol{x})} H(y, f_{t+1}(\boldsymbol{x})) \end{aligned} \quad (8)$$

In this formulation, if f_t represents the minimizer of empirical risk (5), then $f_{t+1} = f_t$ is the minimizer of (8) too. However, one does not seek the global minimizer of empirical risk, but rather the best performance on heldout data. If f_t is obtained by stochastic gradient descent on any risk function, but early stopped according to empirical risk on a heldout set, then using such f_t in (8) to define $R_t(f_{t+1})$ promotes the selection of a mapping f_{t+1} that minimizes empirical risk, but also stays as close as possible to the best performing mapping so far (*i.e.,* f_t). This formulation motivates self-training and GeST as regularizers in the functional space and explains why they can conceivably work.

Class-conditional generative models. One can also factorize the joint distribution $P(\boldsymbol{x}, y)$ as $P(y)P(\boldsymbol{x} \mid y)$ and accordingly utilize a class-conditional generative model $g(\boldsymbol{x} \mid y)$ to derive the following expected risk formulation:

$$R(f) = \mathbb{E}_{y \sim P(y)} \mathbb{E}_{\boldsymbol{x} \sim g(\boldsymbol{x})} H(y, f_{t+1}(\boldsymbol{x})) \; . \quad (9)$$

In this setting pseudo labeling is not needed as synthetic data is already labeled. One can show that the optimal classifier f_g^* that minimizes (9) for a cross entropy loss is given by,

$$f_g^*(y \mid \boldsymbol{x}) = \frac{g(\boldsymbol{x}|y)P(y)}{\sum_{y'} g(\boldsymbol{x}|y')P(y')} \; , \quad (10)$$

that is turning the class-conditional generative model into a classifier by using the Bayes rule yields the optimal solution.

Provided that the accuracy of generative classifiers on natural image and text classification is far behind their discriminate counterparts (*e.g.,* Ravuri & Vinyals (2019)), we think substituting (9) into (8) is not a good idea. Essentially, by substituting (9) into the classification objective, one is regularizing f to remain close to f_g^*, which is not an effective strategy if f_g^* is not competitive. This argument corroborates the evidence from recent work that using class-conditional generative models to augment supervised learning does not provide big gains (Ravuri & Vinyals, 2019).

4.2. Unconditional Domain-Specific Generative Models

Text. Many NLP tasks have a relatively small labeled dataset (Wang et al., 2019b;a). While self-supervised pretraining, followed by supervised fine-tuning (Devlin et al., 2019; Liu et al., 2019b; Clark et al., 2020; Lewis et al., 2019) has become the prominent approach to NLP, previous work has also investigated different data augmentation methods to increase the size of the training datasets (Wang & Yang, 2015; Yu et al., 2018; Wu et al., 2019; Lichtarge et al., 2019; Du et al., 2020). In summary, these papers approach data augmentation for NLP through the lens of lexicon replacement, sentence retrieval, and round-trip machine translation.

Inspired by prior work, we propose the use of unconditional language models for data augmentation, which provides a higher degree of simplicity, flexibility, and expressively. We take a pretrained GPT-2 language model (Radford et al., 2019) and fine-tune it separately on each dataset of interest after removing class labels. We find that training from scratch on these datasets is hopeless, but the larger the pretrained GPT-2 variant, the better the validation perplexity scores are. For tasks modeling a relationship between multiple sentences, we concatenate a separator token "[SEP]" between consecutive sentences. Once a fine-tuned GPT-2 model is obtained, we generate task-specific synthetic data up to $40\times$ larger than the original training sets. For some samples of generated text for GLUE see Table A.1 to A.4.

Tabular data. Features from tabular tasks are often in a well-structured format, *i.e.,* each data point comprises a fixed number of attributes as in Table B.2. This property impedes the acquisition of in-domain unlabeled data, *i.e.,* most augmentation techniques such as round-trip translation and retrieval are hardly applicable. To enable generative modeling on tabular data, we convert each data point (*i.e.,* row from the table) into a sentence by concatenating all of its attributes. This reformatting enables the use of GPT-2 fine-tuning similar to text.

Model	SST-2	QQP	QNLI	RTE	MNLI	MRPC	CoLA	STS-B	Avg
RoBERTa base	94.8	91.5	92.6	78.8	87.7	90.1	63.6	90.8	86.2
+ GeST (iter 1)	95.3	91.8	93.1	81.4	87.9	91.7	65.1	91.4	87.2
+ GeST (iter 2)	95.3	91.7	93.2	82.4	88.0	92.2	65.2	91.5	**87.4**
+ GeST (iter 3)	95.3	91.7	93.2	82.0	87.9	92.2	65.5	91.7	**87.4**
RoBERTa base + self-distillation	95.2	91.5	93.1	79.7	88.1	90.3	63.7	90.4	86.5

Table 1. RoBERTa base and GeST results with few iterations on GLUE dev sets. Reported results are the average of 5 independent runs.

Model	SST-2	QQP	QNLI	RTE	MNLI	MRPC	CoLA	STS-B	Avg
BERT large (Devlin et al., 2019)	93.2	91.3	92.3	70.4	86.6	88.0	60.6	90.0	84.1
RoBERTa large (Liu et al., 2019b)	96.4	92.2	93.9	86.6	90.2	90.9	68.0	92.4	88.8
XLNET large (Yang et al., 2019)	**97.0**	92.3	94.9	85.9	90.8	90.8	69.0	92.5	89.2
ELECTRA large (Clark et al., 2020)	96.9	**92.4**	95.0	88.0	90.9	90.8	69.1	92.6	89.5
DeBERTa large (He et al., 2020)	96.8	92.3	**95.3**	88.3	**91.1**	91.9	70.5	**92.8**	89.9
RoBERTa large + GeST (iter 3)	96.9	92.1	94.7	**90.1**	90.7	**93.0**	**70.8**	92.2	**90.1**

Table 2. RoBERTa large and GeST results (average of 5 runs) on GLUE dev sets in comparison with strong recent baselines.

Images. To investigate the applicability of GeST to vision applications, we adopt an unconditional generative model that is trained from scratch on the input data from CIFAR-10 and Fashion MNIST. Given the low resolution nature and characteristics of these datasets we opted not to fine-tune pretrained generative models. As the generative model, we use the Noise Conditional Score-matching Network (NCSN) of Song & Ermon (2019), which was shown to generate high fidelity samples competitive with state-of-the-art autoregressive models, yet more efficient at sampling time. We adopt a pretrained NCSN model on CIFAR-10 provided by the authors' github repo, and train a new NCSN model on Fashion MNIST. Some samples along with their corresponding pseudo labels are shown in Figures A.1-2. We note that GeST is not tied to a specific generative model, we intend to compare different generative model families for GeST.

5. Experiments

We asses the effectiveness of GeST on NLP, tabular, and computer vision tasks. We conclude by investigating the use of generated data for knowledge distillation.

5.1. NLP Tasks and Ablation Studies

We use the GLUE benchmark (Wang et al., 2019b) for our NLP experiments (see Appendix B for benchmark details). To generate domain-specific synthetic data, we fine-tune GPT-2-large on the training set of each downstream task, excluding labels. For tasks with multiple input sentences, we concatenate input sentences into a long sequences and separate sentences by special [SEP] tokens. We generate new domain-specific data by using top-k random sampling similar to Radford et al. (2019). We do not feed any prompt to the LM, but a special [BOS] token to initiate the generation chain. A generation episode is terminated when a special [EOS] token is produced. We generate diverse sentences by varying the random seed. After collecting enough synthetic data, we only retain unique sentences. For tasks with α input sentences, we discard generated samples that violate this constraint (approximately 10% of samples were rejected). Our final synthetic unlabeled dataset U includes $40\times$ as many examples as the original dataset for each task.

GeST. We fine-tune pretrained RoBERTa models provided by fairseq (Ott et al., 2019) on each task. Fine-tuned RoBERTa serves as the first teacher model for self-training. Each student model is initialized with the original pretrained RoBERTa. We combine the labeled dataset L and the synthetic dataset U with a ratio of 1:1, by oversampling labeled data. This corresponds to $\lambda = 0.5$ in Eq. (8).

Table 1 shows that GeST provides an average improvement of +1.2% over RoBERTa-base. We see consistent improvements with more GeST iterations, but performance saturates after three iterations. We further compare our approach with a self-distillation (Furlanello et al., 2018) baseline, in which the teacher and student models use the same architecture and transfer knowledge via the original labeled training set. Although self-distillation provides a slight improvement, the gains from GeST are more significant. Finally, we apply 3 iterations of GeST to RoBERTa-large and compare with state-of-the-art techniques in Table 2. We observe that RoBERTa-large + GeST outperforms strong recent techniques in terms of average performance on the GLUE tasks.

In what follows, we conduct an in-depth ablation of different components of GeST. Unless stated otherwise, we use a RoBERTa-base model with a combination of the original training data and $40\times$ synthetic data for each experiment.

Synthetic dataset size. Deep neural networks typically benefit from large training datasets (Koehn & Knowles, 2017). Because we use a generative model to synthesize data, we can use as much synthetic data as practically possible given our computational budget. To investigate the impact of syn-

thetic dataset size on GeST, we vary the synthetic dataset size from $1\times$ to $40\times$ of the labeled dataset. We also study the use of synthetic data only, without mixing it with the original labeled dataset. Table 3 shows that for both GeST and synthetic data only settings, larger synthetic datasets translate to better performance. On the other hand, the use of synthetic data only, without mixing in the labeled dataset, does not consistently outperform the RoBERTa baseline.

Setup	SST-2	RTE	MRPC	CoLA
RoBERTa base	94.8	78.8	90.1	63.6
Synthetic-only $1\times$	94.9	73.1	88.7	56.1
Synthetic-only $5\times$	94.9	76.5	90.0	59.1
Synthetic-only $10\times$	95.0	77.6	91.1	59.2
Synthetic-only $40\times$	95.1	80.3	90.7	59.9
GeST $1\times$	95.3	79.1	90.0	63.6
GeST $5\times$	95.3	80.5	91.0	64.9
GeST $10\times$	95.2	80.5	91.3	65.0
GeST $40\times$	95.3	81.4	91.7	65.1

Table 3. The impact of synthetic dataset size on GLUE dev set results. Synthetic dataset size is $k\times$ of the original dataset. GeST leverages both synthetic unlabeled data and labeled data.

Soft v.s. hard pseudo label. We investigate the use of soft and hard pseudo labels within the GeST framework for NLP. The results in Table 4 suggest that GeST using soft pseudo labels is more effective than hard labels on the GLUE benchmark. This finding is compatible with the intuition that soft labels enable measuring the functional similarity of neural networks better (Hinton et al., 2015).

Pseudo label	SST-2	RTE	MRPC	CoLA
hard	95.0	80.7	90.8	63.0
soft	95.3	81.4	91.7	65.1

Table 4. GeST with soft *v.s.* hard pseudo labels on GLUE dev sets.

Class-conditional synthetic data generation. Previous work (Kumar et al., 2020b; Ravuri & Vinyals, 2019) suggests that it is challenging to utilize synthetic data from class-conditional generative models to boost the accuracy of text and image classifiers. Our theory in Section 4.1 points to the potential drawback of class-conditional synthetic data. We empirically study this phenomenon, by fine-tuning GPT-2 in a class-conditional manner. Table 5 shows that not only class-conditional LMs underperform unconditional LMs (GeST), but also they are much worse than the baseline.

Source of synthetic data	SST-2	RTE	MRPC	CoLA
No synthetic data (baseline)	94.8	78.8	90.1	63.6
Class-conditional LM	92.9	74.4	86.0	58.4
Unconditional LM (GeST)	95.3	81.4	91.7	65.1

Table 5. Synthetic data from class-conditional LMs underperforms GeST and original RoBERTa base on GLUE dev sets.

GPT-2 model size. Radford et al. (2019) present a few variants of the GPT-2 model including *GPT-2*, *GPT-2-medium*, and *GPT-2-large*. Larger GPT-2 models yield better perplexity scores and higher generation quality. We utilize these models within the GeST framework to study the impact of the generative model's quality on downstream task's performance. Table 6 shows that SST-2 and RTE datasets are not sensitive to the capacity of the GPT-2 model, but higher quality synthetic text improves the results on MRPC and CoLA datasets. We leave investigation of GPT-2-XL and larger language models to future work.

GPT-2	SST-2	RTE	MRPC	CoLA
small	95.5	81.3	90.9	63.9
medium	95.3	81.3	91.3	63.7
large	95.3	81.4	91.7	65.1

Table 6. GeST with various GPT-2 model sizes on GLUE dev sets.

Quality of synthetic dataset. An effective generative model of text should learn the wording and genre of a given corpus, but still produce novel sentences. In order to study the characteristics of our synthetic datasets, Table 8 reports the number of unique n-grams in the training and synthetic datasets, as well as the number of unique n-grams shared between them. The high degree of overlap on uni-grams suggests that the fine-tuned GPT-2 is somewhat domain-specific. Meanwhile, the large number of unique n-grams in the synthetic dataset suggests that many novel word combinations are generated, which is helpful for GeST.

5.2. Tabular Tasks

We consider two tabular tasks, namely connect-4 (Burton & Kelly, 2006) and Drug Review (Gräßer et al., 2018). The details of these tasks can be found in Appendix B. We follow the same protocol as NLP tasks and generate $40\times$ unlabeled data from a fine-tuned GPT-2. Table 7 shows that GeST achieves decent gains on these tasks even though neither RoBERTa nor GPT-2 are optimized for tabular tasks. XGBoost (Chen & Guestrin, 2016), a strong supervised baseline for tabular tasks underperforms RoBERTa+GeST on connect-4. Since a few attributes from Drug Review include free form text, we do not apply XGBoost to Drug Review. We believe GeST can be successfully combined with XGBoost, but we leave this to future work.

Model	connect-4	Drug Review
XGBoost	86.0	-
RoBERTa base	85.0	84.6
+ GeST (iter 1)	87.0	85.7
+ GeST (iter 2)	87.5	85.8
+ GeST (iter 3)	87.3	85.6

Table 7. RoBERTa-base and GeST results on the connect-4 and Drug Review datasets from the UCI repository.

	SST-2	QNLI	RTE	MRPC	CoLA
1-gram	(15k, 33k, 11k)	(89k, 231k, 55k)	(18k, 34k, 13k)	(15k, 27k, 10k)	(6k, 6k, 4k)
3-gram	(107k, 2M, 38k)	(2M, 10M, 513k)	(120k, 750k, 30k)	(105k, 562k, 27k)	(39k, 198k, 14k)
5-gram	(109k, 4M, 9k)	(2M, 25M, 146k)	(130k, 1M, 4k)	(120k, 1M, 7k)	(35k, 389k, 5k)

Table 8. For each dataset we report the number of unique n-grams in (the original dataset, the synthetic dataset, shared between the two).

Model	Data	SST-2	QQP	QNLI	RTE	MNLI	MRPC	CoLA	STS-B	Avg
BERT base	Original	93.2	89.7	91.6	67.1	84.6	87.9	58.3	88.1	82.6
DistilBERT	Original	91.1	88.7	88.4	60.3	82.4	87.7	52.8	86.8	79.8
BERT-PKD	Original	91.3	88.4	88.4	66.5	81.3	85.7	45.5	86.2	79.2
BERT-Theseus	Original	91.5	89.6	89.5	68.2	82.3	**89.0**	51.1	**88.7**	81.2
DistilBERT	GeST	**92.1**	**89.7**	**90.6**	**70.4**	**83.6**	88.6	**56.6**	88.1	**82.5**

Table 9. Knowledge Distillation results on GLUE dev sets with different models. All models use 6-layer transformer, except BERT base.

5.3. Computer Vision

We assess the effectiveness of GeST on CIFAR-10 (Krizhevsky & Hinton, 2009) and Fashion MNIST (Xiao et al., 2017). We adopt the NCSN model of Song & Ermon (2019) as the generative model of GeST. We use the CIFAR-10 model provided by the authors and train a model on Fashion MNIST using the same configuration as CIFAR-10. We select the model checkpoint resulting in the best FID score (Heusel et al., 2017) based on 1000 samples. We then use the NCSN models to generate up to $10\times$ synthetic unlabeled data, *i.e.,* 500K for CIFAR-10 and 600K for Fashion MNIST. See Appendix A for representative samples.

GeST. We use a variant of FixMatch (Sohn et al., 2020) to conduct self-training for the vision tasks. Specifically, we train a classifier on minibatches of intertwined original labeled and synthetic unlabeled data. In each iteration, we obtain pseudo-labels for the synthetic data, but filter unlabeled examples based on classifier's confidence, *i.e.,* examples are kept on which the largest class probability exceeds τ. Weak augmentation is used to define pseudo labels, but strong augmentations are used to obtain student model's predictions. We randomly sample from the strong augmentations list defined in RandAugment (Cubuk et al., 2020). Unlike FixMatch, we only apply strong augmentations to the synthetic samples and not the original labeled data to ensure a fairer comparison with the baseline.

We conduct experiments on three different convolutional neural network architectures: VGG19 (Simonyan & Zisserman, 2014), WideResnet28-2 (Zagoruyko & Komodakis, 2016), and ResNet110 (He et al., 2016). For the full list of hyperparameters and other implementation details, please refer to Appendix C. Each classifier is trained for 200 epochs and 3 synthetic datasets of size ($1\times$, $5\times$, $10\times$) of the training dataset are used.

Table 10 shows that GeST achieves an average error reduction of 0.8% over the baseline on CIFAR-10 across the 3 architectures tested. Further, it appears that the larger the synthetic dataset size, the better the performance of GeST. We note that the reported results are the average of 3 independent runs. Similarly on Fashion MNIST, we witness consistent gains across all architectures. Fashion MNIST results are included in Appendix D. Our computer vision experiments confirm that even when the generative model is not pretrained on open domain data and solely trained on the dataset at hand, GeST can achieve significant improvements.

Model **# params**	VGG19 1.74M	WRN28-2 1.98M	ResNet110 20.11M
Baseline	6.62	4.93	5.85
GeST $1\times$	5.97	4.52	5.13
GeST $5\times$	5.80	4.41	5.11
GeST $10\times$	**5.65**	**4.31**	**5.10**

Table 10. Classification error rates on CIFAR-10 test set with varying amounts of synthetic data for three different model architectures. Reported results are the average of 3 independent runs.

5.4. Knowledge Distillation

The goal of knowledge distillation (KD) (Buciluǎ et al., 2006; Hinton et al., 2015) is to distill the knowledge of a powerful teacher model into a compact student model with as little loss in performance as possible. This can help with model compression (Jiao et al., 2019; Sun et al., 2019) and multi-task learning (Liu et al., 2019a; Clark et al., 2019). It is known that knowledge distillation on fresh data, unseen during training, performs better (Buciluǎ et al., 2006; Chen et al., 2020c) than knowledge distillation on original training data. Accordingly, we investigate the effectiveness of knowledge distillation using generated unlabeled data.

We use the HuggingFace implementation (Wolf et al., 2020) for KD experiments on NLP and adopt a standard experimental setup consistent with previous work (Sun et al., 2019; Xu et al., 2020). A fine-tuned BERT base model (12-layer transformer) (Devlin et al., 2019) represents the teacher and

a DistilBERT model (6-layer transformer) (Sanh et al., 2019) is used as the student. Similar to GeST, we train the student model on U and L, where U is annotated by a fixed teacher. Table 9 shows that GeST dramatically surpasses all existing KD baselines, including DistilBERT (Sanh et al., 2019), BERT-PKD (Sun et al., 2019) and BERT-Theseus (Xu et al., 2020). All of the baselines use the same student architecture. This marks a new state-of-the-art for KD on NLP.

6. Conclusion

We present Generative Self-Training (GeST): a framework for self-training with generated unlabeled data. We motivate GeST from an expected risk minimization perspective and demonstrate both theoretically and empirically that the use of unconditional generative models for synthetic data generation is more effective than class-conditional generative models, previously used in the literature. GeST leverages advances in deep generative models to help supervised learning and can have implications for learning from limited labeled data. GeST works surprisingly well on NLP, tabular, and vision tasks, and helps improve knowledge distillation. We hope that GeST will stimulate new research on the evaluation and development of deep generative models.

References

Agrawala, A. Learning with a probabilistic teacher. *IEEE Transactions on Information Theory*, 16(4):373–379, 1970. doi: 10.1109/TIT.1970.1054472.

Antoniou, A., Storkey, A., and Edwards, H. Data augmentation generative adversarial networks. *arXiv:1711.04340*, 2017.

Berthelot, D., Carlini, N., Cubuk, E. D., Kurakin, A., Sohn, K., Zhang, H., and Raffel, C. Remixmatch: Semi-supervised learning with distribution alignment and augmentation anchoring. *arXiv:1911.09785*, 2019a.

Berthelot, D., Carlini, N., Goodfellow, I., Papernot, N., Oliver, A., and Raffel, C. A. Mixmatch: A holistic approach to semi-supervised learning. *Advances in Neural Information Processing Systems*, pp. 5049–5059, 2019b.

Bowles, C., Chen, L., Guerrero, R., Bentley, P., Gunn, R., Hammers, A., Dickie, D. A., Hernández, M. V., Wardlaw, J., and Rueckert, D. Gan augmentation: Augmenting training data using generative adversarial networks. *arXiv:1810.10863*, 2018.

Brock, A., Donahue, J., and Simonyan, K. Large scale GAN training for high fidelity natural image synthesis. In *International Conference on Learning Representations*, 2019.

Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., et al. Language models are few-shot learners. arxiv 2020. *arXiv:2005.14165*, 4, 2020.

Buciluǎ, C., Caruana, R., and Niculescu-Mizil, A. Model compression. *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 535–541, 2006.

Burton, A. N. and Kelly, P. H. Performance prediction of paging workloads using lightweight tracing. *Future Generation Computer Systems*, 22(7):784–793, 2006.

Carmon, Y., Raghunathan, A., Schmidt, L., Duchi, J. C., and Liang, P. S. In Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems*, volume 32, pp. 11192–11203. Curran Associates, Inc., 2019.

Chapelle, O., Weston, J., Bottou, L., and Vapnik, V. Vicinal risk minimization. *Advances in neural information processing systems*, 2001.

Chapelle, O., Scholkopf, B., and Zien, A. *Semi-Supervised Learning*. MIT, 2009.

Chen, M., Radford, A., Child, R., Wu, J., Jun, H., Luan, D., and Sutskever, I. Generative pretraining from pixels. *ICML*, 2020a.

Chen, T. and Guestrin, C. Xgboost: A scalable tree boosting system. *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pp. 785–794, 2016.

Chen, T., Kornblith, S., Norouzi, M., and Hinton, G. A simple framework for contrastive learning of visual representations. *International conference on machine learning*, pp. 1597–1607, 2020b.

Chen, T., Kornblith, S., Swersky, K., Norouzi, M., and Hinton, G. Big self-supervised models are strong semi-supervised learners. *NeurIPS*, 2020c.

Chen, Y., Wei, C., Kumar, A., and Ma, T. Self-training avoids using spurious features under domain shift. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., and Lin, H. (eds.), *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020d.

Clark, K., Luong, M.-T., Khandelwal, U., Manning, C. D., and Le, Q. Bam! born-again multi-task networks for natural language understanding. *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 5931–5937, 2019.

Clark, K., Luong, M.-T., Le, Q. V., and Manning, C. D. Electra: Pre-training text encoders as discriminators rather than generators. *International Conference on Learning Representations*, 2020.

Cooper, D. B. and Freeman, J. H. On the asymptotic improvement in the out-come of supervised learning provided by additional nonsupervised learning. *IEEE Transactions on Computers*, 1970.

Cubuk, E. D., Zoph, B., Shlens, J., and Le, Q. V. Randaugment: Practical automated data augmentation with a reduced search space. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pp. 702–703, 2020.

Dai, Z., Yang, Z., Yang, F., Cohen, W. W., and Salakhutdinov, R. Good semi-supervised learning that requires a bad gan. *arXiv preprint arXiv:1705.09783*, 2017.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. Bert: Pre-training of deep bidirectional transformers for language understanding. *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)*, pp. 4171–4186, 2019.

Dolhansky, B., Howes, R., Pflaum, B., Baram, N., and Ferrer, C. C. The deepfake detection challenge (dfdc) preview dataset. *arXiv:1910.08854*, 2019.

Du, J., Grave, E., Gunel, B., Chaudhary, V., Celebi, O., Auli, M., Stoyanov, V., and Conneau, A. Self-training improves pre-training for natural language understanding. *arXiv:2010.02194*, 2020.

Fralick, S. Learning to recognize patterns without a teacher. *IEEE Transactions on Information Theory*, 1967.

Furlanello, T., Lipton, Z., Tschannen, M., Itti, L., and Anandkumar, A. Born again neural networks. *International Conference on Machine Learning*, pp. 1607–1616, 2018.

Gao, T., Fisch, A., and Chen, D. Making pre-trained language models better few-shot learners. *arXiv:2012.15723*, 2020.

Gräßer, F., Kallumadi, S., Malberg, H., and Zaunseder, S. Aspect-based sentiment analysis of drug reviews applying cross-domain and cross-data learning. *Proceedings of the 2018 International Conference on Digital Health*, pp. 121–125, 2018.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

He, P., Liu, X., Gao, J., and Chen, W. Deberta: Decoding-enhanced bert with disentangled attention. *arXiv:2006.03654*, 2020.

Hendrycks, D. and Gimpel, K. A baseline for detecting misclassified and out-of-distribution examples in neural networks. *ICLR*, 2016.

Hernandez, D., Kaplan, J., Henighan, T., and McCandlish, S. Scaling laws for transfer, 2021.

Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., and Hochreiter, S. GANs trained by a two time-scale update rule converge to a local nash equilibrium. *Advances in neural information processing systems*, pp. 6626–6637, 2017.

Hinton, G., Vinyals, O., and Dean, J. Distilling the knowledge in a neural network. *arXiv:1503.02531*, 2015.

Jiao, X., Yin, Y., Shang, L., Jiang, X., Chen, X., Li, L., Wang, F., and Liu, Q. Tinybert: Distilling bert for natural language understanding. *arXiv:1909.10351*, 2019.

Karras, T., Aila, T., Laine, S., and Lehtinen, J. Progressive growing of gans for improved quality, stability, and variation. *arXiv:1710.10196*, 2017.

Karras, T., Laine, S., and Aila, T. A style-based generator architecture for generative adversarial networks. *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4401–4410, 2019.

Karras, T., Laine, S., Aittala, M., Hellsten, J., Lehtinen, J., and Aila, T. Analyzing and improving the image quality of stylegan. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8110–8119, 2020.

Kingma, D. P., Mohamed, S., Rezende, D. J., and Welling, M. Semi-supervised learning with deep generative models. *Advances in neural information processing systems*, pp. 3581–3589, 2014.

Koehn, P. and Knowles, R. Six challenges for neural machine translation. *Proceedings of the First Workshop on Neural Machine Translation*, pp. 28–39, 2017.

Krizhevsky, A. and Hinton, G. Learning multiple layers of features from tiny images. 2009.

Kumar, A., Ma, T., and Liang, P. Understanding self-training for gradual domain adaptation. In III, H. D. and Singh, A. (eds.), *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 5468–5479. PMLR, 13–18 Jul 2020a.

Kumar, V., Choudhary, A., and Cho, E. Data augmentation using pre-trained transformer models. *arXiv:2003.02245*, 2020b.

Lee, D.-H. et al. Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks. *Workshop on challenges in representation learning, ICML*, 2013.

Lewis, M., Liu, Y., Goyal, N., Ghazvininejad, M., Mohamed, A., Levy, O., Stoyanov, V., and Zettlemoyer, L. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv:1910.13461*, 2019.

Lichtarge, J., Alberti, C., Kumar, S., Shazeer, N., Parmar, N., and Tong, S. Corpora generation for grammatical error correction. *arXiv:1904.05780*, 2019.

Liu, X., He, P., Chen, W., and Gao, J. Improving multi-task deep neural networks via knowledge distillation for natural language understanding. *arXiv:1904.09482*, 2019a.

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., and Stoyanov, V. Roberta: A robustly optimized bert pretraining approach. *arXiv:1907.11692*, 2019b.

McClosky, D., Charniak, E., and Johnson, M. Effective self-training for parsing. *Proceedings of the Human Language Technology Conference of the NAACL, Main Conference*, pp. 152–159, 2006.

McLachlan, G. J. and Ganesalingam, S. Updating a discriminant function on the basis of unclassified data. *Communications in Statistics-Simulation and Computation*, 1982.

Mobahi, H., Farajtabar, M., and Bartlett, P. L. Self-distillation amplifies regularization in hilbert space. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., and Lin, H. (eds.), *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020.

Odena, A. Semi-supervised learning with generative adversarial networks. *arXiv preprint arXiv:1606.01583*, 2016.

Oliver, A., Odena, A., Raffel, C., Cubuk, E. D., and Goodfellow, I. J. Realistic evaluation of deep semi-supervised learning algorithms. *NeurIPS*, 2018.

Ott, M., Edunov, S., Baevski, A., Fan, A., Gross, S., Ng, N., Grangier, D., and Auli, M. fairseq: A fast, extensible toolkit for sequence modeling. *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pp. 48–53, 2019.

Oymak, S. and Gulcu, T. C. Statistical and algorithmic insights for semi-supervised learning with self-training. *CoRR*, abs/2006.11006, 2020. URL https://arxiv.org/abs/2006.11006.

Peters, M., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., and Zettlemoyer, L. Deep contextualized word representations. *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pp. 2227–2237, 2018.

Pham, H., Xie, Q., Dai, Z., and Le, Q. V. Meta pseudo labels. *arXiv:2003.10580*, 2020.

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., and Sutskever, I. Language models are unsupervised multitask learners. 2019.

Raghunathan, A., Xie, S. M., Yang, F., Duchi, J., and Liang, P. Understanding and mitigating the tradeoff between robustness and accuracy. In III, H. D. and Singh, A. (eds.), *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 7909–7919. PMLR, 13–18 Jul 2020.

Ravuri, S. and Vinyals, O. Classification accuracy score for conditional generative models. *Advances in Neural Information Processing Systems*, pp. 12268–12279, 2019.

Riloff, E. Automatically generating extraction patterns from untagged text. *Proceedings of the national conference on artificial intelligence*, pp. 1044–1049, 1996.

Rosenberg, C., Hebert, M., and Schneiderman, H. Semi-supervised self-training of object detection models. *Applications of Computer Vision and the IEEE Workshop on Motion and Video Computing, IEEE Workshop on*, 1: 29–36, 2005.

Salimans, T., Goodfellow, I., Zaremba, W., Cheung, V., Radford, A., and Chen, X. Improved techniques for training gans. *Proceedings of the 30th International Conference on Neural Information Processing Systems*, pp. 2234–2242, 2016.

Sanh, V., Debut, L., Chaumond, J., and Wolf, T. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *ArXiv*, abs/1910.01108, 2019.

Scudder, H. Probability of error of some adaptive pattern-recognition machines. *IEEE Transactions on Information Theory*, 1965.

Simonyan, K. and Zisserman, A. Very deep convolutional networks for large-scale image recognition. *arXiv:1409.1556*, 2014.

Sohn, K., Berthelot, D., Li, C.-L., Zhang, Z., Carlini, N., Cubuk, E. D., Kurakin, A., Zhang, H., and Raffel, C. Fixmatch: Simplifying semi-supervised learning with consistency and confidence. *arXiv:2001.07685*, 2020.

Song, Y. and Ermon, S. Generative modeling by estimating gradients of the data distribution. *Proceedings of the 33rd Annual Conference on Neural Information Processing Systems*, 2019.

Sun, S., Cheng, Y., Gan, Z., and Liu, J. Patient knowledge distillation for bert model compression. *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pp. 4314–4323, 2019.

Van Engelen, J. E. and Hoos, H. H. A survey on semi-supervised learning. *Machine Learning*, 109(2):373–440, 2020.

Vapnik, V. Principles of risk minimization for learning theory. *Advances in neural information processing systems*, 1992.

Wang, A., Pruksachatkun, Y., Nangia, N., Singh, A., Michael, J., Hill, F., Levy, O., and Bowman, S. R. SuperGLUE: A stickier benchmark for general-purpose language understanding systems. *arXiv:1905.00537*, 2019a.

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

Wang, W. Y. and Yang, D. That's so annoying!!!: A lexical and frame-semantic embedding based data augmentation approach to automatic categorization of annoying behaviors using# petpeeve tweets. *Proceedings of the 2015 conference on empirical methods in natural language processing*, pp. 2557–2563, 2015.

Wang, X., Zhang, R., Sun, Y., and Qi, J. Kdgan: Knowledge distillation with generative adversarial networks. *NeurIPS*, 2018.

Wei, C., Shen, K., Chen, Y., and Ma, T. Theoretical analysis of self-training with deep networks on unlabeled data. In *International Conference on Learning Representations*, 2021.

Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., Cistac, P., Rault, T., Louf, R., Funtowicz, M., Davison, J., Shleifer, S., von Platen, P., Ma, C., Jernite, Y., Plu, J., Xu, C., Le Scao, T., Gugger, S., Drame, M., Lhoest, Q., and Rush, A. Transformers: State-of-the-art natural language processing. *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pp. 38–45, October 2020. doi: 10.18653/v1/2020.emnlp-demos.6.

Wu, X., Lv, S., Zang, L., Han, J., and Hu, S. Conditional bert contextual augmentation. *International Conference on Computational Science*, pp. 84–95, 2019.

Xiao, H., Rasul, K., and Vollgraf, R. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv:1708.07747*, 2017.

Xie, Q., Dai, Z., Hovy, E., Luong, M.-T., and Le, Q. V. Unsupervised data augmentation for consistency training. *arXiv preprint arXiv:1904.12848*, 2019.

Xie, Q., Luong, M.-T., Hovy, E., and Le, Q. V. Self-training with noisy student improves imagenet classification. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 10684–10695, 2020.

Xu, C., Zhou, W., Ge, T., Wei, F., and Zhou, M. Bert-of-theseus: Compressing bert by progressive module replacing. *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 7859–7869, 2020.

Yalniz, I. Z., Jégou, H., Chen, K., Paluri, M., and Mahajan, D. Billion-scale semi-supervised learning for image classification. *arXiv:1905.00546*, 2019.

Yang, Z., Dai, Z., Yang, Y., Carbonell, J., Salakhutdinov, R. R., and Le, Q. V. Xlnet: Generalized autoregressive pretraining for language understanding. *Advances in Neural Information Processing Systems*, 32:5753–5763, 2019.

Yarowsky, D. Unsupervised word sense disambiguation rivaling supervised methods. *33rd annual meeting of the association for computational linguistics*, pp. 189–196, 1995.

Yu, A. W., Dohan, D., Luong, M.-T., Zhao, R., Chen, K., Norouzi, M., and Le, Q. V. QANet: Combining local convolution with global self-attention for reading comprehension. *ICLR*, 2018.

Zagoruyko, S. and Komodakis, N. Wide residual networks. In Richard C. Wilson, E. R. H. and Smith, W. A. P. (eds.), *Proceedings of the British Machine Vision Conference (BMVC)*, pp. 87.1–87.12. BMVA Press, September 2016. ISBN 1-901725-59-6. doi: 10.5244/C.30.87.

Zellers, R., Holtzman, A., Rashkin, H., Bisk, Y., Farhadi, A., Roesner, F., and Choi, Y. Defending against neural

fake news. *Advances in Neural Information Processing Systems*, 32:9054–9065, 2019.

Zhang, H., Cisse, M., Dauphin, Y. N., and Lopez-Paz, D. mixup: Beyond empirical risk minimization. *ICLR*, 2018.

Zhang, L., Song, J., Gao, A., Chen, J., Bao, C., and Ma, K. Be your own teacher: Improve the performance of convolutional neural networks via self distillation. *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 3713–3722, 2019a.

Zhang, X., Wang, Z., Liu, D., and Ling, Q. Dada: Deep adversarial data augmentation for extremely low data regime classification. *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 2807–2811, 2019b.

Zoph, B., Ghiasi, G., Lin, T.-Y., Cui, Y., Liu, H., Cubuk, E. D., and Le, Q. Rethinking pre-training and self-training. *Advances in Neural Information Processing Systems*, 33, 2020.

A. Generated Unlabeled Examples Annotated with Pseudo Labels

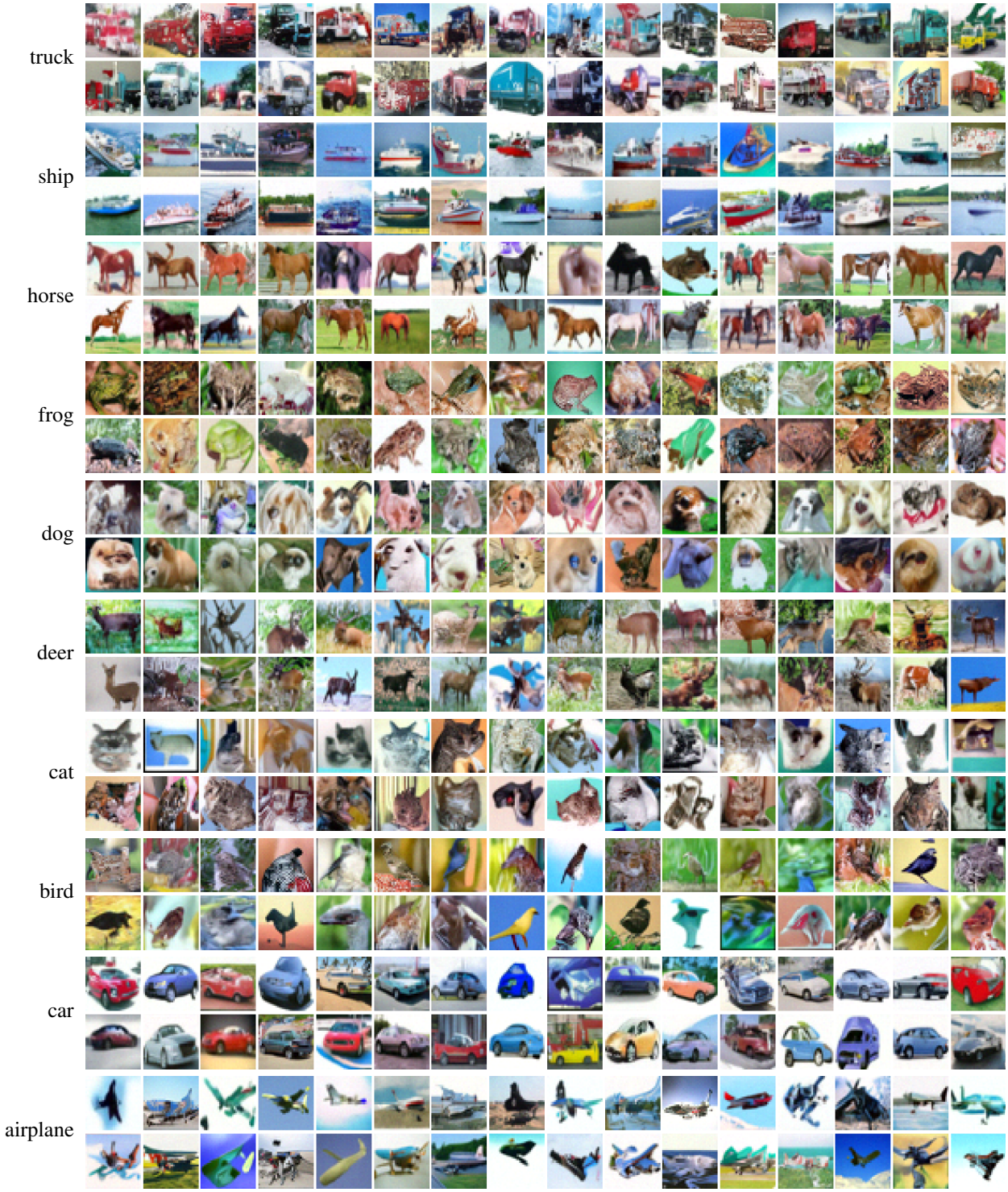

Figure A.1. CIFAR-10 synthetic samples generated by NCSN (Song & Ermon, 2019) and corresponding pseudo-labels. Images are filtered based on a confidence threshold of $\tau = 0.95$ and categorized based on pseudo-labels. For each category, 16 random samples are shown.

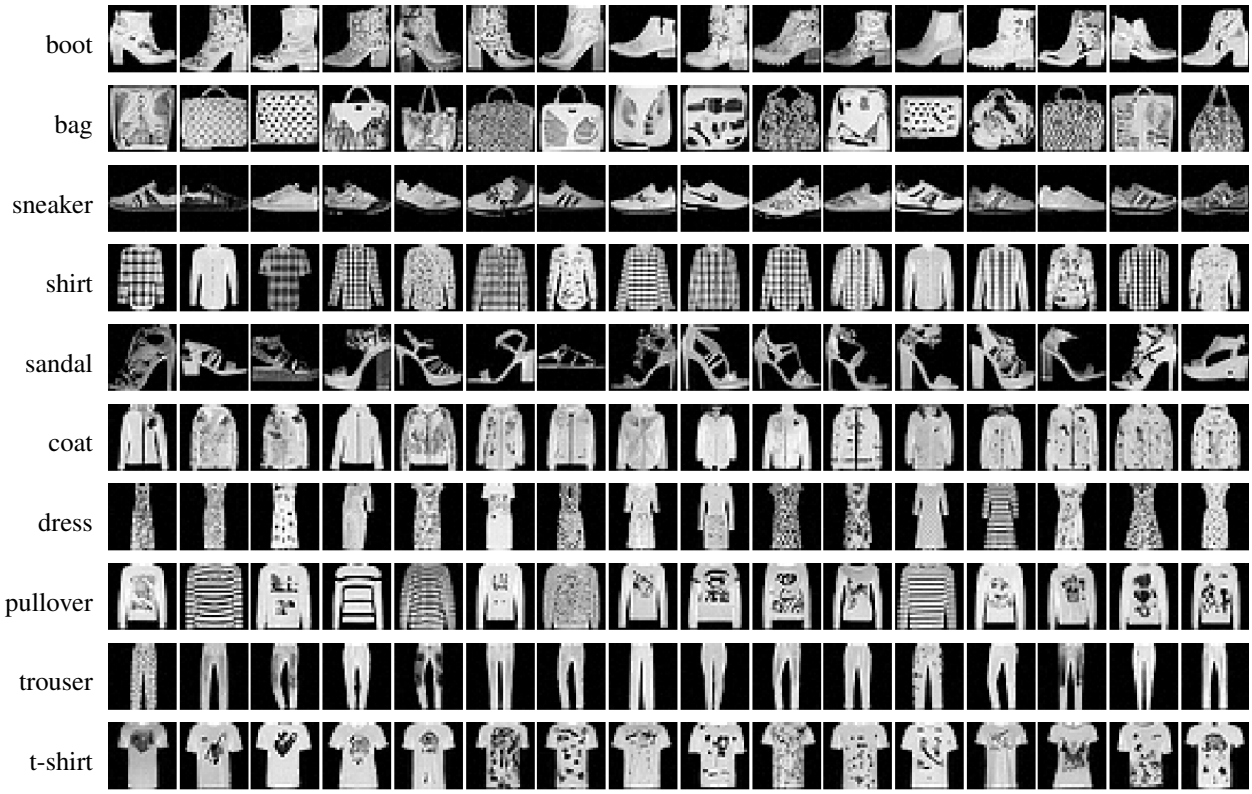

Figure A.2. Fashion MNIST synthetic samples generated by NCSN (Song & Ermon, 2019) and pseudo-labels. Images are filtered based on a confidence threshold of $\tau = 0.95$ and categorized based on pseudo-labels. For each category, 16 random samples are shown.

When did the third Digimon series begin? [SEP] Unlike the two seasons before it and most of the seasons that followed, Digimon Tamers takes a darker and more realistic approach to its story featuring Digimon who do not reincarnate after their deaths and more complex character development in the original Japanese. (**not entailment**)
KNN: 1: What is the name of the third season? [SEP] In addition to the first two seasons, the third season is the season that introduced new characters such as Captain Malice, a supervillain who became the antagonist in season two; and the villains known as the Heartbreakers, who introduced a group of crime fighters. (**not entailment**) 2: When did the "Walking Dead" series end? [SEP] In 2013, AMC announced that it would develop a "superhero series", which would follow the storylines and characters from the "Walking Dead" series in order to bring the popular AMC original series to a new and younger audience. (**not entailment**) 3: What is the main objective of the first season of the X-Files? [SEP] The first season was notable in that the characters were introduced and developed within the space of a single season, as was the format of the show itself. (**not entailment**)
What did Arsenal consider the yellow and blue colors to be after losing a FA Cup final wearing red and white? [SEP] Arsenal then competed in three consecutive FA Cup finals between 1978 and 1980 wearing their "lucky" yellow and blue strip, which remained the club's away strip until the release of a green and navy away kit in 1982–83. (**entailment**)
KNN: 1: Who was the most important player for Arsenal Football Club in the 1950s? [SEP] Wenger continued to use Arsenal's famous red shirts and red kits throughout the 1950s and 1960s, and the red strip became the club's most recognised and recognizable symbol. (**not entailment**) 2: When were the first two teams to play for the trophy in the Premier League? [SEP] The trophy was awarded to Manchester United in 1990-91 and was named after Sir Bobby Charlton, the club's manager until 1990, and later Sir Stanley Matthews, the club's most successful manager. (**not entailment**) 3: What were the last four players to wear the yellow in the final? [SEP] With Arsenal having won all four major trophies in the period, they became the only club to have won five in a row. (**not entailment**)

Table A.1. **QNLI**: Two labeled examples, along with 3 nearest neighbors (based on RoBERTa representations) from our synthetic dataset. We include **labels** for original examples and **pseudo-labels** for synthetic examples in parenthesis.

How is the life of a math student? Could you describe your own experiences? [SEP] Which level of prepration is enough for the exam jlpt5? (**not duplicated**)
KNN: 1: What are the best courses for a mechanical engineering student? [SEP] What is the best course to do after completing a B.Tech in mechanical engineering? (**not duplicated**) 2: How much marks are needed to get through the GATE with electronics? [SEP] What is the average score of the Gate EE exam? What are the cut-offs? (**not duplicated**) 3: What is the best time table for students to prepare for IAS? [SEP] How can one study for IAS in a best time? (**not duplicated**)
How does an IQ test work and what is determined from an IQ test? [SEP] How does IQ test works? (**duplicated**)
KNN: 1: What is the average IQ of the U.S. population? [SEP] How does an IQ test work? (**not duplicated**) 2: Is the Iq test an effective way to measure intelligence? [SEP] How do IQ tests work? (**duplicated**) 3: How is an IQ test on a scale from 1 to 100 scored? [SEP] How do you get your IQ tested? (**not duplicated**)

Table A.2. **QQP**: Two labeled examples, along with 3 nearest neighbors (based on RoBERTa representations) from our synthetic dataset. We include **labels** for original examples and **pseudo-labels** for synthetic examples in parenthesis.

Like the United States, U.N. officials are also dismayed that Aristide killed a conference called by Prime Minister Robert Malval in Port-au-Prince in hopes of bringing all the feuding parties together. [SEP] Aristide had Prime Minister Robert Malval murdered in Port-au-Prince. (**not entailment**)
KNN: 1: The government has been criticized for failing to prevent the mass protests that led to the ouster of President Nicolas Sarkozy earlier this month, which led to his second election defeat since assuming office two years ago. [SEP] Prime Minister Jean-Marc Ayrault is a former president of France. (**not entailment**) 2: The French president, Jacques Chirac, has been urged by both the Vatican and the U.N. Security Council to step up efforts to prevent the return of former dictator Nicolas Sarkozy. [SEP] Nicolas Sarkozy left France. (**not entailment**) 3: The French newspaper Le Monde says the French President Nicolas Sarkozy was advised by U.S. President George W. Bush about a possible trip to Iraq on Thursday. [SEP] Nicolas Sarkozy is a member of the United States. (**not entailment**)
Only a week after it had no comment on upping the storage capacity of its Hotmail e-mail service, Microsoft early Thursday announced it was boosting the allowance to 250MB to follow similar moves by rivals such as Google, Yahoo, and Lycos. [SEP] Microsoft's Hotmail has raised its storage capacity to 250MB. (**entailment**)
KNN: 1: The company, known as Microsoft Office, said it plans to sell all of the copies of its popular Office suite at a loss in the wake of the launch of Microsoft Windows 7, saying it will also make $25 million in advertising costs, a move likely to hurt its long-standing position among consumers and business leaders. [SEP] Microsoft Office is a popular office suite. (**entailment**) 2: The company's shares shot up more than 35% after the company said it has sold all of its remaining inventory of the new Kindle e-readers at $70 each. The shares rose to $65.20 on Wednesday, their highest since March 6, 2011. "The Kindle is our best selling product," said Jeff Bezos, founder and CEO of Amazon.com. [SEP] Amazon.com is based in Seattle. (**not entailment**) 3: In response to concerns expressed by some investors, Microsoft last week said it would reduce the amount of shares that will be available to the public by 10 percent in the first quarter, with a further reduction to 3 percent in the second quarter. The stock price has plunged from $24 to $17, and Microsoft is currently offering $17 to $19 a share to its most senior employees. Some investors had criticized Microsoft's response to concerns about the price of its stock and about the perception that the company is in trouble. [SEP] Microsoft is struggling to sell its stock. (**not entailment**)

Table A.3. **RTE**: Two labeled examples, along with 3 nearest neighbors (based on RoBERTa representations) from our synthetic dataset. We include **labels** for original examples and **pseudo-labels** for synthetic examples in parenthesis.

A BMI of 25 or above is considered overweight ; 30 or above is considered obese . [SEP] A BMI between 18.5 and 24.9 is considered normal , over 25 is considered overweight and 30 or greater is defined as obese . (**paraphrase**)
KNN: 1: The report said that the average woman in her twenties who takes oral contraceptives daily can expect a loss of around 40 per cent of her bone density between the ages of 20 and 45 . [SEP] The study said the average woman in her twenties who used the pill every day , or every day for up to five years , can expect a loss of about 40 per cent of her bone density between the ages of 20 and 45 . (**paraphrase**) 2: The report found that 17 percent of U.S. adults between ages 18 and 64 have a body mass index at or above the " normal " 20 . [SEP] For people of that age , 17.1 percent of adults have a body mass index at or above the " normal " 20 , while 12.6 percent have a body mass index of 30 or above . (**not paraphrase**) 3: The survey shows the proportion of women between 20 and 44 who were obese was 6.3 percent , up from 5.7 percent in 2001 . [SEP] The proportion of women between 20 and 44 who were obese increased to 6.3 percent from 5.7 percent in 2001 . (**paraphrase**)
Shares of Genentech , a much larger company with several products on the market , rose more than 2 percent . [SEP] Shares of Xoma fell 16 percent in early trade , while shares of Genentech , a much larger company with several products on the market , were up 2 percent .(**not paraphrase**)
KNN: 1: Shares in Aventura fell as much as 5 percent , while shares in Medi-Cal climbed 2.5 percent . [SEP] Shares in Aventura were up 2.5 percent , while shares in Medi-Cal rose 2.5 percent . (**paraphrase**) 2: Shares of Amgen rose $ 2.29 , or 2.2 percent , to $ 41.10 in after-hours trading . [SEP] Shares of Amgen , a division of Sanofi-Aventis , rose $ 1.62 , or 1.6 percent , to $ 41.06 in after-hours trading .(**paraphrase**) 3: Shares of General Electric Co . GE.N rose more than 6 percent on the New York Stock Exchange , while shares of PepsiCo Inc . PEP.N rose 4.7 percent . [SPE] General Electric 's shares jumped almost 6 percent on the New York Stock Exchange , while PepsiCo 's climbed 4.7 percent . (**paraphrase**)

Table A.4. **MRPC**: Two labeled examples, along with 3 nearest neighbors (based on RoBERTa representations) from our synthetic dataset. We include **labels** for original examples and **pseudo-labels** for synthetic examples in parenthesis.

One of our number will carry out your instructions minutely. [SEP] A member of my team will execute your orders with immense precision. (**entailment**)
KNN: 1: We are at your disposal to help you with your investigation and provide a full range of pro bono services. [SEP] We are the only ones who can help you with your investigation. (**neutral**) 2: I will speak with the chief officer of the contractor, who will be informed about the results of this effort. [SEP] The contractor is being informed about the results of the effort. (**entailment**) 3: We have an office here to assist you. [SEP] An office is where we will assist you, said the manager. (**neutral**)
Conceptually cream skimming has two basic dimensions - product and geography. [SEP] Product and geography are what make cream skimming work. (**neutral**)
KNN: 1: There are two main types of analysis and they are the case study and the case report. [SEP] The case study is the most popular method used to analyze a subject. (**neutral**) 2: A third approach to capturing and using this type of experience is to engage the program management and finance systems of the organization. [SEP] There are two strategies to capturing and using experience. (**contradiction**) 3: The first is to see the basic elements of a business model in action. [SEP] Basic elements of business models are the most important for the success of any company. (**neutral**)
I don't mean to be glib about your concerns, but if I were you, I might be more concerned about the near-term rate implications of this $1. [SEP] I am concerned more about your issues than the near-term rate implications. (**contradiction**)
KNN: 1: I'm not here to tell you of my own experiences, but they are important to others who might have similar concerns. [SEP] If you were to have similar concerns, I'd like to encourage you to tell them to me. (**neutral**) 2: I don't mean to sound judgmental, but as a person, I think that's an issue you're probably pretty much on your own if you think about it. [SEP] You're probably right if you think about it. (**neutral**) 3: But I don't mean to take your word for it. [SEP] I know you are correct, but I want to make sure it's clear that I do not agree. (**contradiction**)

Table A.5. **MNLI**: Two labeled examples, along with 3 nearest neighbors (based on RoBERTa representations) from our synthetic dataset. We include **labels** for original examples and **pseudo-labels** for synthetic examples in parenthesis.

are more deeply thought through than in most ' right-thinking ' films (**positive**)
KNN: 1: is far more sophisticated , insightful and thought-provoking than his previous films . (**positive**) 2: is more sophisticated than its more obvious and less-than-dazzling counterparts (**positive**) 3: is about as well-thought as the idea of a bad hair day , (**negative**)
contains no wit , only labored gags (**negative**)
KNN: 1: lacks insight , and lacks empathy (**negative**) 2: has little humor or intelligence (**negative**) 3: lacks all wit and humanity (**negative**)

Table A.6. **SST-2**: Two labeled examples, along with 3 nearest neighbors (based on RoBERTa representations) from our synthetic dataset. We include **labels** for original examples and **pseudo-labels** for synthetic examples in parenthesis.

Dataset	**task**	**domain**	**#train**	**#dev**	**#test**	**#classes**
NLP - GLUE Benchmark:						
SST-2	sentiment analysis	movie reviews	67k	872	1.8k	2
QQP	paraphrase	social QA questions	364k	40k	391k	2
QNLI	QA/natural language inference	Wikipedia	105k	5k	5.4k	2
RTE	natural language inference	news, Wikipedia	2.5k	277	3k	2
MNLI	natural language inference	misc.	393k	20k	20k	3
MRPC	paraphrase	news	3.7k	408	1.7k	2
CoLA	acceptability	misc.	8.5k	1043	1k	2
STS-B	sentence similarity	misc.	5.8k	15k	1.4k	–
Tabular Data - UCI:						
connect-4	utility value	gaming	54k	6.8k	6.8k	3
Drug Review	sentiment analysis	medical	2.6k	0.5k	1k	3
Computer Vision:						
CIFAR-10	image classification	real images	50K	N/A	10K	10
Fashion MNIST	image classification	clothing - grey scale	60K	N/A	10K	10

Table B.1. Summary of the three sets of tasks used for evaluation of GeST. STS-B is a regression task, so #classes is not applicable.

	Attributes	**Label**
1	x x o b b b o x o ... b b b x b b b b b	loss
2	b b b b b b b b b ... b b b o x b b b b	win
3	o b b b b b b b b ... b b b o b b b b b	draw

Table B.2. 3 examples of input and labels for the connect-4 tabular task.

B. Datasets

C. Training Details

We use the fairseq codebase (Ott et al., 2019) for implementing both NLP and tabular experiments. Training details are summarized in Table C.1. We use the HuggingFace codebase (Wolf et al., 2020) for KD experiments. All NLP models are trained for 5 epochs with a learning rate of 2e-5 and a batch size of 32.

	SST-2	**QQP**	**QNLI**	**RTE**	**MNLI**	**MRPC**	**CoLA**	**STS-B**	**connect4**	**Drug**
lr	1e-5	1e-5	1e-5	2e-5	1e-5	1e-5	1e-5	2e-5	1e-5	1e-5
#sent.	32	32	32	16	32	16	16	16	32	16
warmup steps	1256	28318	1986	122	7432	137	320	214	1013	116
validate steps	2093	11307	3310	203	12386	203	535	360	1686	212
#epoch	10	10	10	10	10	10	10	10	10	10

Table C.1. Training details for NLP tasks and tabular tasks.

For the CV tasks, we first use the official implementation of NCSN (Song & Ermon, 2019) to generate the synthetic images for CIFAR-10 and Fashion MNIST. We use the pretrained checkpoints provided by the authors for the generation of synthetic CIFAR-10 images and we train a new generative model for Fashion MNIST from scratch with the same hyperparameters of

the CIFAR-10 network. After generating the synthetic images, we apply GeST using a FixMatch-like setup (Sohn et al., 2020), using the hyperparameters listed in Table C.2. We follow Cubuk et al. (2020) for strong augmentations. Finally, the backbone of the classifiers is from this codebase: https://github.com/bearpaw/pytorch-classification.

Parameter	Description	Value
τ	Pseudo-labeling confidence threshold	0.95
batch size	Number of labeled images per batch	64
μ	Ratio between number of unlabeled and labeled images in each batch	7
images per epoch	Number of labeled images per epoch	65536
#epoch	Number of epochs of training	200
lr	learning rate max value (10 epochs warmup then cosine decay)	0.03
weight decay	Weight decay regualrization coefficient	5.00×10^{-4}
momentum	Nesterov momentum for SGD optimizer	0.90

Table C.2. Training details for CV experiments

D. Additional Details

Table D.1 presents GeST results on Fashion MNIST dataset. Similar to CIFAR-10, we observe a performance improvement across the three architectures.

Model **# params**	VGG19 1.74M	WRN28-2 1.98M	ResNet110 20.11M
Baseline	5.41	4.92	5.21
GeST 1×	5.06	**4.63**	**4.74**
GeST 5×	5.14	4.85	4.85
GeST 10×	**4.90**	4.74	4.75

Table D.1. Classification error rates on Fashion MNIST test set with varying amounts of synthetic data for three different model architectures. Results reported are the average over 3 independent runs.

In Tables D.2, and D.3, we present some descriptive statistics of our CIFAR-10 synthetic image dataset to complement the samples shown in Figure A.1 and to help shed some light on the nature of the images generated by the NCSN network.

		Confidence	
Class	**Count**	**Mean**	**Std**
truck	64519	0.932	0.141
ship	32800	0.912	0.156
horse	39604	0.916	0.158
frog	76194	0.887	0.168
dog	38784	0.826	0.188
deer	38829	0.865	0.183
cat	65969	0.826	0.185
bird	37255	0.806	0.193
car	36264	0.936	0.140
airplane	69782	0.897	0.161

Table D.2. Unfiltered CIFAR-10 synthetic data statistics sorted by *Count*. The *Class* pseudo-label for each synthetic image is first obtained using a teacher model trained on the original CIFAR-10 data. *Count* denotes the number of images per class in the entire synthetic dataset (500K images). *Confidence* statistics shows the mean and standard deviation of the teacher model confidence score aggregated over each class.

Class	Count	Confidence Mean	Confidence Std
truck	48796	0.996	0.009
ship	22741	0.995	0.010
horse	28498	0.996	0.010
frog	45923	0.993	0.012
dog	15984	0.989	0.014
deer	21413	0.993	0.012
cat	26311	0.988	0.014
bird	13440	0.988	0.014
car	28329	0.997	0.008
airplane	43745	0.992	0.012

Table D.3. Filtered CIFAR-10 synthetic data statistics sorted by *Count*. The *Class* pseudo-label is first obtained for each synthetic image using a teacher model trained on the original CIFAR-10 data. The dataset is then filtered based on the teacher confidence score where only images with confidence $\geq \tau = 0.95$ are retained. *Count* denotes the number of images per class in the filtered synthetic dataset. *Confidence* statistics shows the mean and standard deviation of the teacher model confidence score aggregated over each class.