Towards Zero-Shot Knowledge Distillation for Natural Language Processing

Ahmad Rashid[1], Vasileios Lioutas[2]*, Abbas Ghaddar[1], Mehdi Rezagholizadeh[1]
[1]Huawei Noah's Ark Lab, [2]University of British Columbia
ahmad.rashid@huawei.com, contact@vlioutas.com,
abbas.ghaddar@huawei.com, mehdi.rezagholizadeh@huawei.com

Abstract

Knowledge distillation (KD) is a common knowledge transfer algorithm used for model compression across a variety of deep learning based natural language processing (NLP) solutions. In its regular manifestations, KD requires access to the teacher's training data for knowledge transfer to the student network. However, privacy concerns, data regulations and proprietary reasons may prevent access to such data. We present, to the best of our knowledge, the first work on Zero-shot Knowledge Distillation for NLP, where the student learns from the much larger teacher without any task specific data. Our solution combines out-of-domain data and adversarial training to learn the teacher's output distribution. We investigate six tasks from the GLUE benchmark and demonstrate that we can achieve between 75% and 92% of the teacher's classification score (accuracy or F1) while compressing the model 30 times.

1 Introduction

Deep learning based natural learning processing (NLP) systems have become state-of-the-art on many applications such as machine translation (MT) (Vaswani et al., 2017; Lioutas and Guo, 2020), natural language understanding (NLU) (Devlin et al., 2019) and language generation (Brown et al., 2020) among others. These models are increasingly trained on huge corpora and with billions of trainable parameters (Brown et al., 2020). This is prohibitive for deploying them on edge devices as well as maintaining them on servers. Moreover, training and evaluating them leaves a significant environmental footprint (Strubell et al., 2019) wherein avoiding the resource hungry training is very challenging (Ghaddar and Langlais, 2019) and may be unavoidable (Li et al., 2020). Model compression approaches make it feasible to employ current state of the art models on edge devices.

Model compression (Sanh et al., 2019; Jiao et al., 2019) has received a lot of attention in the NLP community due to the aforementioned reasons. Some of the algorithms include model pruning (See et al., 2016), quantization (Shen et al., 2019), low-rank matrix factorization (Sainath et al., 2013; Lioutas et al., 2019) and knowledge distillation (KD) (Buciluǎ et al., 2006; Hinton et al., 2015).

KD is one of the most commonly used, application and model agnostic, compression and ensembling algorithm. It is one of the most widely researched algorithms for compressing transformer based language models (Rogers et al., 2020; Rashid et al., 2021; Passban et al., 2021; Jafari et al., 2021; Wu et al., 2020; Kamalloo et al., 2021). In KD, the student needs to be trained with the teacher's training data so as to prevent loss of accuracy. However, we can not assume this access for many practical problems. Some of the concerns preventing access include data privacy, intellectual property, size and transience (Micaelli and Storkey, 2019). e.g. a model trained on patient health records might be available but the data itself may be inaccessible due to patient privacy.

In computer vision (CV), Zero-shot KD (ZSKD) has been proposed to train a student without using any data. In this context Zero-shot refers to training without using data instead of no training at all. Nayak et al. (2019) propose generating "data impressions" by updating noise using backpropogation until it generates 'valid' teacher logits and then training the student on these data impressions. Chen et al. (2019) use a generator to produce synthetic images and use the teacher as discriminator, observing that for real images the softmax function of the teacher encourages a unimodal distribution. Micaelli and Storkey (2019) use a generator to produce synthetic training samples employing adver-

* Work done during an internship at Huawei Noah's Ark Lab.

Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, pages 6551–6561
November 7–11, 2021.

sarial training to improve the quality. Yoo et al. (2019) generate synthetic data by conditioning a generator on output samples from the teacher and a low dimensional representation of the generated samples. These works assume that there is no data available whatsoever for training the student but they do not transfer to text due to the discrete input space (Krishna et al., 2019). We relax this condition and argue that we can still achieve the goals of ZSKD if we use easy to access out-of-domain (OOD), task agnostic data to aid the process. Krishna et al. (2019) put forth a similar argument, albeit for the problem of model extraction, where they use simple heuristic rules to generate training data for a student, of similar or larger size to the teacher, in order to learn the teacher's output distribution. However, they do not put constraints on the size of the student and even propose a student larger than the teacher.

We study the problem of ZSKD for NLP and adapt an OOD dataset similar to (Krishna et al., 2019). In addition we train a text generator to generate samples which maximize the divergence between the teacher and student output while staying close to the OOD distribution. Our contributions are as follows:

- We present one of the first works in NLP on model compression for NLU models using KD without the teacher's training data or any other task-specific data.
- We present a novel KD algorithm which combines OOD data gathering and adversarial training.
- Our algorithm generalizes to different classification tasks for NLP including sentiment analysis, question answering, entailment etc.
- We present an analysis of our algorithm on Natural Language Inference.
- We demonstrate that our algorithm can be competitive in the general fine-tuning setting.

2 Related Work

Knowledge Distillation

KD (Hinton et al., 2015) is a well-known deep learning technique used to transfer the knowledge from an already trained large teacher model to a smaller student network. KD adds a new loss function to the student's regular training loss over the training labels. This new loss function aims at matching the smoothened output probabilities of the student with those of the teacher. More specifically, the training data is fed into the teacher model and the teacher logits are obtained. These are fed, typically, into a softmax function and the temperature parameter is adjusted to smoothen the resulting label distribution. The training loss function for the KD algorithm is as following:

$$\mathcal{L}_{KD} = \alpha * \mathcal{CE}(y, \sigma(z_s; T=1)) + (1-\alpha) * \mathcal{KL}(\sigma(z_t; T=\tau), \sigma(z_s, T=\tau)) \quad (1)$$

where $\mathcal{CE}$ is the cross-entropy, $\mathcal{KL}$ is the Kullback-Leibler divergence and z_s and z_t are the student and teacher logits respectively. σ is the softmax function, and τ and α the temperature and interpolation weights respectively are hyperparameters.

Zero-shot Knowledge Distillation

ZSKD deals with scenarios in which either no training data is available (e.g. in (Nayak et al., 2019)) or at least teacher's training data is not available (for example due to customer's privacy issues). Lopes et al. (2017) introduce a data-free knowledge distillation approach with the assumption that the teacher's network and some meta-data (i.e. the teacher activation records or statistics on the teacher's training data) are given. This work reconstructs the original training data by tweaking a noise input and trying to recover the given meta-data. We are different from (Lopes et al., 2017) in the sense that our model does not need any meta-data for training. Another case in point is (Nayak et al., 2019) which introduces a data-free knowledge distillation approach with no knowledge about the target data distribution. In this regard, their Zero-shot technique models the softmax output of the teacher using the Dirichlet distribution and then builds the underlying data samples (so called Data Impressions) corresponding to the modeled distribution for the teacher. This approach is infeasible for NLP tasks due to the fact that the input data is discrete and the size of the output softmax can be really large.

One potential practical scenario for NLP can be training students without accessing teacher's training data. In this scenario, we are allowed to use any text corpus in the public domain except the

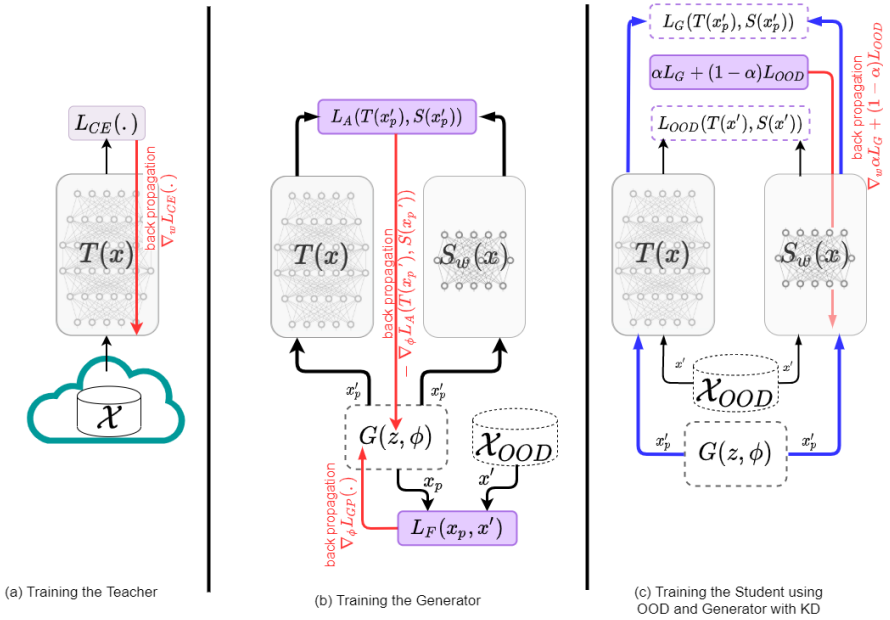

Figure 1: Schematic Diagram of our Zero-shot KD solution. a) We assume access to a pre-trained teacher. b) We gather out-of-domain (OOD) data and train the generator adversarially. c) Finally we use the generated data and the OOD data for KD.

data used for training the teacher network. In this case we can borrow ideas from model extraction techniques such as (Pal et al., 2019; Krishna et al., 2019; Yoo et al., 2019) to facilitate ZSKD training by querying the teacher model using unlabeled data. (Krishna et al., 2019) deal with textual input but do not consider smaller students and the KD scenario. The framework of (Pal et al., 2019) does not apply to pairwise classification tasks. (Yoo et al., 2019) designs a conditional data generator to tackle with lack of training data for the student network and focuses on image classification. However, our solution works on text and our text generator is unconditional.

Adversarial Training

Adversarial examples are small perturbations to training samples indistinguishable to humans but enough to fool neural network classifiers. Goodfellow et al. (2014) proposed adding them to the training set to make CV systems robust to adversarial attacks. Miyato et al. (2016) adapt adversarial training to text classification and improve performance on a few supervised and semi-supervised text classification tasks.

Adversarial training although proposed for model robustness (Ebrahimi et al., 2018; Ghaddar et al., 2021b,a), has been shown to improve state-of-the-art model performance (Cheng et al., 2019; Zhu et al., 2019; Rashid et al., 2021) in NLP. Cheng et al. (2019) study machine translation and propose making the model robust to both source and target perturbations, generated by swapping the word embedding of a word with that of its synonym. They model small perturbations by considering word swaps which cause the smallest increase in loss gradient. Zhu et al. (2019) propose a novel adversarial training algorithm, FreeLB, to make gradient based adversarial training efficient by updating both embedding perturbations and model parameters simultaneously during the backward pass of training. They show improvements on multiple language models on the GLUE benchmark.

Micaelli and Storkey (2019) adapt adversarial training for ZSKD and train an image generator to increase the divergence between student and teacher and train the student to decrease this divergence.

3 Methodology

We rely on an adversarial text generator as the backbone of our method. However, we still need data to pre-train the generator. Since we assume access to a general purpose OOD data, we delineate general principles to extract a training set from this source. Finally, we apply KD on a combination of the OOD training data and the adversarial training data. Figure 1 gives a visual illustration of the

proposed ZSKD method.

3.1 Out-of-Domain Training Data

Our ZSKD method assumes that we do not have the original training data on which the teacher model is trained as well as any other task specific data. Similar to (Krishna et al., 2019), we construct an out-of-domain (OOD) dataset. The idea is that using a general purpose corpus of text, we randomly sample sentences from the text. Then depending on the task we add simple heuristics to make the text suitable for the problems at hand. We summarize a list of targeted tasks all taken from the GLUE benchmark (Wang et al., 2018).

Sentiment Classification (SST-2). We do not modify the sampled sentences for this task but simply feed them to the teacher to get the sentiment output distribution, even though most sentences in the sampled text would have neutral sentiment.

Pairwise Sentence Classification The training sequence typically consists of two input sentences. Depending on the task these can be:

- In Natural Language Inference (NLI), the two input sentences are the hypothesis and the premise. Depending on the task, the goal can be to determine whether the hypothesis is true (entailment), false (contradiction), or undetermined (neutral) given the premise (MNLI) or whether the hypothesis entails the premise in the form of binary classification (RTE). For these tasks, we generate the OOD data by randomly extracting a sentence from the corpus to serve as the premise and then by random chance construct the hypothesis to either be a slightly changed version of the premise or be a completely new random sentence.

- In tasks such as Quora Question Pair (QQP) and Microsoft Research Paraphrase Corpus (MRPC), the goal is to determine if the two input sentences are semantically equivalent or not. We follow a strategy similar to NLI tasks but for the QQP task we post-process the generated sentences by appending a question mark at the end.

Question NLI. The goal of this task is to determine if the given paragraph contains the answer to the input question. We sample a paragraph from our corpus and, randomly, either sample a segment from within the paragraph to form a question or sample an unrelated sentence from the corpus. Then, we randomly append a questioning word such as Who, Where, What etc. to the start of the segment and a question mark at the end.

3.2 Generator Pre-training

Inspired by (Micaelli and Storkey, 2019) and on the promise of adversarial training for NLP (Zhu et al., 2019), the key ingredient of our proposed method is to learn a generator that generates training samples. Our adversarial generation is close to adversarial training and we consider the adversarial samples to be perturbations of the OOD training data D. Therefore we pre-train the generator to follow the distribution of D. Specifically, our generator G is a masked language model, such as BERT, which can generate text from noise such that:

$$x_p = G(z; \phi) \tag{2}$$

where x_p is the output of the generator and is a sequence of tokens, ϕ is the set of generator parameters and $\boldsymbol{z} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{std})$.

The generator is pre-trained by minimizing the following loss function:

$$\mathcal{L}_{GP} = \mathcal{H}_{CE}(x_k, x_p) \tag{3}$$

where $\mathcal{H}_{CE}$ is the cross-entropy loss and x_k is a sample from the OOD training set D. Note that the noise z matches the length and dimension of the embedding of x_k, with the classification token (CLS) added at the beginning and the separator tokens (SEP) inserted at the same locations as in x_k.

3.3 Adversarial Training

Most methods in adversarial training for NLP (Zhang et al., 2020) perturb the word embeddings instead of generating text due to the discreteness problem of text. In order to generate text, we need an argmax operation which breaks end-to-end differentiability. Since our goal is KD, embedding perturbation introduces the problem of size mismatch between the student and teacher embedding. Instead we generate text and sample from the argmax by using the Gumbel-Softmax distribution (Kusner and Hernández-Lobato, 2016; Jang et al., 2016), a continuous distribution over the simplex that can approximate one-hot samples from a discrete distribution.

3.3.1 Adversarial Step

Once pre-trained, the generator is trained with two losses. The first loss maximises the KL-divergence between the teacher and student model on the generated data. The teacher and student model parameters are fixed. The goal is to generate training samples where the teacher and student diverge the most. However, this can lead to degenerate samples which are not useful for transferring teacher knowledge. The second loss is the same as Equation 3 and prevents the generator from diverging too much from the OOD training data. The overall loss, $\mathcal{L}_{\mathcal{T}}$, for generator training is thus:

$$\begin{aligned} \mathcal{L}_A &= -D_{KL}(T(\boldsymbol{x}'_p) \mid\mid S(\boldsymbol{x}'_p)) \\ \mathcal{L}_F &= \mathcal{H}_{CE}(x_k, x_p) \\ \mathcal{L}_T &= \frac{\mathcal{L}_A + \mathcal{L}_F}{2} \end{aligned} \tag{4}$$

where T is the teacher, S is the student, x_k is a sample from the OOD training set, x_p is the softmax output of the generator and x'_p, the one-hot output, is defined as:

$$x'_p = argmax\big(\sigma_{\text{Gumbel}}(x_l)\big) \tag{5}$$

Here σ_{Gumbel} is the Gumbel-Softmax and x_l are the logits of the generator.

3.3.2 Knowledge Distillation

In each training loop we train the generator for n_G steps and the student for n_S steps. Specifically, the student is optimized using a joint KD loss between the data samples generated from the generator G and the data samples coming from the OOD dataset. Overall the student is trained on:

$$\begin{aligned} \mathcal{L}_G &= D_{KL}(T(\boldsymbol{x}'_p) \mid\mid S(\boldsymbol{x}'_p)) \\ \mathcal{L}_{OOD} &= D_{KL}(T(\boldsymbol{x}_k) \mid\mid S(\boldsymbol{x}_k)) \\ \mathcal{L} &= \alpha \cdot \mathcal{L}_G + (1-\alpha) \cdot \mathcal{L}_{OOD} \end{aligned} \tag{6}$$

where x_k and x'_p are as defined above and α is a weight interpolation parameter. Note that unlike regular KD where we have a hard loss and a soft loss, here we have two soft losses. One matches the student and the teacher output on adversarially augmented data and the other on OOD data respectively. Algorithm 1 presents all the steps of our procedure.

Algorithm 1: Zero-shot KD (Complete)

pretrain: $T(\cdot)$
dataset: D
initialize: $G(\cdot;\phi)$
initialize: $S(\cdot;\theta)$

\# Pre-train Generator
for $k \leftarrow 1, 2, ..., N$ **do**
  $\boldsymbol{z} \leftarrow \{\boldsymbol{z_0}, \ldots, \boldsymbol{z_l}\} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{std})$
  $\boldsymbol{x}_k \in D$
  $\boldsymbol{x}_p \leftarrow G(\boldsymbol{z};\phi)$
  $\mathcal{L}_{GP} \leftarrow \mathcal{H}_{CE}(x_k, x_p)$
  $\phi \leftarrow \phi - \lambda \frac{\partial \mathcal{L}_{GP}}{\partial \phi}$
  decay λ
end
\# Adversarial Train
for $k \leftarrow 1, 2, ..., N$ **do**
  $\boldsymbol{x}_k \leftarrow D$
  \# Adversarial Step
  for $1, 2, ..., n_G$ **do**
    $\boldsymbol{z} \leftarrow \{\boldsymbol{z_0}, \ldots, \boldsymbol{z_l}\} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{std})$
    $\boldsymbol{x}_{\text{logits}} \leftarrow G(\boldsymbol{z};\phi)$
    $\boldsymbol{x}_p \leftarrow$ Gumbel-Softmax$(\boldsymbol{x}_{\text{logits}})$
    $\mathcal{L}_A \leftarrow -D_{KL}(T(\boldsymbol{x}'_p) \mid\mid S(\boldsymbol{x}'_p))$
    $\mathcal{L}_F \leftarrow \mathcal{H}_{CE}(x_k, x_p)$
    $\mathcal{L}_T \leftarrow \frac{\mathcal{L}_A + \mathcal{L}_F}{2}$
    $\phi \leftarrow \phi - \eta \frac{\partial \mathcal{L}_T}{\partial \phi}$
  end
  \# Knowledge Distillation
  for $1, 2, ..., n_S$ **do**
    $\boldsymbol{z} \leftarrow \{\boldsymbol{z_0}, \ldots, \boldsymbol{z_l}\} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{std})$
    $\boldsymbol{x}_{\text{logits}} \leftarrow G(\boldsymbol{z};\phi)$
    $\boldsymbol{x}_p \leftarrow$ Gumbel-Softmax$(\boldsymbol{x}_{\text{logits}})$
    $\mathcal{L}_G \leftarrow D_{KL}(T(\boldsymbol{x}'_p) \mid\mid S(\boldsymbol{x}'_p))$
    $\mathcal{L}_{OOD} \leftarrow D_{KL}(T(\boldsymbol{x}_k) \mid\mid S(\boldsymbol{x}_k))$
    $\mathcal{L} \leftarrow \alpha \cdot \mathcal{L}_G + (1-\alpha) \cdot \mathcal{L}_{OOD}$
    $\theta \leftarrow \theta - \eta \frac{\partial \mathcal{L}}{\partial \theta}$
  end
  decay η
end

4 Experiments

We evaluated our proposed adversarial ZSKD approach on six classification tasks from the General Language Understanding Evaluation (GLUE) (Wang et al., 2018) benchmark. The tasks include binary sentiment analysis on the SST-2 dataset (Socher et al., 2013), ternary NLI on MNLI (Williams et al., 2018), binary entailment on the RTE dataset (Bentivogli et al., 2009), semantic equivalence on QQP (Chen et al., 2018) and MRPC (Dolan and Brockett, 2005), and finally question answering adapted to binary classification evaluated on QNLI (Wang et al., 2018). Section 3.1 gives more details about these tasks.

Task	Model	Method	Data Generation	Data Size	Score
SST-2	Teacher	-	Original	67K (×1)	93.0
	Student	-	Original	67K (×1)	87.4
	Student	KD	WikiText-103	269K (×4)	84.9
	Student	KD + Adv (Ours)	WikiText-103	135K (×2)	85.0
	Student	KD + Adv (Ours)	WikiText-103	269K (×4)	**85.9**

Table 1: Results on the single sentence sentiment classification task.

Task	Model	Method	Data Generation	Data Size	Score
MNLI	Teacher	-	Original	392K (×1)	86.6
	Student	-	Original	392K (×1)	75.5
	Student	KD	WikiText-103	1.5M (×4)	62.5
	Student	KD + Adv (Ours)	WikiText-103	785K (×2)	63.8
	Student	KD + Adv (Ours)	WikiText-103	1.5M (×4)	**65.1**
RTE	Teacher	-	Original	2.5K (×1)	70.7
	Student	-	Original	2.5K (×1)	64.2
	Student	KD	WikiText-103	10K (×4)	61.7
	Student	KD + Adv (Ours)	WikiText-103	5K (×2)	62.0
	Student	KD + Adv (Ours)	WikiText-103	10K (×4)	**62.5**

Table 2: Results on the NLI classification tasks.

4.1 Experimental Setup

All models used in this paper, except those in section 4.5, are based on two architecture settings from the BERT (Devlin et al., 2019) model. Specifically, for the teacher model we used the pre-trained version of the $\text{BERT}_{\text{LARGE}}$ model released by the authors. The model consists of 24 layers. The hidden size is 1024 and the number of heads is 16. The total number of parameters is about 340M. For the student model, we decided to use a significantly smaller version of the BERT model. Specifically, we used the $\text{BERT}_{\text{MINI}}$ version which uses 4 layers with 256 hidden dimension and 4 attention heads. The total size of the model is 11M trainable parameters. Both models use a vocabulary of size 30,522 extracted using the Byte Pair Encoding (BPE) (Sennrich et al., 2016) tokenization method. The generator is a pre-trained $\text{BERT}_{\text{MINI}}$ model.

We used the Wikitext-103 (Merity et al., 2016) corpus as our OOD dataset. It is a collection of over 100 million tokens from the set of verified good and featured articles in Wikipedia.

Hyper-parameters We fine-tuned the BERT-based student model for 10 epochs and picked the best checkpoint that gave the lowest loss during training. We report results for all methods on the given Dev set. For each task, we selected the best fine-tuning learning rate among 5e-5, 4e-5, 3e-5, and 2e-5 values. We used the AdamW (Loshchilov and Hutter, 2017) optimizer with the default values. In addition, we used a linear decay learning rate scheduler with no warmup steps. We set the α values from our algorithm to be 0.2 and the std value to 0.01. Additionally, we set the value n_G and n_S (see Algorithm 1) to 10 and 100. Finally, we pre-train the generator for two epochs.

Hardware Details We trained all models using a single NVIDIA V100 GPU. The batch size was set to 64. We used mixed-precision training (Micikevicius et al., 2018) to expedite the training procedure. All experiments were run using the PyTorch[1] framework.

[1] https://pytorch.org/

Task	Model	Method	Data Generation	Data Size	Score
QQP	Teacher	-	Original	363K (×1)	89.9
	Student	-	Original	363K (×1)	83.7
	Student	KD	WikiText-103	1.4M (×4)	70.0
	Student	KD + Adv (Ours)	WikiText-103	728K (×2)	71.7
	Student	KD + Adv (Ours)	WikiText-103	1.4M (×4)	**72.2**
MRPC	Teacher	-	Original	4K (×1)	87.1
	Student	-	Original	4K (×1)	78.5
	Student	KD	WikiText-103	15K (×4)	74.5
	Student	KD + Adv (Ours)	WikiText-103	7K (×2)	75.4
	Student	KD + Adv (Ours)	WikiText-103	15K (×4)	**76.4**

Table 3: Results on the pairwise sentence classification tasks.

Task	Model	Method	Data Generation	Data Size	Score
QNLI	Teacher	-	Original	104K (×1)	91.5
	Student	-	Original	104K (×1)	84.1
	Student	KD	WikiText-103	418K (×4)	78.1
	Student	KD + Adv (Ours)	WikiText-103	209K (×2)	79.1
	Student	KD + Adv (Ours)	WikiText-103	418K (×4)	**79.9**

Table 4: Results on the question NLI task.

4.2 Results

Table 1 presents our result on SST-2. For all the tasks, we present the original large teacher score, the smaller student score when trained on the training data, the student trained with KD on the OOD data and two experiments with different training set sizes using our algorithm. Our baseline is the KD with OOD data and is adapted from (Krishna et al., 2019). They presented the results where the student was the same size as the teacher or larger. Moreover, they applied their method only to SST-2 and MNLI from the GlUE benchmark. We implemented their method, applied it to the smaller student setting and extended the OOD generation process for the 4 other datasets of GLUE.

The data size (x1, x2 and x4) are the OOD data sizes compared to the task specific training data size. The adversarially trained student, in additional to the OOD data, generates an equal number of adversarial examples. On SST-2, we attain close to the student accuracy using the OOD training data. Our method using x2 OOD data does just as well as the baseline but when we use all the OOD data used by the baseline we increase the accuracy by 1%.

The results of the NLI classification tasks, MNLI and RTE, are on Table 2. MNLI is one of the two hardest tasks that we evaluated on. Looking at the accuracy scores we can see that the student trained on the training data falls well short of the teacher. On this task, we can see the strength of our method as the adversarial training improves the score both when we use x2 OOD data and even further when we use x4 OOD data. High model capacity is important for MNLI. We see a similar trend for RTE.

On pairwise sentence classification, on Table 3 we see that MRPC follows a similar trend where the adversarial training algorithm improves the F1 score both when used with x2 OOD data and with x4 OOD data. The same applies for the QQP task. Similar to MNLI, the model capacity and the amount of training data appears to be important for this task. Table 4 presents the result on

Model (Network)	CoLA	SST-2	MRPC	STS-B	QQP	MNLI	QNLI	RTE	Score
RoBERTa-Large (Teacher)	60.5	96.3	89.9	91.7	91.0	89.1	93.0	79.0	82.97
DistilRoBERTa (Student)	56.6	92.7	84.0	87.2	90.8	84.1	91.3	65.7	78.78
DistilRoBERTa + Adv ($\alpha = 0.2$)	62.0	93.1	86.1	88.9	91.9	84.5	91.3	70.7	80.53

Table 5: Results of experiments on GLUE dev set. WNLI results are not presented since they are 56.3 for all models. The are included in calculating the score.

	N	C	E	Overall
OOD Samples	67.2	68.6	51.6	62.5
+0.75M Adv	70.8	66.1	55.7	63.8
+1.5M Adv	73.2	65.3	57.8	65.1

Table 6: Per-class F1 scores on MNLI of students trained on OOD samples with incremental subsets of adversarial examples. N,C and E are Neutral, Contradiction and Entailment respectively. No Adv refer to the student trained on randomized sentence pairs.

the QNLI task and we see improvements using our algorithm both when using half the OOD data as the baseline and when using the same OOD data. On average we see an improvement of 1.4 over all the tasks.

Overall, we were able to recover between 98.2% (SST-2) and 86.2% (MNLI and QQP) of the performance of a version of the student model trained with the original dataset. Similarly, we recovered between 92.3% (SST-2) and 75.1% (MNLI) of the teacher performance.

4.3 Analysis

We inspected the per-class results for MNLI to gain insight into the properties of the adversarially generated samples. Table 6 shows that adding adversarial examples continuously improves the performances on *neutral* and *entailment* classes.

Our manual inspection shows that adding the generator to the loop makes the student more robust on examples where the premise and hypothesis doesn't significantly overlap. The gain could be imputable to the diversity of the adversarial examples, although, the generator may produce a nonsensical sequence of words. We observed that the premise and hypothesis rarely share common words, contrary to heuristically populated examples [2]. Adversarial examples prevent the student from relying on the superficial syntactic properties of OOD samples.

[2]premise and hypothesis are almost identical

Task	Model	LM (GPT-2)	WikiText-103
SST-2	Student+KD	83.1	83.2
MNLI	Student+KD	60.0	60.2

Table 7: Results when using GPT-2 for OOD generation

4.4 Alternative for OOD Generation

We explored the use of a language model for OOD generation. Here instead of sampling OOD data from a corpus, we used distilGPT-2 (Wolf et al., 2019) a lighter version of the GPT-2 (Radford et al., 2019) model. We expected the model to perform slightly better since GPT-2 is trained on a much larger dataset. However, as seen in Table 7, we do not observe any improvement and the algorithm is much slower in comparison due to the complexity of executing a language model in the training loop.

4.5 Fine-tuning Setting

We apply our data generator to fine-tuning a 6-layer transformer model on the GLUE benchmark. In this setting we use the training data and demonstrate that our algorithm can be used for data augmentation. We use $\text{RoBERTa}_{\text{LARGE}}$ (Liu et al., 2019) as the teacher and distilRoBERTa (Sanh et al., 2019) as the generator. The student is also initialized with the weights of a pre-trained distilRoBERTa model. Our baseline is a fine-tuned distilRoBERTa model. We train the student using all the steps in Algorithm 1. The only difference is that since we have access to the labels we apply cross-entropy loss on the training data and KL-divergence on augmented data. Table 5 presents the results and we observe that we can improve the average performance on the baseline by almost 2 points. Note that the baseline is trained with just the cross-entropy loss.

5 Conclusion

We present the first study on Zero-shot Knowledge Distillation (ZSKD) for NLP. We present an algorithm based on OOD data generation and ad-

versarial learning and evaluate on six tasks from the GLUE benchmark reaching to within 75% of the teacher performance on all tasks while attaining a 30x compression. The next steps are to a) explore a generic methodology for OOD data creation and b) study sequence generation tasks such as machine translation and abstractive summarization and achieve compression without the original training data while having access to a teacher.

Acknowledgments

We thank Mindspore[3] which is a new deep learning computing framework for the partial support of this work.

References

Luisa Bentivogli, Peter Clark, Ido Dagan, and Danilo Giampiccolo. 2009. The fifth pascal recognizing textual entailment challenge. In *TAC*.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners.

Cristian Buciluǎ, Rich Caruana, and Alexandru Niculescu-Mizil. 2006. Model compression. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 535–541.

Hanting Chen, Yunhe Wang, Chang Xu, Zhaohui Yang, Chuanjian Liu, Boxin Shi, Chunjing Xu, Chao Xu, and Qi Tian. 2019. Data-free learning of student networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3514–3522.

Z. Chen, H. Zhang, X. Zhang, and L. Zhao. 2018. Quora question pairs.

Yong Cheng, Lu Jiang, and Wolfgang Macherey. 2019. Robust neural machine translation with doubly adversarial inputs. *arXiv preprint arXiv:1906.02443*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

William B. Dolan and Chris Brockett. 2005. Automatically constructing a corpus of sentential paraphrases. In *Proceedings of the Third International Workshop on Paraphrasing (IWP2005)*.

Javid Ebrahimi, Anyi Rao, Daniel Lowd, and Dejing Dou. 2018. Hotflip: White-box adversarial examples for text classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 31–36.

Abbas Ghaddar and Philippe Langlais. 2019. Contextualized word representations from distant supervision with and for ner. In *Proceedings of the 5th Workshop on Noisy User-generated Text (W-NUT 2019)*, pages 101–108.

Abbas Ghaddar, Philippe Langlais, Ahmad Rashid, and Mehdi Rezagholizadeh. 2021a. Context-aware adversarial training for name regularity bias in named entity recognition. *Trans. Assoc. Comput. Linguistics*, 9:586–604.

Abbas Ghaddar, Philippe Langlais, Mehdi Rezagholizadeh, and Ahmad Rashid. 2021b. End-to-end self-debiasing framework for robust NLU training. In *Findings of the Association for Computational Linguistics: ACL/IJCNLP 2021, Online Event, August 1-6, 2021*, volume ACL/IJCNLP 2021 of *Findings of ACL*, pages 1923–1929. Association for Computational Linguistics.

Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. 2014. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*.

Aref Jafari, Mehdi Rezagholizadeh, Pranav Sharma, and Ali Ghodsi. 2021. Annealing knowledge distillation. *arXiv preprint arXiv:2104.07163*.

Eric Jang, Shixiang Gu, and Ben Poole. 2016. Categorical reparameterization with gumbel-softmax. *arXiv preprint arXiv:1611.01144*.

Xiaoqi Jiao, Yichun Yin, Lifeng Shang, Xin Jiang, Xiao Chen, Linlin Li, Fang Wang, and Qun Liu. 2019. Tinybert: Distilling bert for natural language understanding. *arXiv preprint arXiv:1909.10351*.

Ehsan Kamalloo, Mehdi Rezagholizadeh, Peyman Passban, and Ali Ghodsi. 2021. Not far away, not so close: Sample efficient nearest neighbour data augmentation via minimax. *arXiv preprint arXiv:2105.13608*.

[3] https://www.mindspore.cn/

Kalpesh Krishna, Gaurav Singh Tomar, Ankur P Parikh, Nicolas Papernot, and Mohit Iyyer. 2019. Thieves on sesame street! model extraction of bert-based apis. *arXiv preprint arXiv:1910.12366*.

Matt J. Kusner and José Miguel Hernández-Lobato. 2016. Gans for sequences of discrete elements with the gumbel-softmax distribution.

Zhuohan Li, Eric Wallace, Sheng Shen, Kevin Lin, Kurt Keutzer, Dan Klein, and Joseph E Gonzalez. 2020. Train large, then compress: Rethinking model size for efficient training and inference of transformers. *arXiv preprint arXiv:2002.11794*.

Vasileios Lioutas and Yuhong Guo. 2020. Time-aware large kernel convolutions. In *Proceedings of the 37th International Conference on Machine Learning (ICML)*.

Vasileios Lioutas, Ahmad Rashid, Krtin Kumar, Md Akmal Haidar, and Mehdi Rezagholizadeh. 2019. Distilled embedding: non-linear embedding factorization using knowledge distillation. *arXiv preprint arXiv:1910.06720*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Raphael Gontijo Lopes, Stefano Fenu, and Thad Starner. 2017. Data-free knowledge distillation for deep neural networks. *arXiv preprint arXiv:1710.07535*.

Ilya Loshchilov and Frank Hutter. 2017. Decoupled weight decay regularization.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. 2016. Pointer sentinel mixture models. *arXiv preprint arXiv:1609.07843*.

Paul Micaelli and Amos J Storkey. 2019. Zero-shot knowledge transfer via adversarial belief matching. In *Advances in Neural Information Processing Systems*, pages 9547–9557.

Paulius Micikevicius, Sharan Narang, Jonah Alben, Gregory Diamos, Erich Elsen, David Garcia, Boris Ginsburg, Michael Houston, Oleksii Kuchaiev, Ganesh Venkatesh, and Hao Wu. 2018. Mixed precision training. In *International Conference on Learning Representations*.

Takeru Miyato, Andrew M Dai, and Ian Goodfellow. 2016. Adversarial training methods for semi-supervised text classification. *arXiv preprint arXiv:1605.07725*.

Gaurav Kumar Nayak, Konda Reddy Mopuri, Vaisakh Shaj, R Venkatesh Babu, and Anirban Chakraborty. 2019. Zero-shot knowledge distillation in deep networks. *arXiv preprint arXiv:1905.08114*.

Soham Pal, Yash Gupta, Aditya Shukla, Aditya Kanade, Shirish Shevade, and Vinod Ganapathy. 2019. A framework for the extraction of deep neural networks by leveraging public data. *arXiv preprint arXiv:1905.09165*.

Peyman Passban, Yimeng Wu, Mehdi Rezagholizadeh, and Qun Liu. 2021. ALP-KD: attention-based layer projection for knowledge distillation. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021*, pages 13657–13665. AAAI Press.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. *OpenAI Blog*, 1(8):9.

Ahmad Rashid, Vasileios Lioutas, and Mehdi Rezagholizadeh. 2021. MATE-KD: Masked adversarial TExt, a companion to knowledge distillation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1062–1071, Online. Association for Computational Linguistics.

Anna Rogers, Olga Kovaleva, and Anna Rumshisky. 2020. A primer in BERTology: What we know about how BERT works. *Transactions of the Association for Computational Linguistics*, 8:842–866.

Tara N Sainath, Brian Kingsbury, Vikas Sindhwani, Ebru Arisoy, and Bhuvana Ramabhadran. 2013. Low-rank matrix factorization for deep neural network training with high-dimensional output targets. In *2013 IEEE international conference on acoustics, speech and signal processing*, pages 6655–6659. IEEE.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*.

Abigail See, Minh-Thang Luong, and Christopher D Manning. 2016. Compression of neural machine translation models via pruning. *arXiv preprint arXiv:1606.09274*.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.

Sheng Shen, Zhen Dong, Jiayu Ye, Linjian Ma, Zhewei Yao, Amir Gholami, Michael W Mahoney, and

Kurt Keutzer. 2019. Q-bert: Hessian based ultra low precision quantization of bert. *arXiv preprint arXiv:1909.05840*.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1631–1642, Seattle, Washington, USA. Association for Computational Linguistics.

Emma Strubell, Ananya Ganesh, and Andrew McCallum. 2019. Energy and policy considerations for deep learning in nlp. *arXiv preprint arXiv:1906.02243*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2018. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 353–355, Brussels, Belgium. Association for Computational Linguistics.

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1112–1122, New Orleans, Louisiana. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2019. Huggingface's transformers: State-of-the-art natural language processing. *ArXiv*, abs/1910.03771.

Yimeng Wu, Peyman Passban, Mehdi Rezagholizadeh, and Qun Liu. 2020. Why skip if you can combine: A simple knowledge distillation technique for intermediate layers. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 1016–1021. Association for Computational Linguistics.

Jaemin Yoo, Minyong Cho, Taebum Kim, and U Kang. 2019. Knowledge extraction with no observable data. In *Advances in Neural Information Processing Systems*, pages 2701–2710.

Wei Emma Zhang, Quan Z Sheng, Ahoud Alhazmi, and Chenliang Li. 2020. Adversarial attacks on deep-learning models in natural language processing: A survey. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 11(3):1–41.

Chen Zhu, Yu Cheng, Zhe Gan, Siqi Sun, Thomas Goldstein, and Jingjing Liu. 2019. Freelb: Enhanced adversarial training for language understanding. *arXiv preprint arXiv:1909.11764*.