Embedding Hallucination for Few-Shot Language Fine-tuning

Yiren Jian[*]
Dartmouth College
yiren.jian.gr@dartmouth.edu

Chongyang Gao [*]
Northwestern University
cygao@u.northwestern.edu

Soroush Vosoughi
Dartmouth College
soroush@dartmouth.edu

Abstract

Few-shot language learners adapt knowledge from a pre-trained model to recognize novel classes from a few-labeled sentences. In such settings, fine-tuning a pre-trained language model can cause severe over-fitting. In this paper, we propose an Embedding Hallucination (EmbedHalluc) method, which generates auxiliary embedding-label pairs to expand the fine-tuning dataset. The hallucinator is trained by playing an adversarial game with the discriminator, such that the hallucinated embedding is indiscriminative to the real ones in the fine-tuning dataset. By training with the extended dataset, the language learner effectively learns from the diverse hallucinated embeddings to overcome the over-fitting issue. Experiments demonstrate that our proposed method is effective in a wide range of language tasks, outperforming current fine-tuning methods. Further, we show that EmbedHalluc outperforms other methods that address this over-fitting problem, such as common data augmentation, semi-supervised pseudo-labeling, and regularization. The code will be made available at: https://github.com/yiren-jian/EmbedHalluc.

1 Introduction

Fine-tuning a pre-trained language model (LM) on a downstream task with the labeled data has been the de facto approach in many NLP tasks (Wang et al., 2019; Devlin et al., 2019). Conventional fine-tuning has been shown to be effective when a few thousands of labeled examples are available. Data augmentation (Wei and Zou, 2019), regularization (Lee et al., 2019) and re-initialization (Zhang et al., 2021) further improve the results.

However, the performance drops drastically when the number of examples falls to only a few dozens. Experiments from recent work (Gao et al., 2021) have shown that fine-tuning performs poorly in the setting where only 16 examples per class are given. Indeed, tuning a language model with hundreds of millions of parameters (e.g., BERT-large has 300M parameters) with only a few examples inevitably faces the over-fitting problem.

Prior work have proposed regularization methods to overcome this problem (Lee et al., 2019; Zhang et al., 2021). However, we show in our experiments that these methods fail in extreme data scarce setting. We speculate that the key to solve this issue is by data augmentation.

Current common text data augmentation methods, such as EDA (Wei and Zou, 2019) (which have been used in recent few-shot learning papers (Wei et al., 2021; Basu et al., 2021)) and AEDA (Karimi et al., 2021) operate at the lexical level, which while resulting in human readable texts, lead to limited diversity due to the discrete nature of the lexical space. In this work, we propose to use a generative augmentation method at the *embedding space* for few-shot learning. The underlying hypothesis is that the intra-class relation of the observed examples can be modeled and that this can be learned from a few-samples to hallucinate diverse unseen examples. To be specific, we adapt a conditional Wasserstein Generative Adversarial Network (cWGAN) (Arjovsky et al., 2017) as our hallucinator to hallucinate embeddings of sentences. By observing the real embeddings of examples from the fine-tuning dataset, the cWGAN plays an adversarial game to hallucinate embeddings that can fool the discriminator, while the discriminator is trying to classify the fake embeddings from the real ones. Once the halluciantor is trained, we condition it on labels to generate diverse embeddings at each fine-tuning step. This effectively extends the fine-tuning dataset with diverse embedding-label pairs which carry intra-class variation that can be a useful learning signal for the language learner.

We evaluate our method, called Embedding Hallucination (Embedhalluc), on 15 tasks and show that it generally improves over recent fine-tuning

[*] Both authors contributed equally to this research.

Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pages 5522 - 5530
July 10-15, 2022 ©2022 Association for Computational Linguistics

methods. We further experimentally show the overall superiority of EmbedHalluc when comparing to regularization methods proposed to address the problem of over-fitting during fine-tuning of LMs, such as Mixout (Lee et al., 2019) and Re-Init (Zhang et al., 2021). Finally, since our method is a form of data augmentation, we also compare EmbedHalluc to a common data augmentation technique EDA, and semi-supervised learning where unlabeled data is already available.

2 Related Work

Fine-tuning of Language Models. Better fine-tuning of language models can be achieved by proper initialization (Dodge et al., 2020), regularization (Lee et al., 2019) or prompts (Schick and Schütze, 2021). Other tricks include bias correction in optimizer and re-initialization of top layers in Transformer (Zhang et al., 2021). Instead of fine-tuning all parameters in a model, other work explore only learning a few vectors (Lester et al., 2021; Li and Liang, 2021; Guo et al., 2021) or a few additional parameters (Houlsby et al., 2019).

Hallucination Methods. Feature Hallucination of examples is first introduced for visual recognition (Hariharan and Girshick, 2017) by meta-learning (Wang et al., 2018), variational inference (Luo et al., 2021; Lazarou et al., 2022), and adversarial learning (Li et al., 2020; Tjio et al., 2022). Label Hallucination (Jian and Torresani, 2022) assigns soft pseudo-labels for unlabelled images to extend the fine-tuning few-shot dataset.

Learning from limited labeled data (few-shot learning) in Computer Vision is usually achieved by meta-learning (Ren et al., 2018a,b; Jian et al., 2020; Jian and Gao, 2021) or transfer learning (Tian et al., 2020). In NLP, few-shot learning has been successfully applied to machine translation (Arthaud et al., 2021), abstract summarizing (Fabbri et al., 2021), question and answering (Hua et al., 2020; Ram et al., 2021), and entity recognition (de Lichy et al., 2021; Tong et al., 2021; Ding et al., 2021), by meta learning (Li and Zhang, 2021; Bansal et al., 2020; Sharaf et al., 2020), data augmentation (Wei et al., 2021; Wei and Zou, 2019; Karimi et al., 2021; Jian et al., 2022), and prompts (Gao et al., 2021; Tam et al., 2021).

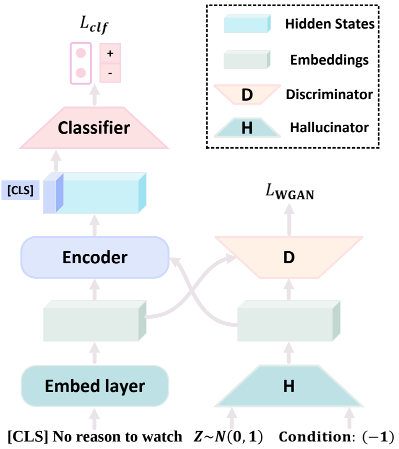

Figure 1: Overview of our method: The encoder model takes real embeddings of sentences and hallucinated embeddings as input and learns from their mixture. Hallucinator H generates fake embeddings conditioned on the class label. The discriminator D discriminates real embeddings and fake embeddings leading to the GAN loss $\mathcal{L}_{WGAN}$.

Our method is a generative data augmentation method in the embedding space. Different from (Wei et al., 2021) which uses EDA (Wei and Zou, 2019) to augment examples at the discrete input space, we hallucinate auxiliary examples at the embedding space. Our method shares similarity to FDA (Kumar et al., 2019), which is also a generative data augmentation method, but at the feature space. Also, different from FDA which is focused on two intent classification tasks, our method can be applied to a wide-range of NLP task as shown by our experiments on 15 diverse tasks.

3 Method

3.1 Conditional Wasserstein GAN

GAN (Goodfellow et al., 2014) has led the revolution of generative models to achieve impressive results in synthesizing images (Zhu et al., 2017) and higher dimensional data (Wang et al., 2020). Wasserstein GAN (WGAN) (Arjovsky et al., 2017) uses the Wasserstein distance as the objective function to stabilize the training of GAN.

Our hallucinator is trained under the conditional WGAN framework. After the training, we use it to generate pseudo-embeddings of examples by feeding it with random noisy vectors z sampled from $\mathcal{N}(0, 1)$ and the corresponding condition class labels c_i. The hallucinated embeddings s_{halluc}, in principal, are indiscriminative to the embeddings of observed examples in that class.

3.2 Fine-tuning with Hallucinated Embedding

For a single input sentence, we first pass it through the embedding layer to get the sentence embedding s_{sent}. We then concatenate s_{sent} with $s_{\text{halluc}}(c_i)$ to form a batch of mixture of real and fake embeddings $[s_{\text{sent}}, s_{\text{halluc}}(c_i)]$. The encoder learns from the batch with the corresponding labels $[c_{\text{sent}}, c_i]$.

Label Calibration. The hallucinated embedding $s_{\text{halluc}}(c_i)$ is conditioned on its label c_i. However, this hard label may not best represent the class information of the hallucinated embedding. We propose Label Calibration (LabelCalib) by pseudo-labeling from a teacher model $\mathcal{F}_{\text{GEN0}}$ (LM_1 in Algorithm 1), where $\mathcal{F}_{\text{GEN0}}$ is first fine-tuned on the original training set (without augmentation). The soft-label of the embedding $s_{\text{halluc}}(c_i)$ is then $c_{\text{pseudo},i} = \mathcal{F}_{\text{GEN0}}(s_{\text{halluc}}(c_i))$. Finally, the language model $\mathcal{M}$ learns from the hallucinated embedding by KL-divergence

$$\mathcal{L}_{\text{halluc}} = \text{KL}(\mathcal{M}(s_{\text{halluc}}(c_i)), c_{\text{pseudo},i}) \quad (1)$$

The total loss of our method is

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{real}} + \mathcal{L}_{\text{halluc}} \quad (2)$$

where $\mathcal{L}_{\text{real}}$ is the loss learning from real embedding-label pairs. The pseudo-code for fine-tuning of few-shot language learners with hallucinated embeddings is shown in Algorithm 1.

Note that baselines considered in this paper use total loss $\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{real}}$. Computing $\mathcal{L}_{\text{halluc}}$ requires one additional forward pass of the hallucinator and one more forward pass and backward pass of the language model. Thus, our method has about ×2 computational overhead compared to the baselines.

4 Experiments

4.1 Evaluation Datasets and Protocol

We evaluate our method on 15 classification tasks. The evaluations are conducted by averaging results on 5 different train test splits. We sample 16 examples per class to form a training set and construct a validation set with the same size as the training set.

4.2 Training Details for Embedding Hallucinators

The training of Embedding Hallucinators involves training a generator and discriminator in the cWGAN framework. The generator is a 4-blocks model, with each block containing a FullyConnect layer followed by a BatchNorm and LeakyReLU. The hidden dimensions of the generator are $128, 256, 512, 1024$. The hallucinated embeddings, i.e., outputs of the generator are tensors of $L \times 1024$, where the length of the generated embeddings L is set to be 128. The discriminator is a 3-blocks model, each bock having a sequence of FullyConnect-BatchNorm-LeakyReLU with the same hidden dimension of 512.

Algorithm 1 Our method: EmbedHalluc

```
$Max\_Step = 1000$,
$LM$: Language model,
$H$: Emebedding hallucinator (pre-trained),
$Train\_Set$: Training set,
$Sample$: Randomly sampling function,
$CE$: Cross Entropy loss,
$KL$: KL-divergence loss.
for i in $Max\_Step$ do ▷ Training LM$_1$
    $sent, y = Sample(Train\_Set)$
    $output_1 = LM_1(sent)$
    $L = CE(output_1, y)$
    $L.backward()$
    $optimizer.step()$
end for
for i in $Max\_Step$ do ▷ Training LM$_2$
    $sent, y = Sample(Train\_Set)$
    $embed = H(\mathcal{N}(0,1)), c)$
        ▷ Learning from real text
    $output_1 = LM_2(sent)$
    $L_{real} = CE(output_1, y)$
    $L_{real}.backward()$
    $optimizer.step()$
        ▷ Learning from hallucination
    $prob_2 = LM_1(embed)$
    $output_2 = LM_2(embed)$
    $L_{halluc} = KL(prob_2, output_2)$
    $L_{halluc}.backward()$
    $optimizer.step()$
end for
return $LM_2$
```

We train the Embedding Hallucinators for 150 epochs using a batch size of 64, the Adam optimizer ($\beta = (0.5, 0.999)$), and a learning rate of 0.0002. The real embeddings are collected from the language few-shot training set by passing text into the embedding layer of the language model. We apply gradient penalty with weight of loss 100 for training the cWGAN.

4.3 Training Details for Few-Shot Language Learners

We draw two mini-batches during the training of our few-shot language learners, i.e., one from the real language few-shot training set, another one by sampling the hallucinators (see Algorithm 1).

To fairly compare our method with baselines and other methods, when learning with real sentences, we use the same learning rate of $1e^{-5}$ (further justification of using this learning rate can be found in Appendix D). Our method learns from hallucinated embeddings with a grid search of learning rate of $1e^{-5}, 5e^{-6}, 1e^{-6}$, and batch size of $4, 6, 8$. We use the same search for EDA (Wei and Zou, 2019) and semi-supervised pseduo-labeling (SSL) when learning with additional augmented or pseudo-labeled data.

The models are selected based on the validation accuracy every 100 steps. Finally, results are reported by testing the models on the testing dataset. The algorithm is implemented in PyTorch-1.10 and experiments are conducted on Nvidia RTX-6000 and RTX-A6000 GPU.

4.4 Main Results on 15 Tasks

We compare our method EmbedHalluc (w/o or w/ LabelCalib) using RoBERTa-large on 15 tasks with two fine-tuning methods: conventional (Table 1) and prompt-based fine-tuning (Table 2). Results for BERT-large-cased can be found in Appendix B.

Task	Fine-tuning	EmbedHalluc	w/LabelCalib
SST-2 (acc)	76.8 (4.2)	**82.6** (5.6)	82.0 (4.7)
Subj (acc)	90.3 (1.5)	**91.3** (0.8)	**91.3** (0.9)
SST-5 (acc)	40.6 (2.2)	40.3 (1.5)	**41.6** (2.6)
CoLA (Matt.)	36.0 (9.9)	**39.7** (10.8)	38.1 (11.8)
TREC (acc)	83.0 (4.9)	**88.1** (2.5)	87.9 (1.0)
MNLI (acc)	41.6 (5.2)	48.0 (9.5)	**49.6** (5.8)
MNLI-mm (acc)	42.7 (5.9)	49.7 (10.5)	**51.8** (6.1)
SNLI (acc)	52.9 (6.7)	**54.4** (3.4)	52.3 (5.3)
QNLI (acc)	55.3 (2.7)	60.2 (5.3)	**64.9** (5.1)
QQP (acc)	59.2 (8.6)	64.6 (5.0)	**66.7** (5.3)
RTE (acc)	52.9 (1.4)	53.4 (1.7)	**55.9** (4.3)
MRPC (F1)	76.3 (5.2)	**78.7** (1.9)	78.1 (3.0)
MR (acc)	74.5 (5.9)	79.4 (5.5)	**80.8** (3.2)
MPQA (acc)	65.0 (1.5)	70.1 (7.0)	**70.5** (4.6)
CR (acc)	71.7 (7.5)	75.1 (5.6)	**78.0** (3.8)

Table 1: Comparison of conventional fine-tuning and our EmbedHalluc, using RoBERTa-large. Our Label Calibration (LabelCalib) can further improve the results.

In conventional fine-tuning, EmbedHalluc improves over the baseline in 14 tasks, only marginally under-performs in SST-5 (40.3 vs. 40.6 of baseline). When combining with LabelCalib, our method outperforms in all tasks. When applying to prompt-based fine-tuning, while our method under-performs in MNLI, MNLI-mm and RTE, it outperforms for all other tasks, with substantial improvements over the baseline in CoLA, TREC, QNLI, MRPC.

The relatively smaller improvements for prompt-based methods may be due to the inconsistency and randomness in the learning process since we have to insert `[mask]` token to a random position in the hallucinated embedding s_{halluc}, for the calculation of the loss. Whereas, in conventional fine-tuning, the `[CLS]` token is always appended to the beginning of s_{halluc} and the classification is performed at the `[CLS]` token.

Task	Prompt-based	EmbedHalluc	w/LabelCalib
SST-2 (acc).	92.7 (0.4)	92.8 (0.7)	**93.1** (0.7)
Subj (acc)	91.3 (1.0)	**92.0** (0.4)	91.7 (1.3)
SST-5 (acc)	48.8 (1.0)	49.0 (2.2)	**49.4** (1.4)
CoLA (Matt.)	7.3 (5.8)	12.3 (7.6)	**22.1** (15.6)
TREC (acc)	83.8 (5.3)	85.5 (3.3)	**87.1** (2.9)
MNLI (acc)	**69.7** (2.0)	68.0 (2.8)	68.5 (1.7)
MNLI-mm (acc)	**71.5** (1.9)	69.9 (3.0)	70.6 (1.7)
SNLI (acc)	78.0 (3.0)	**78.8** (2.3)	78.4 (2.3)
QNLI (acc)	68.6 (2.8)	69.6 (0.3)	**71.6** (2.0)
QQP (acc)	70.2 (4.3)	71.9 (5.2)	**74.2** (0.9)
RTE (acc)	**70.9** (3.3)	69.9 (3.3)	66.9 (3.4)
MRPC (F1)	74.6 (6.8)	78.0 (4.9)	**80.3** (3.5)
MR (acc)	86.8 (0.9)	87.2 (0.9)	**87.5** (0.9)
MPQA (acc)	85.4 (1.8)	84.2 (1.9)	**85.4** (1.9)
CR (acc)	91.1 (1.0)	91.1 (0.9)	**91.3** (0.3)

Table 2: Comparison of prompt-based fine-tuning and our EmbedHalluc, using RoBERTa-large.

4.5 Comparing to EDA and SSL

Since our method is a generative data augmentation (DA) method, we compare it to another DA method EDA. We also consider semi-supervised learning (SSL) which relies on unlabeled data (64 examples per class in our experiments). We apply pseudo-labeling (Cascante-Bonilla et al., 2021) for SSL, *i.e.*, we first fine-tune the model with the few-shot training set and use the fine-tuned model to pseudo-label the unlabeled data, finally we fine-tune the model again with the few-shot training set combined with the pseudo-labeled set.

EDA edits the input sentences by applying synonym replacement, random swap, random deletion and random insertion for a default 10% (α) of tokens. EDA either greatly change the sentence with a large α or fails to introduce substantial variations (which is crucial in the extreme low data setting) of inputs with a small α. Since it operates in the continuous embedding space, EmbedHalluc hallucinates diverse embeddings that follow the distribution of few-shot set. Thus, we observe in Table 3 that EmbedHalluc is overall superior to EDA.

EmbedHalluc is still competitive when comparing against SSL which assumes to have additional 64 examples per class from the task distribution.

4.6 Negative Results from Regularizations

Our method can also be viewed as an implicit regularization method. Thus, we also compare to two

Task	fine-tuning	EmbedHalluc	EDA	SSL
SST-2	76.8 (4.2)	82.6 (5.6)	82.3 (6.2)	**83.2** (6.0)
Subj	90.3 (1.5)	**91.3** (0.8)	89.2 (2.0)	91.2 (1.0)
SST-5	40.6 (2.2)	40.3 (1.5)	38.8 (3.7)	**41.7** (1.9)
CoLA	36.0 (9.9)	**39.7** (10.8)	25.5 (11.0)	39.6 (11.8)
TREC	83.0 (4.9)	**88.1** (2.5)	84.0 (1.9)	87.4 (3.4)
MNLI	41.6 (5.2)	**48.0** (9.5)	42.0 (3.9)	43.9 (4.2)
MNLI-mm	42.7 (5.9)	**49.7** (10.5)	44.2 (3.4)	45.6 (4.7)
SNLI	52.9 (6.7)	54.4 (3.4)	48.0 (4.7)	**54.9** (7.7)
QNLI	55.3 (2.7)	**60.2** (5.3)	58.7 (5.3)	53.6 (1.8)
QQP	59.2 (8.6)	**64.6** (5.0)	60.7 (6.8)	63.2 (7.1)
RTE	52.9 (1.4)	53.4 (1.7)	53.0 (4.9)	**53.9** (1.2)
MRPC	76.3 (5.2)	**78.7** (1.9)	73.8 (7.5)	77.3 (5.4)
MR	74.5 (5.9)	**79.4** (5.5)	78.1 (2.5)	77.9 (4.9)
MPQA	65.0 (1.5)	70.1 (7.0)	**72.8** (7.8)	68.8 (3.5)
CR	71.7 (7.5)	75.1 (5.6)	**80.7** (5.2)	75.6 (8.6)

Table 3: Comparison of EmbedHalluc, EDA, and SSL by pseudo-labeling, using RoBERTa-large as the base model and conventional fine-tuning as the base learning method.

Task	fine-tuning	EmbedHalluc	Re-init	Mixout
SST-2	76.8 (4.2)	**82.6** (5.6)	82.5 (1.9)	78.5 (9.4)
Subj	90.3 (1.5)	**91.3** (0.8)	91.1 (2.4)	90.3 (0.8)
SST-5	40.6 (2.2)	40.3 (1.5)	**41.2** (1.9)	37.5 (3.0)
CoLA	36.0 (9.9)	**39.7** (10.8)	33.4 (8.1)	38.6 (5.9)
TREC	83.0 (4.9)	**88.1** (2.5)	81.8 (5.6)	86.0 (3.4)
MNLI	41.6 (5.2)	**48.0** (9.5)	43.7 (4.3)	42.7 (4.6)
-mm	42.7 (5.9)	**49.7** (10.5)	45.2 (4.8)	45.0 (5.4)
SNLI	52.9 (6.7)	**54.4** (3.4)	52.1 (2.2)	53.7 (3.8)
QNLI	55.3 (2.7)	**60.2** (5.3)	59.8 (5.0)	57.1 (3.5)
QQP	59.2 (8.6)	**64.6** (5.0)	60.2 (10.6)	62.4 (6.0)
RTE	52.9 (1.4)	53.4 (1.7)	52.5 (5.4)	**53.5** (2.5)
MRPC	76.3 (5.2)	**78.7** (1.9)	67.0 (20.1)	77.4 (2.7)
MR	74.5 (5.9)	**79.4** (5.5)	71.3 (9.0)	67.6 (10.0)
MPQA	65.0 (1.5)	**70.1** (7.0)	68.8 (7.5)	68.0 (5.6)
CR	71.7 (7.5)	75.1 (5.6)	**83.0** (2.2)	67.5 (5.4)

Table 4: Comparisons of EmbedHalluc to Re-init and Mixout, using RoBERTa-large as base models and conventional fine-tuning as the base learning method.

latest methods for better fine-tuning language models with regularization. Zhang et al. (2021) find that fine-tuning can be achieved by: correcting bias in the optimizer, re-initialization of top layers, and training longer. Correcting bias in the optimizer is already fixed by the default optimizer in Huggingface Transformer and training longer surely will lead to further over-fitting in our extreme data scarce scenario. Thus, we consider re-initialization (Re-Init) of top layers as one of our comparisons. We further compare against Mixout (Lee et al., 2019), which is shown to be an effective regularization when fine-tuning with a few thousand examples. We used the public code for both of these methods. Since we adapt their code to our extreme data deficient setting, we re-search the hyper-parameters of both methods (including their suggested values). For Re-Init, we search the top 1,2,3,4,5 layers; and for Mixout, we search mixout rate from $0.1, 0.2, ..., 0.9$ and report their best results in Table 4, using RoBERTa-large. Results for BERT-large-cased can be found in Appendix C.

We find that those two methods fail to alleviate the over-fitting problem in such extreme setting, though they have been to be effective when given a few thousands examples.

5 Comparing to Adversarial Training

Adversarial training adds noise into the training data to increase the robustness of a model. It has been shown that adversarial training can also improve the performance of language models. Here, we compare EmbedHalluc to two recent adversarial training methods, freeLB (Zhu et al., 2020) and SMART (Jiang et al., 2020) adapted to our setting. For freeLB, we use the publicly available code and suggested hyper-parameters for each task. In addition to the default batch size and learning rate used in the baseline fine-tuning and EmbedHalluc, we also search additional batch sizes and learning rates for freeLB. We use the default setting for SMART. As shown in Table 5, with one exception, our method largely outperforms freeLB and SMART.

Task	EmbedHalluc	freeLB	SMART
SST-2	82.6 (5.6)	78.5 (8.8)	**83.6** (3.1)
CoLA	**39.7** (10.8)	31.6 (11.1)	34.2 (4.1)
MNLI	**48.0** (9.5)	40.8 (3.5)	39.1 (4.1)
-mm	**49.7** (10.5)	42.1 (4.3)	40.0 (4.9)
QNLI	**60.2** (5.3)	58.2 (5.0)	55.5 (2.9)
QQP	**64.6** (5.0)	62.4 (3.8)	56.5 (6.1)
RTE	**53.4** (1.7)	52.3 (4.2)	49.2 (2.6)
MRPC	**78.7** (1.9)	76.8 (3.5)	77.1 (3.1)

Table 5: Comparisons of EmbedHalluc to freeLB and SMART, using RoBERTa-large as base models and conventional fine-tuning as the base learning method.

6 Limitations

While EmbedHalluc works well empirically, it relies on hallucinating non-interpretable embeddings to facilitate the learning process. Besides, the learning of cWGAN requires careful human attention to maintain a stable training.

7 Conclusion

In this paper, we introduce an embedding hallucination method for data augmentation for few-shot learning, based on cWGAN. The proposed method improves over the baselines in 15 tasks and outperforms a common augmentation method, and two recent regularization methods.

8 Ethics Statement

As far as we are aware, our proposed work does not have any explicit ethical concerns. However, our work relies on pre-trained language models, which have been shown to be biased in prior work (Liang et al., 2021). As such, users of such models, specially for sensitive applications, should be aware of and if possible address such issues.

References

Martin Arjovsky, Soumith Chintala, and Léon Bottou. 2017. Wasserstein generative adversarial networks. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 214–223. PMLR.

Farid Arthaud, Rachel Bawden, and Alexandra Birch. 2021. Few-shot learning through contextual data augmentation. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 1049–1062, Online. Association for Computational Linguistics.

Trapit Bansal, Rishikesh Jha, Tsendsuren Munkhdalai, and Andrew McCallum. 2020. Self-supervised meta-learning for few-shot natural language classification tasks. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 522–534.

Samyadeep Basu, Karine lp Kiun Chong, Amr Sharaf, Alex Fischer, Vishal Rohra, Michael Amoake, Hazem El-Hammamy, Ehi Nosakhare, Vijay Ramani, and Benjamin Han. 2021. Semi-supervised few-shot intent classification and slot filling. *CoRR*, abs/2109.08754.

Paola Cascante-Bonilla, Fuwen Tan, Yanjun Qi, and Vicente Ordonez. 2021. Curriculum labeling: Revisiting pseudo-labeling for semi-supervised learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 6912–6920.

Cyprien de Lichy, Alexa AI Amazon, Hadrien Glaude, and William Campbell. 2021. Meta-learning for few-shot named entity recognition. *MetaNLP 2021*, page 44.

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, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics.

Ning Ding, Guangwei Xu, Yulin Chen, Xiaobin Wang, Xu Han, Pengjun Xie, Hai-Tao Zheng, and Zhiyuan Liu. 2021. Few-nerd: A few-shot named entity recognition dataset. In *ACL-IJCNLP*.

Jesse Dodge, Gabriel Ilharco, Roy Schwartz, Ali Farhadi, Hannaneh Hajishirzi, and Noah A. Smith. 2020. Fine-tuning pretrained language models: Weight initializations, data orders, and early stopping. *ArXiv*, abs/2002.06305.

Alexander Fabbri, Simeng Han, Haoyuan Li, Haoran Li, Marjan Ghazvininejad, Shafiq Joty, Dragomir Radev, and Yashar Mehdad. 2021. Improving zero and few-shot abstractive summarization with intermediate fine-tuning and data augmentation. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 704–717, Online. Association for Computational Linguistics.

Tianyu Gao, Adam Fisch, and Danqi Chen. 2021. Making pre-trained language models better few-shot learners. In *Association for Computational Linguistics (ACL)*.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative adversarial nets. In *Advances in Neural Information Processing Systems*, volume 27. Curran Associates, Inc.

Demi Guo, Alexander Rush, and Yoon Kim. 2021. Parameter-efficient transfer learning with diff pruning. 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 4884–4896, Online. Association for Computational Linguistics.

Bharath Hariharan and Ross Girshick. 2017. Low-shot visual recognition by shrinking and hallucinating features. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3018–3027.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-efficient transfer learning for nlp. In *International Conference on Machine Learning*, pages 2790–2799. PMLR.

Yuncheng Hua, Yuan-Fang Li, Gholamreza Haffari, Guilin Qi, and Tongtong Wu. 2020. Few-shot complex knowledge base question answering via meta reinforcement learning. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5827–5837.

Yiren Jian, Karim Ahmed, and Lorenzo Torresani. 2020. Task meta-transfer from limited parallel labels. *Meta-Learning workshop, NeurIPS 2020*.

Yiren Jian and Chongyang Gao. 2021. Metapix: Domain transfer for semantic segmentation by meta pixel weighting. *Image and Vision Computing*, 116:104334.

Yiren Jian, Chongyang Gao, and Soroush Vosoughi. 2022. Contrastive learning for prompt-based few-shot language learners. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics.

Yiren Jian and Lorenzo Torresani. 2022. Label hallucination for few-shot classification. In *Proceedings of the AAAI Conference on Artificial Intelligence*.

Haoming Jiang, Pengcheng He, Weizhu Chen, Xiaodong Liu, Jianfeng Gao, and Tuo Zhao. 2020. SMART: Robust and efficient fine-tuning for pre-trained natural language models through principled regularized optimization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2177–2190, Online. Association for Computational Linguistics.

Akbar Karimi, Leonardo Rossi, and Andrea Prati. 2021. AEDA: an easier data augmentation technique for text classification. In *Findings of the Association for Computational Linguistics: EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 16-20 November, 2021*, pages 2748–2754. Association for Computational Linguistics.

Varun Kumar, Hadrien Glaude, Cyprien de Lichy, and Wlliam Campbell. 2019. A closer look at feature space data augmentation for few-shot intent classification. In *Proceedings of the 2nd Workshop on Deep Learning Approaches for Low-Resource NLP (DeepLo 2019)*, pages 1–10, Hong Kong, China. Association for Computational Linguistics.

Michalis Lazarou, Tania Stathaki, and Yannis Avrithis. 2022. Tensor feature hallucination for few-shot learning. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 3500–3510.

Cheolhyoung Lee, Kyunghyun Cho, and Wanmo Kang. 2019. Mixout: Effective regularization to finetune large-scale pretrained language models. In *International Conference on Learning Representations*.

Brian Lester, Rami Al-Rfou, and Noah Constant. 2021. The power of scale for parameter-efficient prompt tuning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 3045–3059, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Judith Yue Li and Jiong Zhang. 2021. Semi-supervised meta-learning for cross-domain few-shot intent classification. *MetaNLP 2021*, page 67.

Kai Li, Yulun Zhang, Kunpeng Li, and Yun Fu. 2020. Adversarial feature hallucination networks for few-shot learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13470–13479.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. 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 4582–4597, Online. Association for Computational Linguistics.

Paul Pu Liang, Chiyu Wu, Louis-Philippe Morency, and Ruslan Salakhutdinov. 2021. Towards understanding and mitigating social biases in language models. In *International Conference on Machine Learning*, pages 6565–6576. PMLR.

Qinxuan Luo, Lingfeng Wang, Jingguo Lv, Shiming Xiang, and Chunhong Pan. 2021. Few-shot learning via feature hallucination with variational inference. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 3963–3972.

Ori Ram, Yuval Kirstain, Jonathan Berant, Amir Globerson, and Omer Levy. 2021. Few-shot question answering by pretraining span selection. *arXiv e-prints*, pages arXiv–2101.

Mengye Ren, Eleni Triantafillou, Sachin Ravi, Jake Snell, Kevin Swersky, Joshua B Tenenbaum, Hugo Larochelle, and Richard S Zemel. 2018a. Meta-learning for semi-supervised few-shot classification. In *International Conference on Learning Representations*.

Mengye Ren, Wenyuan Zeng, Bin Yang, and Raquel Urtasun. 2018b. Learning to reweight examples for robust deep learning. In *ICML*.

Timo Schick and Hinrich Schütze. 2021. Exploiting cloze-questions for few-shot text classification and natural language inference. In *EACL*.

Amr Sharaf, Hany Hassan, and Hal Daumé III. 2020. Meta-learning for few-shot nmt adaptation. *ACL*, page 43.

Derek Tam, Rakesh R Menon, Mohit Bansal, Shashank Srivastava, and Colin Raffel. 2021. Improving and simplifying pattern exploiting training. In *Empirical Methods in Natural Language Processing (EMNLP)*.

Yonglong Tian, Yue Wang, Dilip Krishnan, Joshua B Tenenbaum, and Phillip Isola. 2020. Rethinking few-shot image classification: a good embedding is all you need? In *European Conference on Computer Vision*, pages 266–282. Springer.

Gabriel Tjio, Ping Liu, Joey Tianyi Zhou, and Rick Siow Mong Goh. 2022. Adversarial semantic hallucination for domain generalized semantic segmentation.

In *IEEE Winter Conf. on Applications of Computer Vision*.

Meihan Tong, Shuai Wang, Bin Xu, Yixin Cao, Minghui Liu, Lei Hou, and Juanzi Li. 2021. Learning from miscellaneous other-class words for few-shot named entity recognition. 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 6236–6247, Online. Association for Computational Linguistics.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *International Conference on Learning Representations*.

Jiancong Wang, Yuhua Chen, Yifan Wu, Jianbo Shi, and James Gee. 2020. Enhanced generative adversarial network for 3d brain mri super-resolution. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 3627–3636.

Yu-Xiong Wang, Ross Girshick, Martial Hebert, and Bharath Hariharan. 2018. Low-shot learning from imaginary data. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7278–7286.

Jason Wei, Chengyu Huang, Soroush Vosoughi, Yu Cheng, and Shiqi Xu. 2021. Few-shot text classification with triplet networks, data augmentation, and curriculum learning. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5493–5500, Online. Association for Computational Linguistics.

Jason Wei and Kai Zou. 2019. EDA: Easy data augmentation techniques for boosting performance on text classification tasks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 6383–6389, Hong Kong, China. Association for Computational Linguistics.

Tianyi Zhang, Felix Wu, Arzoo Katiyar, Kilian Q Weinberger, and Yoav Artzi. 2021. Revisiting few-sample {bert} fine-tuning. In *International Conference on Learning Representations*.

Chen Zhu, Yu Cheng, Zhe Gan, Siqi Sun, Tom Goldstein, and Jingjing Liu. 2020. Freelb: Enhanced adversarial training for natural language understanding. In *International Conference on Learning Representations*.

Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. 2017. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Computer Vision (ICCV), 2017 IEEE International Conference on*.

A Best Learning Rate for RoBERTa-prompt

Here, we provide best learning rates (LR, searched from $1e^{-5}, 5e^{-6}, 1e^{-6}$ as discussed in main paper) for $\mathcal{L}_{\text{halluc}}$ of EmbedHalluc for each task used in RoBERTa-large prompt-based fine-tuning.

Task	LR
SST-2	$1e^{-6}$
Subj	$1e^{-5}$
SST-5	$1e^{-6}$
CoLA	$1e^{-5}$
TREC	$1e^{-6}$
MNLI	$1e^{-5}$
MNLI-mm	$1e^{-5}$
SNLI	$1e^{-6}$
QNLI	$5e^{-6}$
QQP	$1e^{-6}$
RTE	$1e^{-6}$
MRPC	$1e^{-6}$
MR	$5e^{-6}$
MPQA	$5e^{-6}$
CR	$5e^{-6}$

Table A.1: Best learning rate for $\mathcal{L}_{\text{halluc}}$ found for RoBERTa-large prompt-based fine-tuning.

B EmbedHalluc with BERT

In addition to the experiments using RoBERTa shown in the main paper, here we show the results of BERT-large-cased with conventional fine-tuning as a further check on robustness of our method with respect to the choice of model. Table B.1 shows the results of the experiments. EmbedHalluc outperforms the baseline across 14 of the 15 tasks with an average improvement of 2.43 over the baseline.

Task	fine-tuning	EmbedHalluc
SST-2 (acc)	73.9 (5.4)	**76.6** (3.8)
Subj (acc)	85.2 (3.4)	**89.0** (0.9)
SST-5 (acc)	37.6 (4.5)	**38.9** (4.2)
CoLA (Matt.)	21.9 (10.0)	**28.7** (7.2)
TREC (acc)	77.6 (6.3)	**81.8** (3.1)
MNLI (acc)	35.5 (0.9)	**36.1** (0.8)
MNLI-mm (acc)	36.0 (0.9)	**36.7** (1.5)
SNLI (acc)	39.7 (3.6)	**41.0** (3.2)
QNLI (acc)	53.9 (2.5)	**55.1** (3.0)
QQP (acc)	56.7 (4.3)	**59.5** (2.9)
RTE (acc)	50.9 (3.5)	**54.2** (2.5)
MRPC (F1)	72.3 (6.5)	**76.7** (2.8)
MR (acc)	**73.2** (6.3)	66.4 (8.9)
MPQA (acc)	64.6 (5.6)	**65.3** (4.0)
CR (acc)	67.9 (9.0)	**77.5** (12.6)

Table B.1: Comparison of conventional fine-tuning and our EmbedHalluc, using BERT-large-cased.

C Regularization Methods with BERT

Besides the experiments with RoBERTa-large shown in the main paper, we present Re-Init and Mixout using BERT-large-cased in this section. The results are shown in Table C.1.

Qualitatively similar to what we observe with experiments using RoBERTa-large in the main paper, Re-Init and Mixout fail to outperform EmbedHalluc in most tasks, with the exceptions of SNLI and QNLI.

Task	fine-tuning	EmbedHalluc	Re-init	Mixout
SST-2	73.9 (5.4)	**76.6** (3.8)	73.6 (4.2)	71.9 (3.9)
Subj	85.2 (3.4)	**89.0** (0.9)	87.0 (2.5)	85.6 (1.1)
SST-5	37.6 (4.5)	**38.9** (4.2)	36.4 (2.5)	36.0 (3.1)
CoLA	21.9 (10.0)	**28.7** (7.2)	23.8 (11.8)	13.0 (11.2)
TREC	77.6 (6.3)	**81.8** (3.1)	79.7 (3.4)	78.8 (4.4)
MNLI	35.5 (0.9)	**36.1** (0.8)	35.1 (1.6)	33.2 (0.6)
MNLI-mm	36.0 (0.9)	**36.7** (1.5)	35.6 (2.2)	33.6 (1.0)
SNLI	39.7 (3.6)	41.0 (3.2)	**45.3** (3.3)	42.5 (3.2)
QNLI	53.9 (2.5)	55.1 (3.0)	**55.9** (2.0)	55.0 (2.7)
QQP	56.7 (4.3)	**59.5** (2.9)	58.9 (2.9)	56.6 (6.1)
RTE	50.9 (3.5)	**54.2** (2.5)	51.5 (3.8)	50.8 (1.8)
MRPC	72.3 (6.5)	**76.7** (2.8)	63.4 (4.6)	74.1 (3.2)
MR	**73.2** (6.3)	66.4 (8.9)	60.8 (4.4)	63.0 (4.5)
MPQA	64.6 (5.6)	**65.3** (4.0)	64.8 (5.8)	60.9 (2.9)
CR	67.9 (9.0)	**77.5** (12.6)	71.6 (8.3)	75.9 (6.1)

Table C.1: Comparisons of EmbedHalluc to Re-init and Mixout, using BERT-large as the base models and conventional fine-tuning as the base learning method.

D Learning Rate for Baselines

The baseline has only one loss $\mathcal{L}_{\text{real}}$, whereas we are learning with an additional loss $\mathcal{L}_{\text{halluc}}$, making the total loss to be $\mathcal{L}_{\text{real}} + \mathcal{L}_{\text{halluc}}$. The learning rate for $\mathcal{L}_{\text{real}}$ in the baselines and ours are kept the same. Note that we do not search for this learning rate for our method. We choose $1e^{-5}$, which is the most common learning rate to finetune BERT/RoBERTa. As we show in Table D.1, this learning rate produces reasonably good results for the baselines, being the best for 13 tasks and only marginally under-performing in the other 2 tasks. The results in Table D.1 are generated by running the baselines with a batch size of 2 and different learning rates $1e^{-5}$, $2e^{-5}$, $5e^{-5}$ suggested by Gao et al. (2021).

prompt	1e-5	2e-5	5e-5
SST-2	**92.7** (0.4)	91.0 (2.4)	83.6 (7.8)
subj	**91.3** (3.3)	87.8 (3.3)	83.9 (3.0)
SST-5	**48.8** (1.0)	48.1 (1.0)	43.6 (1.9)
CoLA	7.3 (5.8)	8.7 (5.7)	**10.5** (7.2)
trec	**83.8** (5.3)	79.3 (5.2)	78.0 (0.7)
MNLI	**69.7** (2.0)	66.5 (3.2)	60.0 (5.2)
MNLI-mm	**71.5** (1.9)	68.7 (3.2)	62.9 (5.3)
SNLI	**78.0** (3.0)	76.2 (2.3))	54.7 (10.3)
QNLI	**68.6** (2.8)	64.9 (3.1)	63.0 (7.2)
QQP	**70.2** (4.3)	64.2 (5.0)	57.7 (4.9)
RTE	**70.9** (3.3)	63.3 (4.8)	55.8 (8.4)
MRPC	**74.9** (6.8)	73.9 (3.8)	72.8 (2.6)
MR	**86.8** (0.9)	84.5 (1.8)	80.1 (3.9)
MPQA	85.4 (1.8)	**85.6** (1.1)	80.1 (3.9)
CR	**91.1** (1.0)	90.6 (0.9)	85.3 (2.6)

Table D.1: Results of baseline with Roberta-large prompt-based fine-tuning on different learning rate. $1e^{-5}$ is what we used in main experiments.