テキスト生成モデルを利用したデータセット蒸留

前川 在　小杉 哲　船越 孝太郎　奥村 学
東京工業大学
{maekawa, kosugi, funakoshi, oku}@lr.pi.titech.ac.jp

概要

データセット蒸留は，訓練データセット中の知識を蒸留することで，ニューラルモデルを効率的に学習可能な少量の合成データを獲得する．本研究では，学習効果の高い訓練データを生成するようテキスト生成モデルを学習することで，合成データをテキストとして獲得する手法を提案する．実験では，提案法を複数のテキスト分類タスクのデータセットに適用し，元のデータセットから選択された代表サンプル集合よりも高い性能でモデルを学習可能であることを示した．また，提案法によりテキストとして獲得した合成データは，蒸留時と異なる事前学習済みモデルの学習に対する汎化性能だけでなく，大規模言語モデルの in-context learning に対しても効果的であることを示した．

1　はじめに

深層学習モデルは，大規模なニューラルネットワークを大量の訓練データを用いて学習することにより，驚異的な性能を達成している．しかし，そのような大規模なモデルの学習には膨大な時間と計算資源を要するため，新しいモデルの開発はおろか，追学習さえ困難となっている．このような背景から，訓練データセット中の知識を蒸留することで，学習効果を保持しつつ訓練データセットを圧縮するデータセット蒸留 [1] が近年注目されている．データセット蒸留は，モデルを効率的に学習可能な少量の人工的なデータを作成することで，モデルの学習コストの大幅な削減を実現する．データセット蒸留は，画像分野を中心に様々な手法が提案されており [2, 3, 4, 5]，ニューラルアーキテクチャ探索 [6, 7]，連合学習 [8, 9]，継続学習 [10, 11]，データのプライバシー保護 [12, 13] といった応用の観点からも関心を集めている．

データセット蒸留では，勾配法を用いてデータを直接最適化することにより，学習効果の高い合成データを獲得する手法が一般的である．しかし，これらの手法はピクセル単位の連続値データとみなせる画像データには適用可能である一方で，離散データであるテキストには直接適用できない．そこで従来法では，離散的なテキストの代わりに連続値である単語埋め込みベクトルの系列として合成データを最適化する [14, 15, 16, 17]．しかし，この方法により単語埋め込みレベルで構築した合成データは，異なる単語埋め込みを持つモデルの学習には適用できないため，実応用の観点で致命的な問題となる．

そこで本研究では，合成データをテキストとして獲得可能なデータセット蒸留手法として，学習効果の高いテキストデータを生成するようテキスト生成モデルを学習する手法を提案する．つまり，離散性により最適化が困難なテキストデータの代わりに，それらを生成するモデルのパラメータを学習することで勾配法による最適化を可能にする．具体的には，テキスト生成モデルの生成サンプルで計算されたモデルの勾配と，訓練データセット中の実サンプルで計算された平均勾配が一致するようテキスト生成モデルを学習する (図 1)．

実験では，SST-2，QQP，MNLI-m の 3 つのテキスト分類タスクのデータセットに対して提案法を適用し，訓練データセットから代表的な実サンプルの集合を選択するコアセット選択の手法と比較して高い性能でモデルを学習可能な合成データセットを獲得できることを示した．さらに，提案法によりテキストとして獲得した合成データは，蒸留時と異なるモデルの学習に対しても汎化性能を示すだけでなく，大規模言語モデルの in-context learning に対しても有効であることが判明した．

2　提案法

2.1　訓練データ生成の事前学習

大規模なテキストデータで事前学習された Transformer ベースの言語モデル [18] の高いテキス

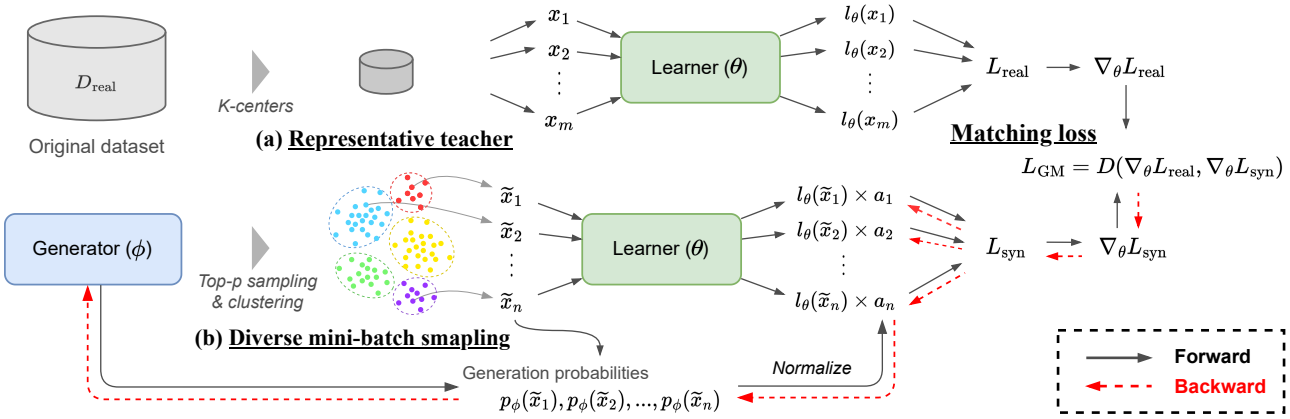

図 1　提案法の概略図．(a) は代表サンプルを用いた教師勾配，(b) は多様性を考慮したミニバッチ生成をそれぞれ示す．

ト生成能力に着目し，訓練データの生成モデルとして利用する．まず，蒸留の対象とする訓練データセットの実サンプルを用いて言語モデルを学習することで，実サンプルと同程度の学習効果を持つ訓練データを生成可能なテキスト生成モデルを獲得する．

訓練データの入力テキスト $x = w_1 \cdots w_{|x|}$ に対し，次式で計算される言語モデルの損失 l_ϕ を用いてテキスト生成モデルのパラメータ ϕ を最適化する．

$$l_\phi(x) = -\log p_\phi(x), \tag{1}$$

$$\text{where} \quad p_\phi(x) = \left\{ \prod_{w_t \in x} p_\phi(w_t | w_{<t}) \right\}^{\frac{1}{|x|}}. \tag{2}$$

その際，各訓練サンプルの入力テキストの先頭と末尾に，クラスラベルに対応する開始トークン <bos_i> と終了トークン <eos> をそれぞれ付与して学習することで，特定のクラスラベルに対応するサンプルを生成するよう制御する:

<bos_i> クラス i のテキスト <eos>.

また，2 文間の関係を分類するタスクについては，分割トークン <sep> を用いて結合し，2 文続けて生成するよう学習する:

<bos_i> テキスト 1 <sep> テキスト 2 <eos>.

以上の方法で学習したテキスト生成モデルを，次の 2.2 節で述べる勾配の一致度に基づく追学習の初期値として利用する．

2.2　勾配の一致度に基づく追学習

実サンプルよりも学習効果の高い訓練データを生成するため，データセット蒸留の手法の一つとして用いられる勾配の一致度に基づく損失 [2, 19] を目的関数として，テキスト生成モデルを追学習する．実サンプルのミニバッチ $\{x_i\}_{i=1}^M$ および生成サンプルのミニバッチ $\{\tilde{x}_i\}_{i=1}^N$ に対し，それぞれに対する分類モデルのパラメータ θ の勾配の一致度に基づく損失 L_{GM} は次のように計算される:

$$L_{\text{GM}} = D\left(\nabla_\theta L_{\text{real}}, \nabla_\theta L_{\text{syn}}\right) \quad \text{where}$$
$$L_{\text{real}} = \frac{1}{M}\sum_{i=1}^{M} l_\theta(x_i),\ L_{\text{syn}} = \frac{1}{N}\sum_{i=1}^{N} l_\theta(\tilde{x}_i). \tag{3}$$

ここで，$l_\theta(\cdot)$ は分類モデルの損失関数，$D(\cdot,\cdot)$ は次式で計算されるコサイン類似度に基づく距離関数である:

$$D(A, B) = 1 - \frac{A \cdot B}{\|A\|\|B\|}. \tag{4}$$

先行研究 [2, 19] に従い，L_{GM} はクラスラベルごとに計算し，その平均を目的関数とする．また，初期値だけでなく，学習過程全体を通した分類モデルの勾配を考慮するために，分類モデルを初期化する outer-loop と，分類モデルの更新および L_{GM} に基づく生成モデルの更新を行う inner-loop の 2 重ループのアルゴリズムを利用する (詳細は付録 A を参照)．

各 inner-loop においてテキスト生成モデルを更新するために，L_{GM} の勾配をテキスト生成モデルのパラメータ ϕ まで逆伝播させる必要がある．しかし，順伝播で介する生成サンプル $\{\tilde{x}_i\}_i^N$ は離散的なテキストであるため微分不可能である．そこで提案法では，平岡ら [20] の手法に着想を得て，各生成サンプルの生成確率に基づく損失の重みづけを導入することで逆伝播計算を可能にする (図 1)．L_{syn} を計算する際に，式 3 で示したように単にすべての生成サンプルに対する損失を平均する代わりに，テキスト生成モデルの生成確率 $p_\phi(\tilde{x}_i)$ に基づく重み付き平均を計算する:

$$L_{\text{syn}} = \sum_{i=1}^{N} a_i l_\theta(\tilde{x}_i), \quad \text{where} \quad a_i = \frac{p_\phi(\tilde{x}_i)}{\sum_{j=1}^{N} p_\phi(\tilde{x}_j)}. \tag{5}$$

これにより，L_{GM} の勾配は生成確率に基づく損失の重み $\{a_i\}_{i=1}^{N}$ を介してテキスト生成モデルまで逆伝播可能となる．

また，提案法の性能向上および学習安定化のために次の 2 つの手法を導入する[1)]．

(a) 代表サンプルを用いた教師勾配　Liu ら [21] に着想を得て，コアセット選択の手法の一つである K-centers [22, 23] で選択された代表サンプル集合を実サンプルのミニバッチ $\{x_i\}_{i=1}^{M}$ として用いることで，性能向上および学習の高速化を図る (図 1a)．K-centers では，特定のクラスラベルを持つ訓練データセット中の全てのサンプルを，K-means を利用して M クラスタに分割し，各クラスタ中心の最近傍サンプルを代表サンプルとする．これにより選択された代表サンプル集合は，勾配が大きく支配的になりやすい決定境界上のサンプルを排除しつつ，訓練データ全体をカバーする多様なサンプルをミニバッチに含めることで，教師として適切な勾配を提供すると考えられる．多様性と頑健性を考慮し，学習開始時に異なる乱数シードを用いて 10 通りの代表サンプル集合を作成し，inner-loop の各ステップにおいて，そのうち一つを実サンプルのミニバッチとして利用する．

(b) 多様性を考慮したミニバッチ生成　テキスト生成モデルは，自身が生成したサンプルのミニバッチ $\{\tilde{x}_i\}_{i}^{N}$ のうち勾配の一致度を向上させるサンプルの生成確率を上げるよう学習されるため，各ミニバッチに含まれる生成サンプルの偏りが学習に大きく影響を与える．そこで，inner-loop の各ステップで N 個のサンプルを生成する代わりに，I_{int} ステップおきに $N \times I_{\mathrm{int}}$ 個のサンプルをまとめて生成し，K-means を利用して N クラスタに分割する．inner-loop の各ステップにおいて，各クラスタからランダムに 1 サンプルずつ取り出して生成サンプルのミニバッチ $\{\tilde{x}_i\}_{i}^{N}$ として利用する (図 1b)．これにより，生成サンプルのミニバッチに多様性を持たせ，サンプル生成時の偏りの影響を緩和する．

2.3　合成データセットの構築

前節で述べた手順で学習したテキスト生成モデルを用いて合成データセットを構築する．生成時の偏りの影響を緩和するために，各クラスラベルごとに所望の合成データセットのサイズの 100 倍のサンプルを，多様性を重視した top-p サンプリング ($p = 0.95$) を用いて生成し，それらの K-center サンプルを合成データとして利用する．これにより，生成時の偏りやサンプルの冗長性を緩和し，多様なサンプルを合成データセットに含めることを可能にする．

3　実験

3.1　実験設定

評価実験には，GLUE [24] から SST-2, QQP, MNLI-m の 3 つのテキスト分類タスクのデータセットを利用した．SST-2 と MNLI-m については accuracy を，QQP については accuracy と F1 の平均を性能評価に用いた．

先行研究に倣い，Random，K-centers [22, 23]，Herding [25] の 3 つのコアセット選択の手法と比較した．また，単語埋め込みレベルのデータセット蒸留の従来法である TDD [14] とも比較した[2)]．さらに，提案法の勾配の一致度に基づく追学習の効果を確認するために，提案法において追学習を行わなかった場合との比較を行なった．

提案法のテキスト生成モデルとして GPT-2[3)] を，追学習時の勾配の一致度を計算する分類モデルとして $\mathrm{BERT}_{\mathrm{BASE}}$[4)] を利用した．計算コストを抑えるために，$\mathrm{BERT}_{\mathrm{BASE}}$ の全パラメータの勾配を計算する代わりに，ランダム初期化される最終層のパラメータの勾配のみを利用した．その他の実験設定の詳細は付録 C に記載した．

3.2　実験結果

$\mathrm{BERT}_{\mathrm{BASE}}$ に対する性能比較　表 1 に提案法の学習時と同じ $\mathrm{BERT}_{\mathrm{BASE}}$ を分類モデルとした場合の異なるデータサイズ (DPC: data-per-class) の実験結果を示す．表のスコアは異なる乱数シードで 100 回学習を行った分類モデルの性能の平均と標準偏差である[5)]．まず，単に訓練データの生成を学習しただけの追学習前のモデルでは，コアセット選択の手法の性能を明らかに下回った．これは直感の通りではあるが，テキスト生成モデルの生成サンプルの質が元の訓練データセットの実サンプルと比較して低いこ

1)　付録 B に ablation study の実験結果を示す．

2)　TDD は単語埋め込みだけでなく，クラスラベルおよび学習率も最適化することに注意して欲しい．

3)　https://huggingface.co/gpt2

4)　https://huggingface.co/bert-base-uncased

5)　Herding と TDD 以外は，異なる乱数シードで生成または選択した 20 通りのデータセットでそれぞれ 5 回ずつ学習した．

表 1　蒸留時と同じ $\text{BERT}_{\text{BASE}}$ に対する性能の比較結果．ただし TDD については，学習過程全体を通した二次勾配の逆伝播計算を要する手法の特性から，GPU メモリコストの観点で DPC=10, 20 の実験は実施できなかった．

	SST-2 (2 classes, 67.3k)			**QQP** (2 classes, 364k)			**MNLI-m** (3 classes, 393k)		
DPC (data-per-class)	5	10	20	5	10	20	5	10	20
Random	58.1±5.2	64.3±7.4	70.3±6.8	51.5±5.6	56.0±4.8	59.1±3.8	35.6±2.1	37.7±2.6	40.1±3.2
K-centers	70.8±4.1	75.9±4.7	79.8±3.5	**60.7±3.8**	60.9±3.1	62.6±2.7	36.2±2.4	41.8±3.2	45.3±3.0
Herding	70.2±5.7	73.2±5.7	76.9±4.4	56.0±5.6	59.7±4.1	62.3±3.4	36.2±3.8	38.7±3.7	42.8±3.5
TDD (embed.)	89.6±0.4	-	-	81.5±0.2	-	-	75.6±0.2	-	-
TDD (text)	50.2±1.6	-	-	39.6±6.8	-	-	33.4±1.8	-	-
提案法 (追学習前)	65.2±6.8	71.7±6.8	77.6±4.1	56.7±4.4	59.3±3.8	62.5±3.3	36.3±2.7	40.5±2.9	43.6±3.1
提案法 (追学習後)	**72.5±5.9**	**76.3±4.6**	**80.3±2.8**	58.8±5.2	**62.2±3.3**	**64.4±2.6**	**39.7±2.7**	**44.8±3.1**	**48.7±2.6**
Full dataset		92.7			89.6			86.7	

とを示唆している．一方で，勾配の一致度に基づく追学習を適用することで性能は大幅に改善され，コアセット選択の手法をほぼ全ての設定で上回った．特に K-centers との性能差は，提案法が実サンプルよりも学習効果の高い訓練データを生成していることを示している．また，TDD については，最適化した合成データを単語埋め込み列として直接利用することで高い性能を発揮する一方で，最近傍の埋め込みを持つ単語列に置き換え，テキストとして利用した場合，ランダム予測と同程度まで性能劣化が生じた (表の赤のハイライト)．これは，異なる単語埋め込みを持つモデルに適用する際は一度テキストに変換する必要があるため，汎用性の観点で大きな問題となる．

異なるモデルへの汎化性能　提案法はテキストとして合成データセットを構築するため，任意の単語埋め込みを持つモデルの学習に適用可能である．そこで，提案法の学習時に利用した $\text{BERT}_{\text{BASE}}$ と異なる，$\text{RoBERTa}_{\text{BASE}}$，$\text{BERT}_{\text{LARGE}}$，$\text{XLNet}_{\text{BASE}}$ の 3 つのモデルの学習に対する汎化性能を評価した．実験結果を表 2 に示す．実験結果から，提案法で獲得された合成データセットは，異なるモデルに対しても高い汎化性能を持つことが判明した．特に，$\text{BERT}_{\text{BASE}}$ と同じモデル構造を持つ $\text{RoBERTa}_{\text{BASE}}$ や $\text{BERT}_{\text{LARGE}}$ だけでなく，自己回帰モデルである $\text{XLNet}_{\text{BASE}}$ に対しても高い学習効果を持つことが確認された．このような高い汎化性能は，テキストデータとして合成データを構築可能な提案法の利点をさらに強調する結果であると言える．

In-context learning における性能　提案法の汎化性能に関するさらなる分析として，大規模言語モデルの in-context learning における性能を評価した．表 3 に提案法で生成した SST-2 の合成データセットを，GPT-2-XL，OPT，Llama 2 の 3 つの異なるサイズのモデルの 5-shot プロンプトとして利用した場合の性能を示す．実験結果から，提案法で獲得した合成データセットは勾配法による学習だけでなく，in-context learning に対しても効果的であることがわかった．

表 2　蒸留時と異なるモデルへの汎化性能 (DPC=20)．(**S**) は提案法の蒸留時に利用した分類モデル，および K-centers の Encoder を示す．

Dataset	**Model**	**Random**	**K-centers**	**提案法**
SST-2	$\text{BERT}_{\text{BASE}}$ (**S**)	70.3±6.8	79.8±3.5	80.3±2.8
	$\text{RoBERTa}_{\text{BASE}}$	74.4±5.3	73.9±5.2	**78.1±3.8**
	$\text{BERT}_{\text{LARGE}}$	74.7±8.4	80.4±9.1	**83.1±6.2**
	$\text{XLNet}_{\text{BASE}}$	69.9±6.2	71.8±5.8	**77.9±4.7**
QQP	$\text{BERT}_{\text{BASE}}$ (**S**)	59.1±3.8	62.6±2.7	64.4±2.6
	$\text{RoBERTa}_{\text{BASE}}$	60.1±4.0	63.9±3.2	**66.4±2.3**
	$\text{BERT}_{\text{LARGE}}$	58.8±6.9	59.0±8.9	**62.9±8.6**
	$\text{XLNet}_{\text{BASE}}$	59.1±3.5	60.9±3.0	**64.4±2.2**
MNLI-m	$\text{BERT}_{\text{BASE}}$ (**S**)	40.1±3.2	45.3±3.0	48.7±2.6
	$\text{RoBERTa}_{\text{BASE}}$	39.6±2.5	44.5±2.6	**45.0±2.8**
	$\text{BERT}_{\text{LARGE}}$	40.9±4.5	48.7±4.2	**49.6±4.4**
	$\text{XLNet}_{\text{BASE}}$	39.0±2.0	43.5±2.7	**44.7±2.7**

表 3　SST-2 の 5-shot プロンプトとしての性能比較．

Models	**Random**	**K-centers**	**提案法**
GPT-2-XL (1.5B)	64.8±12.0	64.8±13.3	**71.1±13.0**
OPT (2.7B)	89.3±5.9	91.5±3.1	**92.7±1.9**
Llama 2 (7B)	93.6±2.9	94.6±0.7	**95.1±0.7**

4　まとめ

本研究では，テキスト生成モデルを利用することで合成データをテキストとして獲得するデータセット蒸留手法を提案した．3 つのテキスト分類データセットを対象とした実験の結果，提案法はコアセット選択よりも高い性能でモデルを学習可能な合成データセットを構築することを示した．さらに提案法で生成した合成データは，蒸留時と異なるモデルの学習だけでなく，大規模言語モデルの in-context learning に対しても有効であることを示した．

参考文献

[1] Tongzhou Wang, Jun-Yan Zhu, Antonio Torralba, and Alexei A. Efros. Dataset distillation. **CoRR**, Vol. abs/1811.10959, , 2018.

[2] Bo Zhao, Konda Reddy Mopuri, and Hakan Bilen. Dataset condensation with gradient matching. In **9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021**. OpenReview.net, 2021.

[3] George Cazenavette, Tongzhou Wang, Antonio Torralba, Alexei A. Efros, and Jun-Yan Zhu. Dataset distillation by matching training trajectories. In **IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops, CVPR Workshops 2022, New Orleans, LA, USA, June 19-20, 2022**, pp. 4749–4758. IEEE, 2022.

[4] Kai Wang, Bo Zhao, Xiangyu Peng, Zheng Zhu, Shuo Yang, Shuo Wang, Guan Huang, Hakan Bilen, Xinchao Wang, and Yang You. Cafe: Learning to condense dataset by aligning features. In **2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)**, pp. 12186–12195, 2022.

[5] Bo Zhao and Hakan Bilen. Dataset condensation with distribution matching. In **IEEE/CVF Winter Conference on Applications of Computer Vision, WACV 2023, Waikoloa, HI, USA, January 2-7, 2023**, pp. 6503–6512. IEEE, 2023.

[6] Felipe Petroski Such, Aditya Rawal, Joel Lehman, Kenneth Stanley, and Jeffrey Clune. Generative teaching networks: Accelerating neural architecture search by learning to generate synthetic training data. In Hal Daumé III and Aarti Singh, editors, **Proceedings of the 37th International Conference on Machine Learning**, Vol. 119 of **Proceedings of Machine Learning Research**, pp. 9206–9216. PMLR, 13–18 Jul 2020.

[7] Dmitry Medvedev and Alexander D'yakonov. Learning to generate synthetic training data using gradient matching and implicit differentiation. In Evgeny Burnaev, Dmitry I. Ignatov, Sergei Ivanov, Michael Yu. Khachay, Olessia Koltsova, Andrei Kutuzov, Sergei O. Kuznetsov, Natalia V. Loukachevitch, Amedeo Napoli, Alexander Panchenko, Panos M. Pardalos, Jari Saramäki, Andrey V. Savchenko, Evgenii Tsymbalov, and Elena Tutubalina, editors, **Recent Trends in Analysis of Images, Social Networks and Texts - 10th International Conference, AIST 2021, Tbilisi, Georgia, December 16-18, 2021, Revised Supplementary Proceedings**, Vol. 1573 of **Communications in Computer and Information Science**, pp. 138–150. Springer, 2021.

[8] Jie Zhang, Chen Chen, Bo Li, Lingjuan Lyu, Shuang Wu, Shouhong Ding, Chunhua Shen, and Chao Wu. Dense: Data-free one-shot federated learning. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, editors, **Advances in Neural Information Processing Systems**, Vol. 35, pp. 21414–21428. Curran Associates, Inc., 2022.

[9] Yuanhao Xiong, Ruochen Wang, Minhao Cheng, Felix Yu, and Cho-Jui Hsieh. Feddm: Iterative distribution matching for communication-efficient federated learning. In **IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2023, Vancouver, BC, Canada, June 17-24, 2023**, pp. 16323–16332. IEEE, 2023.

[10] Felix Wiewel and Bin Yang. Condensed composite memory continual learning. In **International Joint Conference on Neural Networks, IJCNN 2021, Shenzhen, China, July 18-22, 2021**, pp. 1–8. IEEE, 2021.

[11] Mattia Sangermano, Antonio Carta, Andrea Cossu, and Davide Bacciu. Sample condensation in online continual learning. In **International Joint Conference on Neural Networks, IJCNN 2022, Padua, Italy, July 18-23, 2022**, pp. 1–8. IEEE, 2022.

[12] Tian Dong, Bo Zhao, and Lingjuan Lyu. Privacy for free: How does dataset condensation help privacy? In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato, editors, **Proceedings of the 39th International Conference on Machine Learning**, Vol. 162 of **Proceedings of Machine Learning Research**, pp. 5378–5396. PMLR, 17–23 Jul 2022.

[13] Dingfan Chen, Raouf Kerkouche, and Mario Fritz. Private set generation with discriminative information. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, editors, **Advances in Neural Information Processing Systems**, Vol. 35, pp. 14678–14690. Curran Associates, Inc., 2022.

[14] Ilia Sucholutsky and Matthias Schonlau. Soft-label dataset distillation and text dataset distillation. In **2021 International Joint Conference on Neural Networks (IJCNN)**, pp. 1–8, 2021.

[15] Yongqi Li and Wenjie Li. Data distillation for text classification. **CoRR**, Vol. abs/2104.08448, , 2021.

[16] Aru Maekawa, Naoki Kobayashi, Kotaro Funakoshi, and Manabu Okumura. Dataset distillation with attention labels for fine-tuning BERT. In **Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)**, pp. 119–127, Toronto, Canada, July 2023. Association for Computational Linguistics.

[17] Shivam Sahni and Harsh M. Patel. Exploring multilingual text data distillation. **CoRR**, Vol. abs/2308.04982, , 2023.

[18] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. **OpenAI blog**, Vol. 1, No. 8, p. 9, 2019.

[19] Bo Zhao and Hakan Bilen. Dataset condensation with differentiable siamese augmentation. In Marina Meila and Tong Zhang, editors, **Proceedings of the 38th International Conference on Machine Learning**, Vol. 139 of **Proceedings of Machine Learning Research**, pp. 12674–12685. PMLR, 18–24 Jul 2021.

[20] Tatsuya Hiraoka, Sho Takase, Kei Uchiumi, Atsushi Keyaki, and Naoaki Okazaki. Optimizing word segmentation for downstream task. In **Findings of the Association for Computational Linguistics: EMNLP 2020**, pp. 1341–1351, Online, November 2020. Association for Computational Linguistics.

[21] Yanqing Liu, Jianyang Gu, Kai Wang, Zheng Zhu, Wei Jiang, and Yang You. Dream: Efficient dataset distillation by representative matching. In **Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)**, pp. 17314–17324, October 2023.

[22] Gert W. Wolf. Facility location: concepts, models, algorithms and case studies. series: Contributions to management science. **Int. J. Geogr. Inf. Sci.**, Vol. 25, No. 2, pp. 331–333, 2011.

[23] Ozan Sener and Silvio Savarese. Active learning for convolutional neural networks: A core-set approach. In **6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings**. OpenReview.net, 2018.

[24] Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 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**, pp. 353–355, Brussels, Belgium, November 2018. Association for Computational Linguistics.

[25] Max Welling. Herding dynamical weights to learn. In **Proceedings of the 26th Annual International Conference on Machine Learning**, ICML '09, p. 1121–1128, New York, NY, USA, 2009. Association for Computing Machinery.

A　追学習のアルゴリズムの詳細

Algorithm 1 に提案法における勾配の一致度に基づく追学習のアルゴリズムの概要を示す．従来の勾配の一致度を利用するデータセット蒸留の手法 [2, 19] と同様に，S ステップからなる outer-loop と，T ステップからなる inner-loop で構成される．outer-loop では，各ステップの開始時に分類モデルのパラメータ θ の初期化を行う．inner-loop では，各クラスラベルごとに勾配の一致度に基づく損失を計算し，その平均に基づいてテキスト生成モデルのパラメータ ϕ を更新する．また，ϕ の更新後に，訓練データセットの実サンプルを利用して分類モデルを K ステップ学習する．これにより，テキスト生成モデルは，異なる初期値，異なる学習段階の分類モデルのパラメータ θ の勾配を考慮して学習される．

Algorithm 1: 追学習のアルゴリズムの概要

Input : D_{real}: original dataset; ϕ: generator model; θ: learner model; S: # of outer loop; T: # of inner loop; K: # of learner updating loop in each inner step; M: batch size of real data; N: batch size of synthetic data; η: learning rate of θ; α: learning rate of ϕ.

// Outer loop
for $s = 1, \ldots, S$ **do**
  // Initialize learner
  Initialize $\theta \sim p(\theta_0)$
  // Inner loop
  for $t = 1, \ldots, T$ **do**
    // Compute gradient matching loss for each class
    for $c = 1, \ldots, C$ **do**
      // Compute loss with real samples
      $\{x_i^{(c)}\}_{i=1}^{M} \sim D_{\text{real}}^{(c)}$
      $L_{\text{real}}^{(c)} \leftarrow \frac{1}{M} \sum_{i=1}^{M} l_\theta(x_i^{(c)})$
      // Compute loss with synthetic samples
      $\{\tilde{x}_i^{(c)}\}_{i=1}^{N} \sim p_\phi(\tilde{x})$
      for $i = 1, \ldots, N$ **do**
        $a_i \leftarrow p_\phi(\tilde{x}_i^{(c)}) / \sum_{j=1}^{N} p_\phi(\tilde{x}_j^{(c)})$
      $L_{\text{syn}}^{(c)} \leftarrow \sum_{i=1}^{N} a_i l_\theta(\tilde{x}_i^{(c)})$
      // Gradient matching loss
      $L_{\text{GM}}^{(c)} \leftarrow D(\nabla_\theta L_{\text{real}}^{(c)}, \nabla_\theta L_{\text{syn}}^{(c)}))$
    // Update generator
    $\phi \leftarrow \phi - \alpha \nabla_\phi \frac{1}{C} \sum_{c=1}^{C} L_{\text{GM}}^{(c)}$
    // Update learner for K steps
    for $k = 1, \ldots, K$ **do**
      $X_{\text{real}} \sim D_{\text{real}}$
      $\theta \leftarrow \theta - \eta \nabla_\theta L_\theta(X_{\text{real}})$

Output: ϕ: parameters of generator model.

B　提案法の分析

表 4 に提案法における代表サンプルを利用した教師勾配，および多様性を考慮したミニバッチ生成に関する ablation study の実験結果を示す．実験結果から，いずれの手法についても提案法に対して効果的であることがわかる．

表 4　提案法に関する ablation study．(a) は代表サンプルを利用した教師勾配，(b) は多様性を考慮したミニバッチ生成をそれぞれ表す．

(a)	(b)	SST-2	QQP	MNLI-m
✓	✓	**72.5 ± 5.9**	**58.8 ± 5.2**	**39.7 ± 2.7**
✓	-	71.3 ± 5.6	57.5 ± 4.4	38.8 ± 3.0
-	✓	70.9 ± 5.9	57.6 ± 5.0	39.5 ± 2.8
-	-	69.2 ± 6.2	57.7 ± 5.2	38.3 ± 2.8

C　実験設定の詳細

表 5 に学習および評価時に使用したハイパーパラメータ設定を示す．

表 5　実験に使用したハイパーパラメータ設定．

提案法の事前学習に関する設定 (2.1 節)	
Optimizer	AdamW
Learning rate	1.0×10^{-5}
Learning rate scheduler	Linear warm-up and cosine annealing
Warmup ratio	0.05
Waight decay	0.01
Gradient clipping	1.0
Dropout ratio	0.1
# of training steps	80,000
Batch size	64
提案法の追学習に関する設定 (2.2 節)	
Optimizer	AdamW
Learning rate	3.0×10^{-7}
Learning rate scheduler	Linear warm-up and cosine annealing
Warmup ratio	0.05
Waight decay	0.01
Gradient clipping	1.0
Dropout ratio	0.1
# of outer loop (S)	20,000
# of inner loop (T)	10
# of learner updating steps (K)	20
Batch size of real samples (M)	200
Batch size of synthetic samples (N)	64
Generation interval (I_{int})	200
評価時の分類モデルの学習に関する設定	
Oprimizer	AdamW
Learning rate	1.0×10^{-4}
Learning rate scheduler	Linear warm-up and cosine annealing
Warmup ratio	0.5
Waight decay	0.01
Gradient clipping	1.0
Dropout ratio	0.1
# of training steps	200
Batch size	64