DEEP LEARNING JP
[DL Papers]

"BEYOND AUTOREGRESSION: FAST LLMS VIA SELF-DISTILLATION THROUGH TIME"

2024.11.07 Toshiharu Maeba, Matsuo-Iwasawa Lab (M1)

http://deeplearning.jp/

書誌情報

紹介論文

タイトル　BEYOND AUTOREGRESSION:
FAST LLMS VIA SELF-DISTILLATION THROUGH TIME

出典: Arxiv(2024.10)preprint

著者: Justin Deschenaux, Caglar Gulcehre
CLAIRE Lab, School of Computer and Communication

概要

- Self-Distillation Through Time（SDTT）を用いた離散拡散言語モデル
- サンプリングステップ数を減らすことで推論を高速化

※画像は出典記載のないものは、本論文から引用

目次

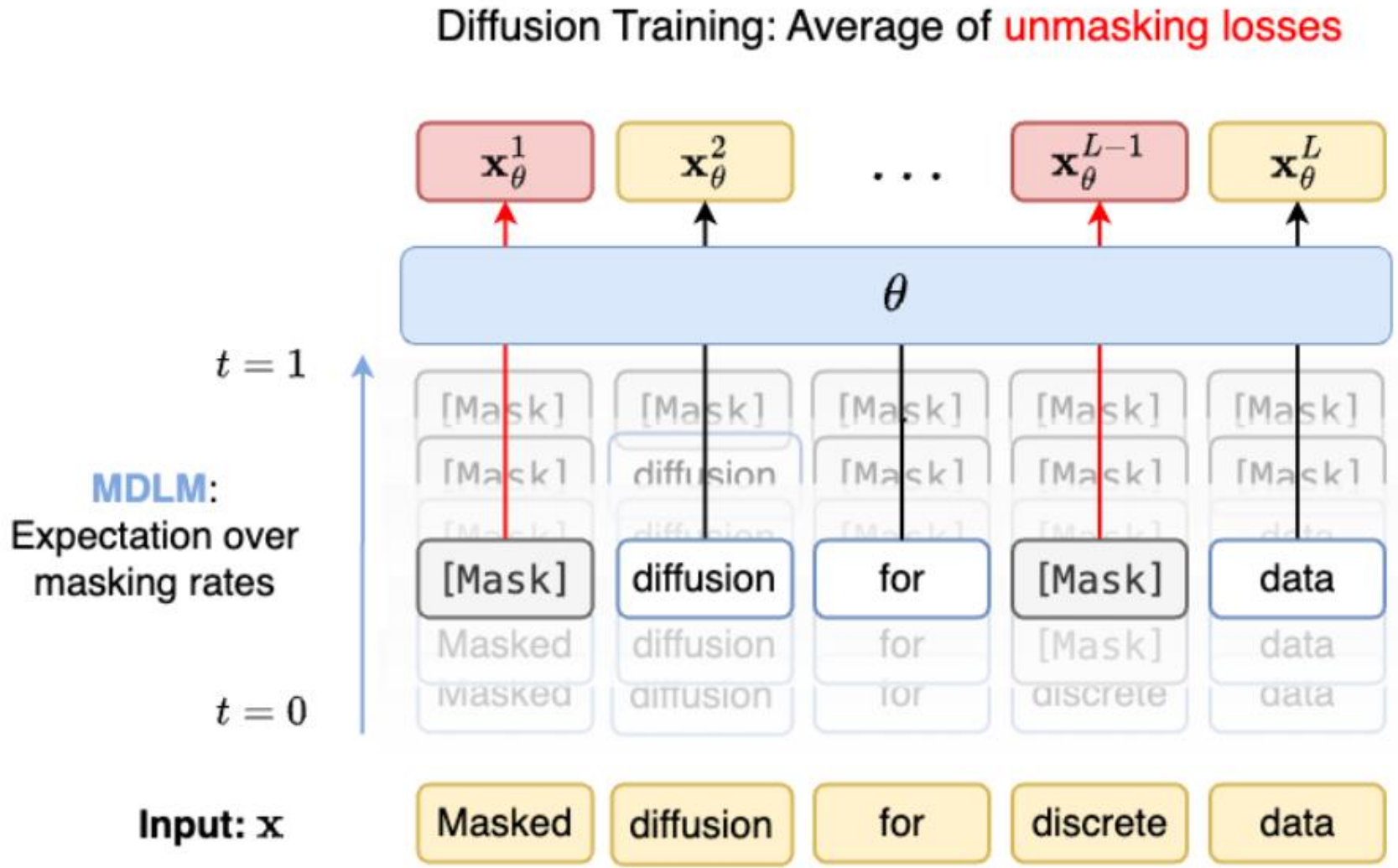
- 概要
- 関連研究
- 手法
- 実験
- まとめ・感想

概要：主な貢献

- SDTT(Self-Distillation Through Time)を導入し、一度に少なくとも32個のトークンを生成することができ、 nucleus samplingによるGPT-2よりも優れたperplexityを実現
- SDTTは非常にシンプルで実装が簡単
- SDTTは、KVキャッシングを使用するARモデルよりも最大8倍高速にトークンを生成可能
- 最大860Mのパラメータを持つ言語モデルに対するSDTTの有効性を実証

関連研究：MASKED DIFFUSION LANGUAGE MODELING

- MDLMは、テキストのマスクされた部分を段階的に復元することで言語生成
- モデルは、テキストにノイズ（マスク）を加えていき、段階的に除去する過程で、隠されたマスク部分を予測し、元のテキストを再構成



関連研究：MASKED DIFFUSION LANGUAGE MODELING

- MDLM ：データにノイズを付加するフォワード プロセスと、データの回復を学習するバックワード プロセス

forward

$$q(\mathbf{z}_t|\mathbf{x}) := \mathrm{Cat}(\mathbf{z}_t; \alpha_t \mathbf{x} + (1-\alpha_t)\boldsymbol{\pi}), \tag{1}$$

$\mathbf{x}$：元のドキュメントxで定義されたone-hot分布
$\boldsymbol{\pi}$：定常分布、トークンをマスク
α_t：ノイズ注入スケジュール、t∈[0, 1]、α_t∈[0, 1]、 α_t はtの減少関数（$\alpha_0 \approx 1$、$\alpha_1 \approx 0$）

backward

$$q(\mathbf{z}_s|\mathbf{z}_t, \mathbf{x}) = \mathrm{Cat}\left(\mathbf{z}_s; \frac{[\alpha_{t|s}\mathbf{z}_t + (1-\alpha_{t|s})\mathbf{1}\boldsymbol{\pi}^\top \mathbf{z}_t] \odot [\alpha_s \mathbf{x} + (1-\alpha_s)\boldsymbol{\pi}]}{\alpha_t \mathbf{z}_t^\top \mathbf{x} + (1-\alpha_t)\mathbf{z}_t^\top \boldsymbol{\pi}}\right). \tag{2}$$

t > s 、$\alpha_{t|s} = \frac{\alpha_t}{\alpha_s}$

損失関数

NELBO は、Ground Truth $\mathbf{x}$とモデル予測 x_θ の間の重み付きクロス エントロピー損失に単純化

$$\mathcal{L}_{\mathrm{NELBO}}^{\infty} = \mathbb{E}_q \int_{t=0}^{t=1} \frac{\alpha_t'}{1-\alpha_t} \log \langle \mathbf{x}_\theta(\mathbf{z}_t, t), \mathbf{x} \rangle \mathrm{d}t. \tag{3}$$

制約
①ノイズ除去されたトークンは、サンプリング中に再マスクされない
②すでにノイズ除去されたトークンは、次のサンプリングステップに持ち越す

関連研究：知識蒸留

- 知識蒸留は、生徒モデルを訓練して、より複雑な教師モデルの予測を模倣する手法
- 蒸留の主な利点の１つは、大規模なLLMからのサンプリングに関連する推論コストを削減しながら、蒸留なしでトレーニングされた小規模なモデルのパフォーマンスを上回ることができること

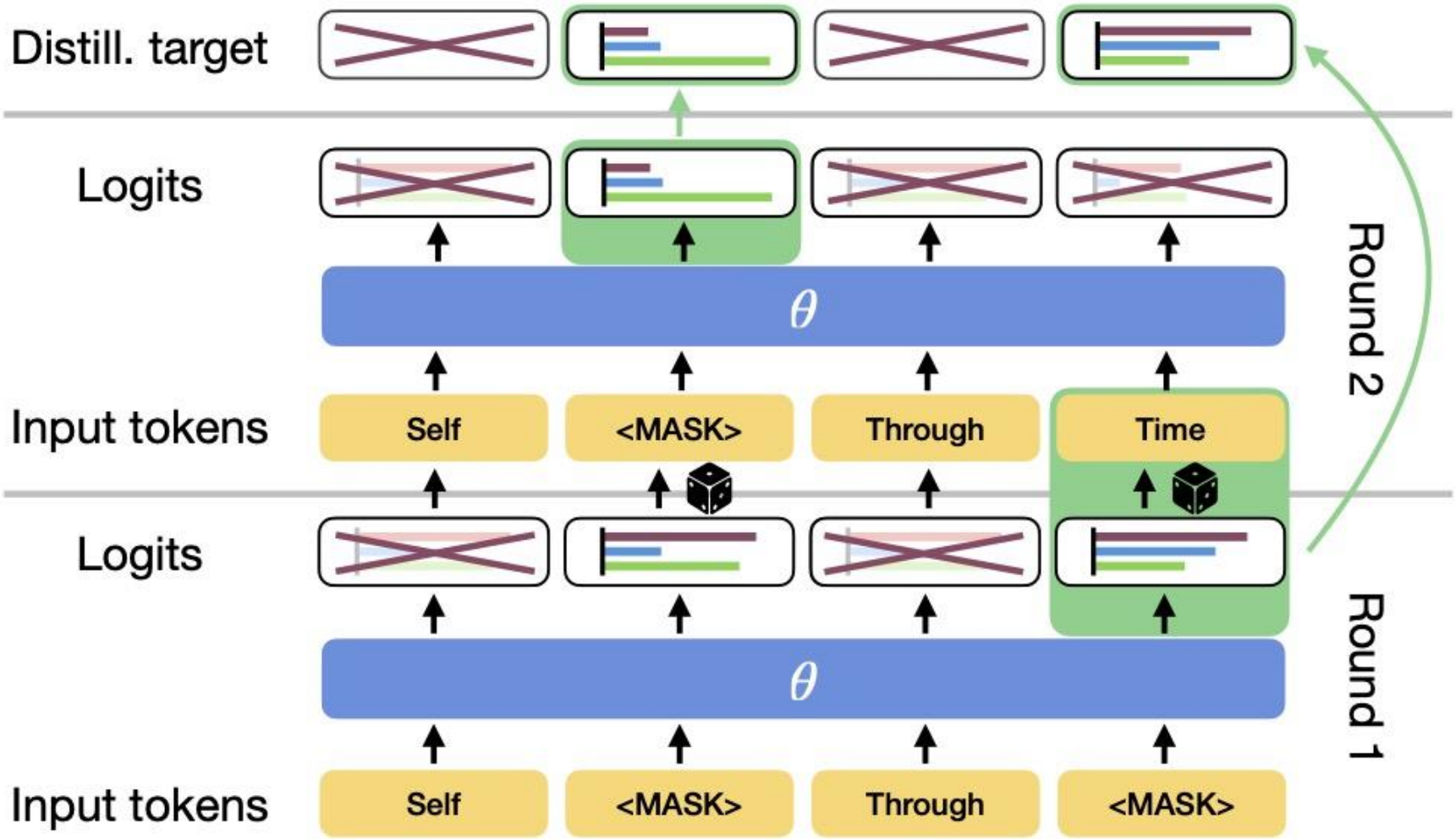
- 今回の手法では、ダイバージェンスδを使用して教師と生徒の予測を一致させる蒸留方法

$$\mathbb{E}_{\mathbf{x}\sim\mathcal{D}}\left[\delta(\mu_s(\mathbf{x}_t|\mathbf{x}_{<t});\mu_t(\mathbf{x}_t|\mathbf{x}_{<t}))\right], \tag{4}$$

μ_s、μ_t：それぞれ生徒と教師のAR分布
D：訓練データセット

提案手法：SDTT

- 通常、ステップ数が少ないと、逆プロセスの近似精度が低下するため、サンプルの品質が低下
- SDTTは、推論時に複数のステップを学生に蒸留することにより、サンプリング速度を向上
- 蒸留した学生を教師として使用して、SDTTを複数回適用



提案手法：SDTT

SDTT は、パラメーター ν のデノイザーを学習させて、$p_\theta^{(m)}$ と $p_\nu^{(k)}$ の間のダイバージェンス d を最小化する問題（k <m、mはkで割り切れるものとする (たとえば、m = 1024 と k = 512)）

$p_\theta^{(m)}$: パラメータ θ のデノイザーを使用して、m ステップで生成されたサンプルの分布

$$\min_{\nu} d(p_\nu^{(k)} || p_\theta^{(m)}). \tag{5}$$

x_θ と x_ν をそれぞれ教師からの多くのステップのデコードに使用されるデノイザー、生徒からの数ステップのデコードに使用されるデノイザーとする

サンプリング プロセスの唯一の学習可能な要素であるため、サンプリング分布 $p_\theta^{(m)}$ と $p_\nu^{(k)}$ を完全に決定

x_θ の予測と一致するように x_ν を学習するために、m/k ステップについて教師からサンプリング

$$\tilde{\mathbf{x}}_\theta^{\text{teacher}}(\mathbf{z}_t, t, m/k)$$

$$\min_{\nu} \mathbb{E}_{\mathbf{z}_0 \sim \mathcal{D}, \mathbf{z}_t \sim q_t(\mathbf{z}_t|\mathbf{z}_0)} \left[\delta(\mathbf{x}_\nu(\mathbf{z}_t, t) || \tilde{\mathbf{x}}_\theta^{\text{teacher}}(\mathbf{z}_t, t, m/k)) \right], \tag{6}$$

提案手法：SDTT（アルゴリズム）

Algorithm 1 Computing the *Self-Distillation Through Time* targets $\tilde{\mathbf{x}}_\theta^{\text{teacher}}(\mathbf{z}_t, t, {}^m/_k)$

Inputs: Noisy tensor $\mathbf{x}_t \in \mathbb{R}^{N \times L}$, Starting sampling time $t_{\text{start}} \in [0,1]^N$, Number of sampling steps ${}^m/_k \geq 2$, such that ${}^m/_k \in \mathbb{N}_+$, Sampling step size $\Delta \in (0,1)$, Mask token index $M \in \mathbb{N}$, Minimal sampling time ϵ.
Output: Distillation targets $\tilde{\mathbf{x}}_\theta^{\text{teacher}}(\mathbf{z}_t, t, {}^m/_k)$

target $\leftarrow$ zeros(N, L, K) ▷ Allocate empty tensor for $\tilde{\mathbf{x}}_\theta^{\text{teacher}}(\mathbf{z}_t, t, {}^m/_k)$
$\mathbf{z} \leftarrow \mathbf{x}_t$
for $i = 0, \ldots, {}^m/_k - 1$ **do**
    $t_{\text{curr}} \leftarrow \max(t_{\text{start}} - i \cdot \Delta, \epsilon)$ ▷ Sampling step for the current time
    $\mathbf{z}_{\text{new}}, \ell_{\text{teacher}} \leftarrow$ `reverse_sample`$(\mathbf{z}, t_{\text{curr}}, \Delta)$ ▷ Updated $\mathbf{z}$ & log-probabilities $x_\theta(\mathbf{z}, t_{\text{curr}})$
    U = $\mathbf{z}_{\text{new}} \neq \mathbf{z}$ ▷ Create mask U of tokens that were denoised
    target$[U] \leftarrow \ell_{\text{teacher}}[U]$ ▷ Extract log-probs for the denoised tokens
    $\mathbf{z} \leftarrow \mathbf{z}_{\text{new}}$ ▷ Update $\mathbf{z}$ for the next iteration
end for
target$[\mathbf{z} == M]$ = $\ell_{\text{teacher}}[\mathbf{z} == M]$ ▷ Use log-probs of the last denoising step for masked tokens
return target ▷ Target log-probs for all masked tokens in $\mathbf{x}_t$

提案手法：SDTT（アルゴリズム）

Algorithm 2 *Self-Distillation Through Time* training loop.

Inputs: Training set $\mathcal{D}$, Teacher $\mathbf{x}_\theta$, Divergence measure δ, Number of sampling steps m/k, Sampling step size Δ, Mask token index M, Total number of training steps H
Output: Distilled student $\mathbf{x}_\nu$.

$\nu \leftarrow \theta$ ▷ Initialize the student with the teacher weights
for $i = 0, ..., H-1$ **do**
  $\mathbf{x}_0 \leftarrow \texttt{sample\_example}\,(\mathcal{D})$ ▷ Sample a training example
  $t_{\text{start}} \sim \mathcal{U}[0, 1]$ ▷ Sample t uniformly at random
  $\mathbf{x}_t \sim q_t(\mathbf{x}_t|\mathbf{x}_0)$ ▷ Forward diffusion process. See eq. (1)
  $\mathbf{x}_{\text{student}} \leftarrow \mathbf{x}_\nu(\mathbf{x}_t, t)$
  $\mathbf{x}_{\text{teacher}} \leftarrow \texttt{teacher\_SDTT}(\mathbf{x}_t, t_{\text{start}}, m/k, \Delta, M, \texttt{1e-5})$ ▷ See algorithm 1
  $\mathcal{L} \leftarrow \delta(\mathbf{x}_{\text{student}}||\mathbf{x}_{\text{teacher}})$ ▷ Compute divergence between student and SDTT targets.
  $\nu \leftarrow \texttt{backprop\_optim}\,(\mathcal{L}, \nu)$ ▷ Update the parameters of the student with AdamW
end for
return $\mathbf{x}_\nu$

実験設定

使用モデルなど

- ベースモデル： diffusion transformer、MDLMの原論文のチェックポイントを再利用
- 訓練データセット：OpenWebTextデータセット

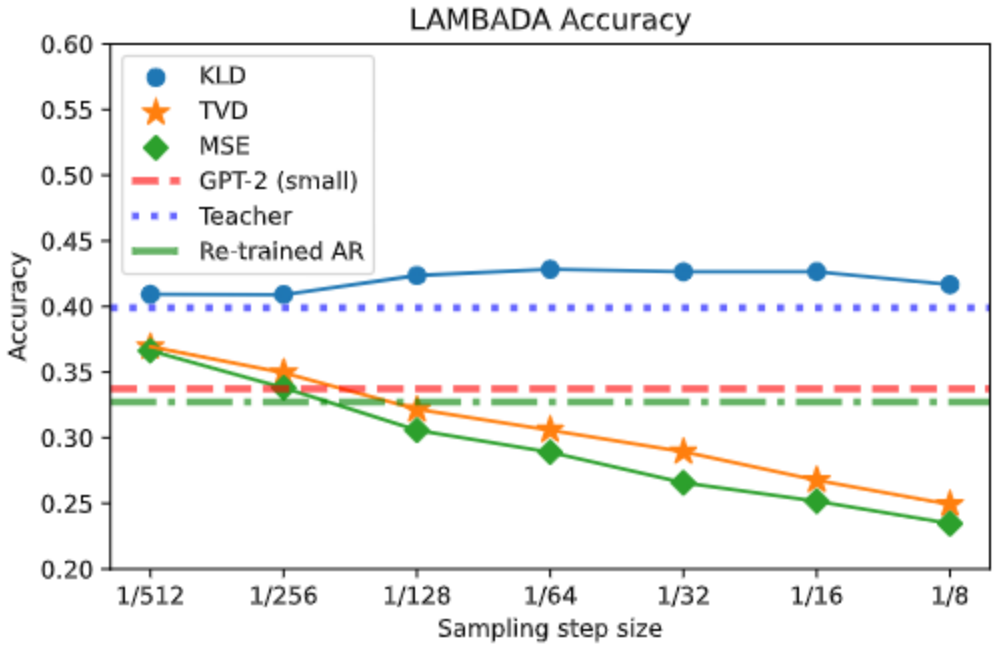
  - 学習率が 6e－5、500ステップで直線的に増加し、その後は一定
  - オプティマイザー：重み減衰がない Adam
  - バッチ サイズが 128、

評価指標

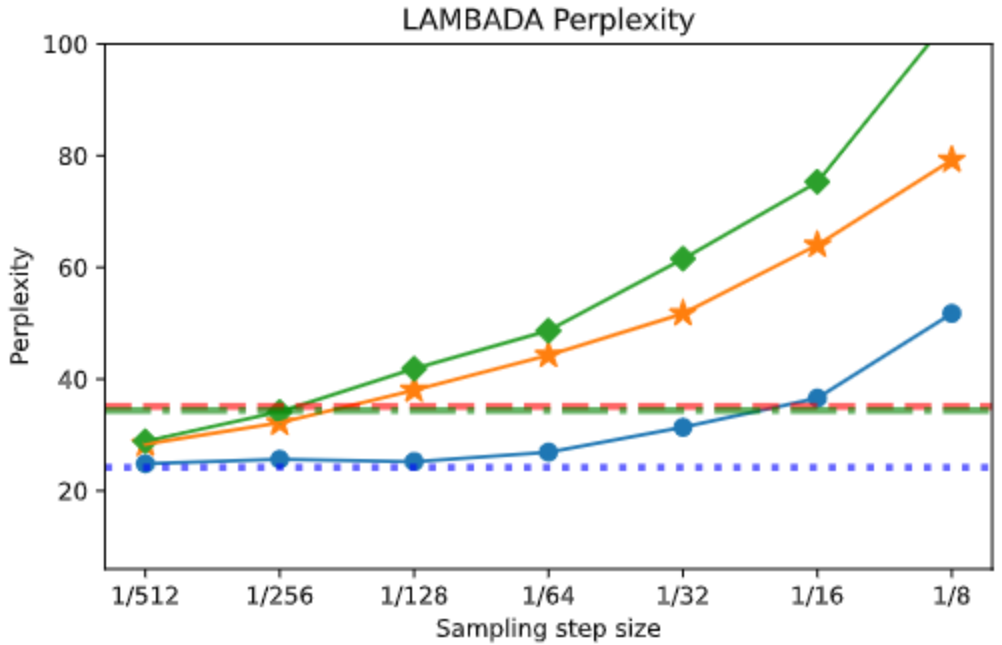
- MAUVE により、モデルがプロンプトにどの程度従うかを評価
- LAMBADAデータセットにより、精度とperplexityを評価
  - 拡散モデルは、マスクされたすべてのトークンが 1 つのステップで正しくデコードされた場合にのみ正解とする

実験：目的関数

- SDTTでは損失関数として使用するダイバージェンスδを選択する必要があるため、平均二乗誤差(MSE)、全変動距離(TVD)、KLダイバージェンス(KLD)のいずれが適しているかを実験
  - KLDが最も良い結果



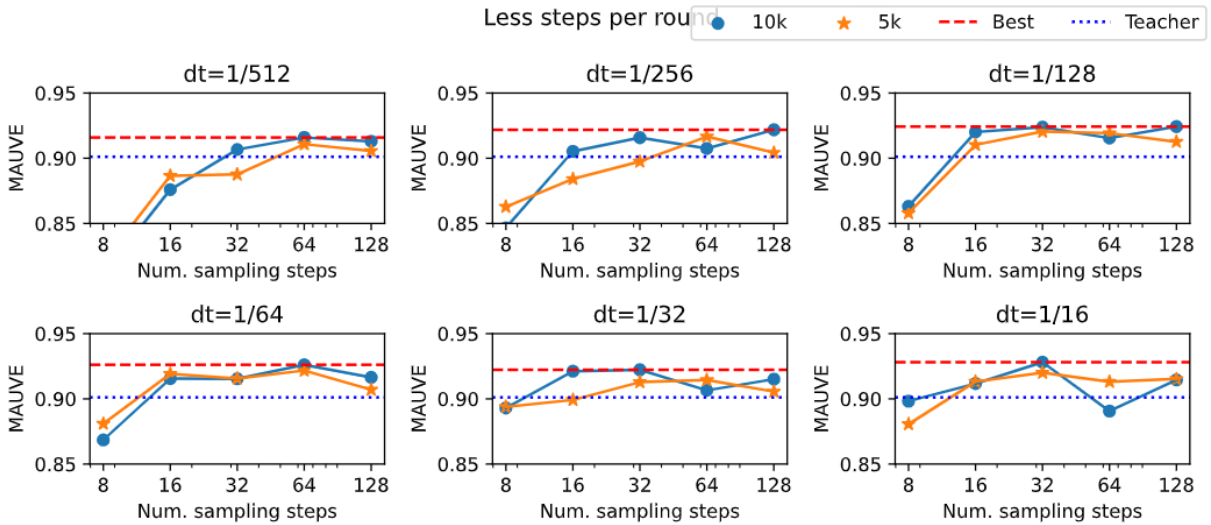
KLDのみ教師を上回る



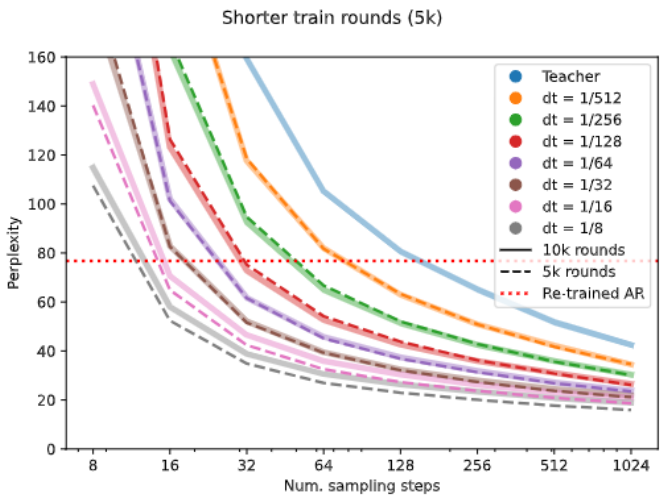
生徒が1024ステップでなく16ステップで訓練してもGPT-2に匹敵する性能

アブレーション：ステップ数

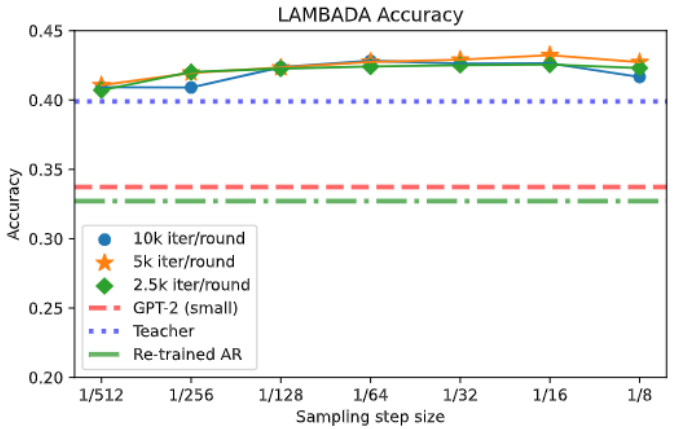
- 蒸留損失の大きさは収束を確実に示していないため、より短いラウンドで実験
- 短いラウンドはわずかに最終的な生成的なperplexityを改善
  - SDTTはELBOを直接最適化しないため、ステップ数が増えるとperplexityの増加が予想

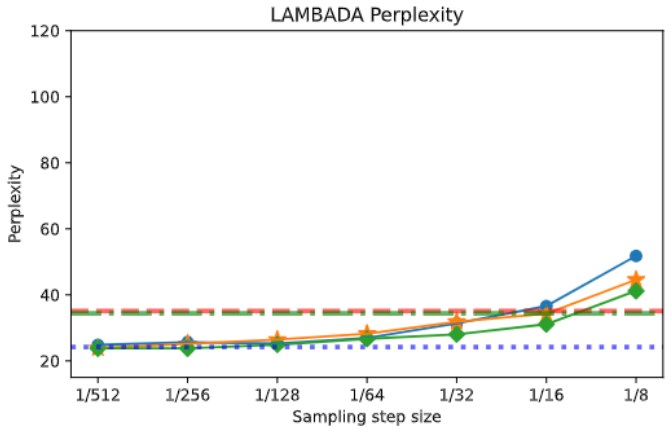


(a) 10k vs 5k iter/round.



(a) 10k vs 5k iter/round.

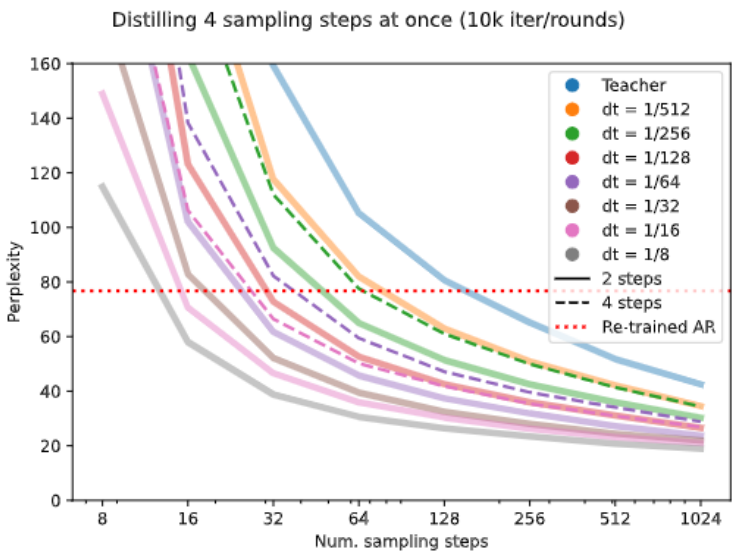




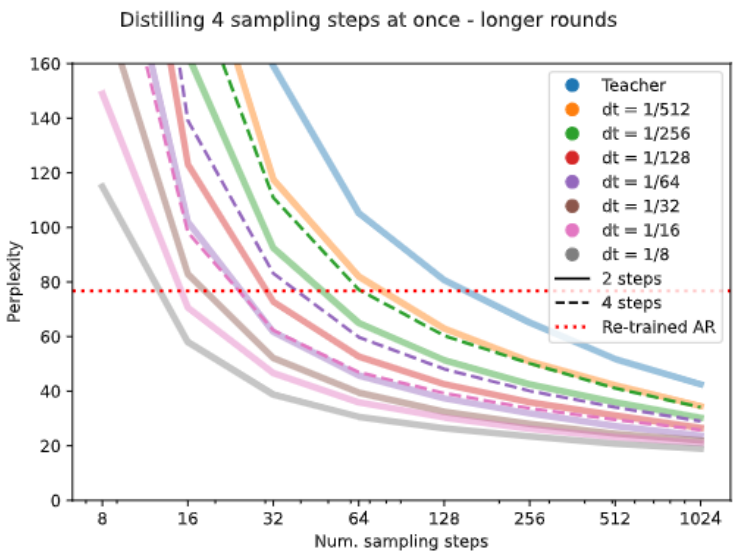
LAMBADAの精度は、ラウンドが短くても変化しなかった

アブレーション：教師のsampling steps

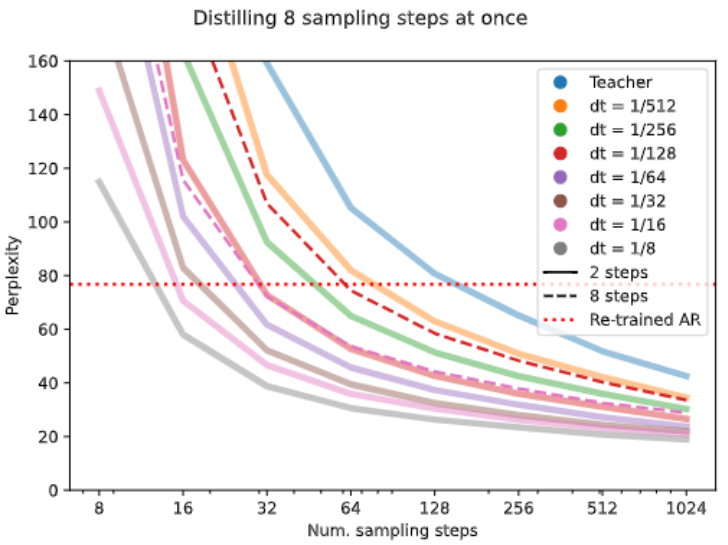
- 教師モデルのターゲットは２サンプリングステップ→変化させるとどうなるか実験
  - 全体として、一度に2ステップ以上を蒸留すると、パフォーマンスが低下
    - この結果は、4ステップまたは8ステップで生成されたターゲットの不確実性が高いため、生徒にとってタスクが難しすぎることを示唆



(a) 4 steps.



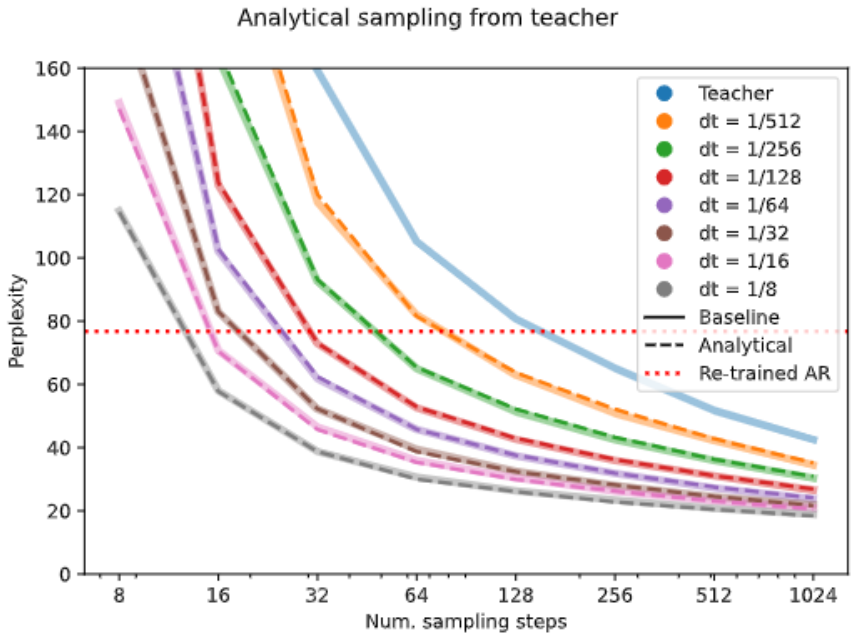
(b) 4 steps and 15k iter/round.



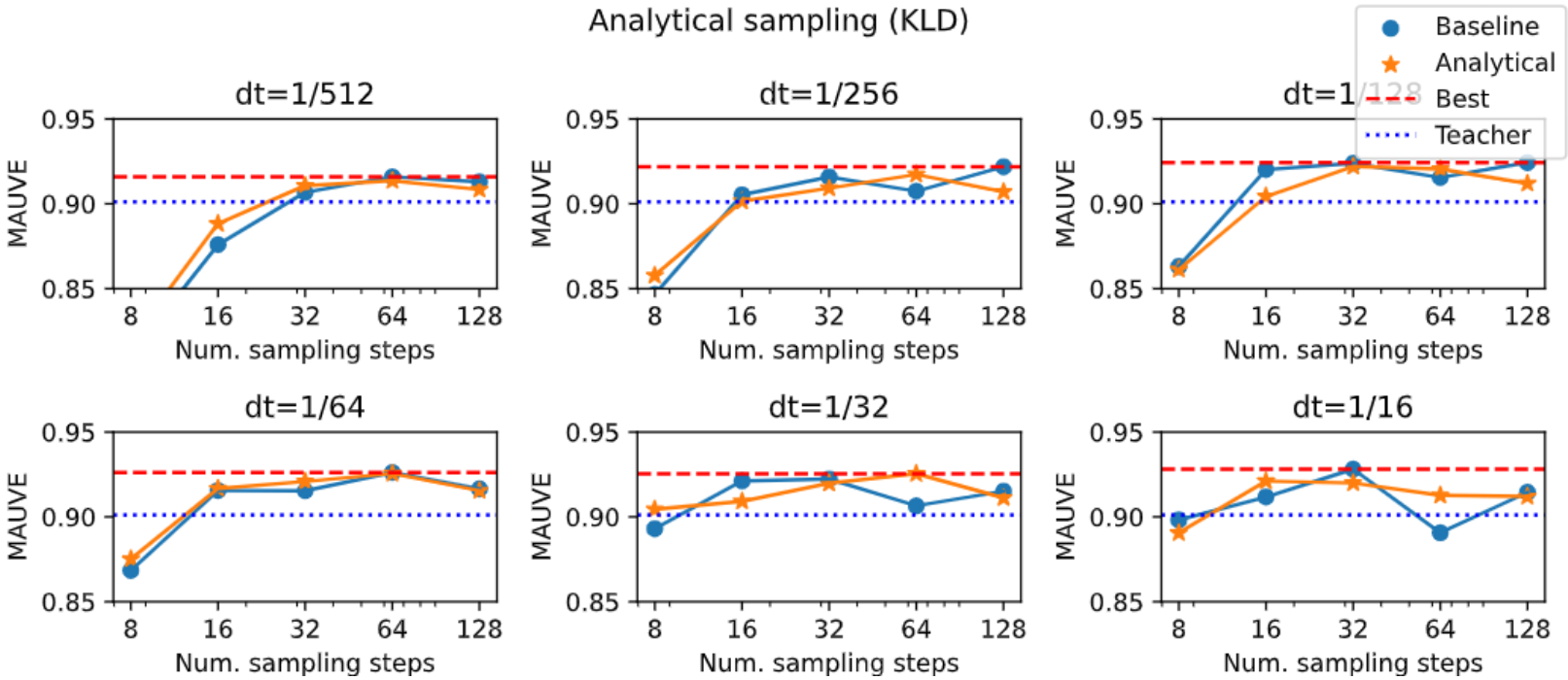
(c) 8 steps.

アブレーション：sampler

- Analytical sampler はより高品質なサンプルが得られることが知られているため、ancestral sampling（ここまで使用）とanalytical samplerを比較
  - Sampler による差はほとんどない

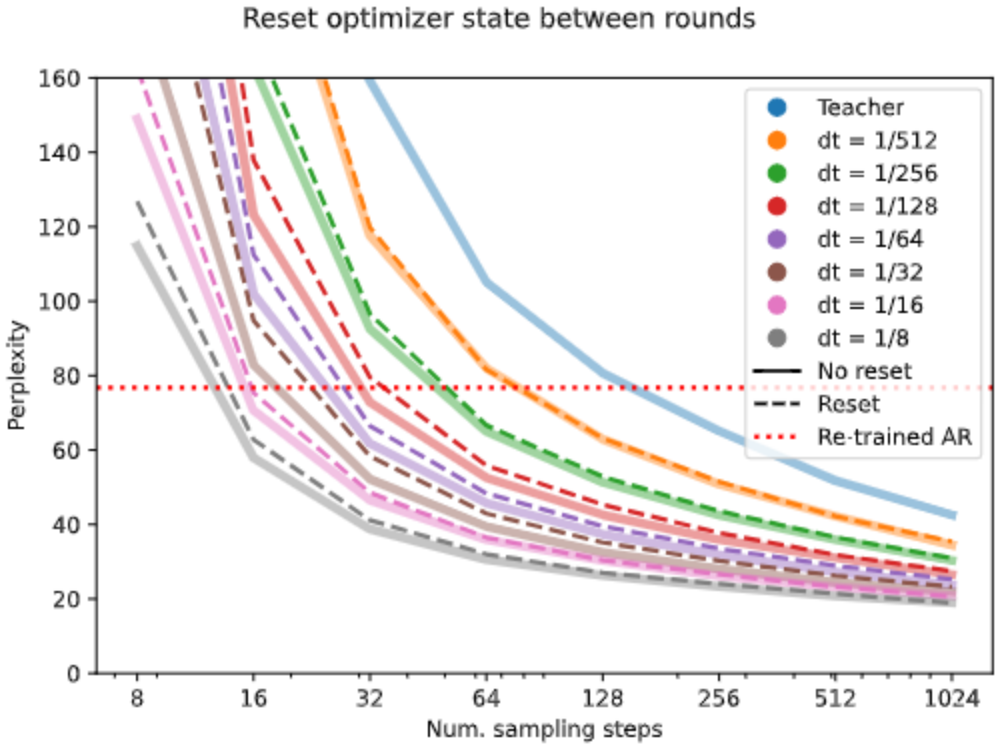


(a) Generative perplexity.

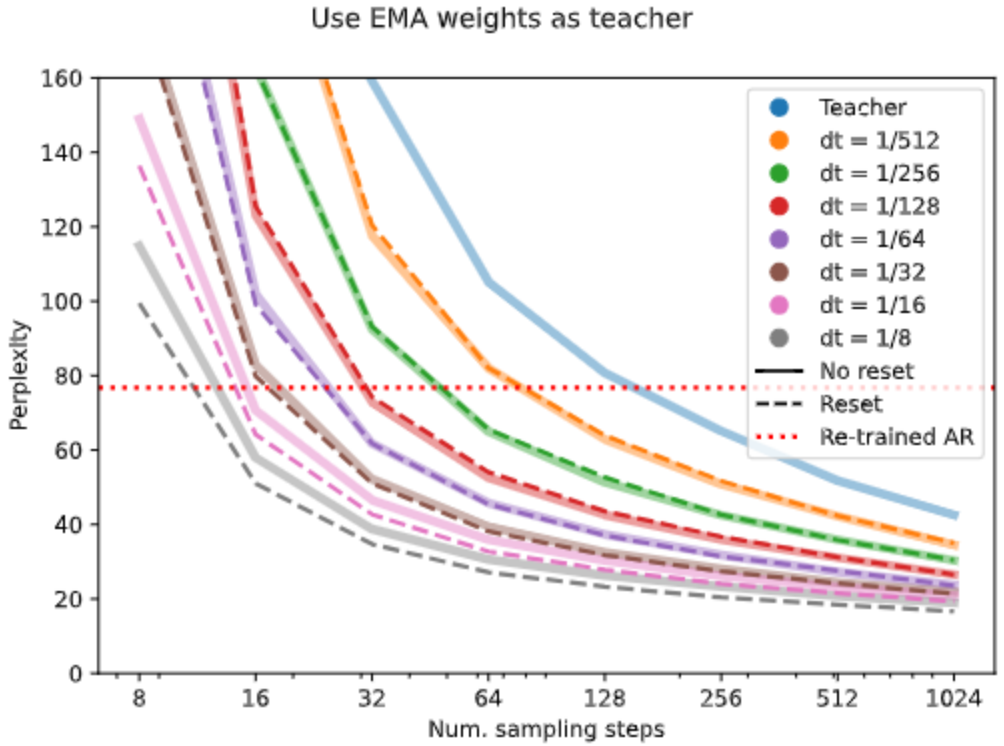


アブレーション：EMA

- 重みの指数移動平均(EMA)を使用すると、拡散モデルからのサンプルの品質が向上することが知られているため、教師モデルに使用する場合を実験
  - オプティマイザのみのリセットは効果がない
  - 重みのEMAを教師に使用すると、パフォーマンスがわずかに向上する可能性



(a) Resetting the optimizer state between rounds.

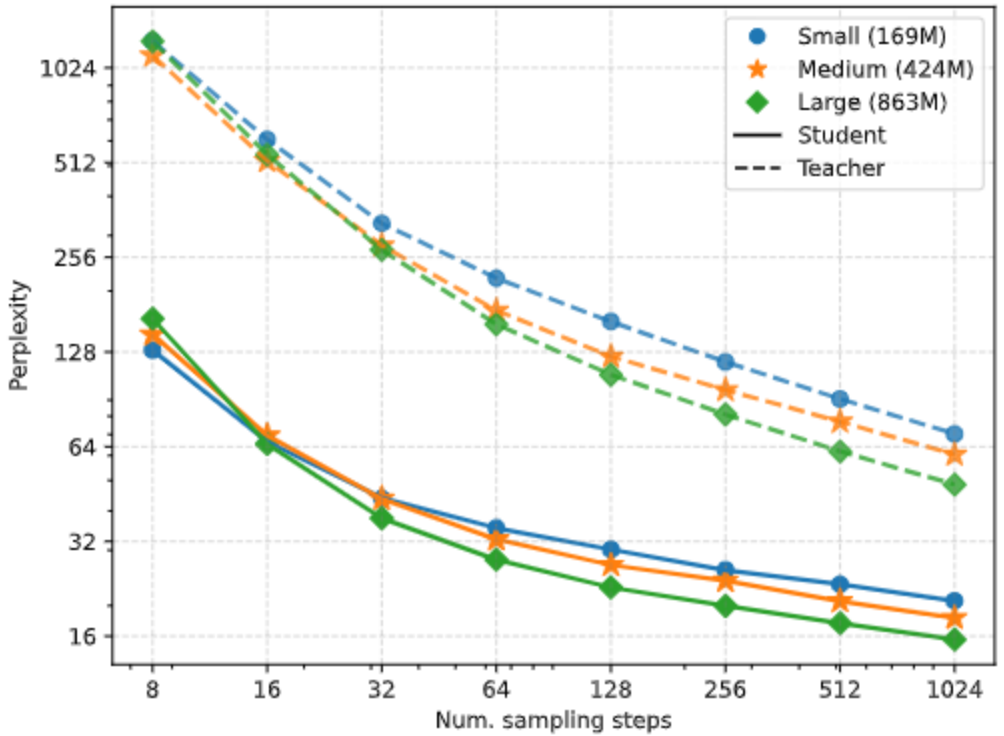


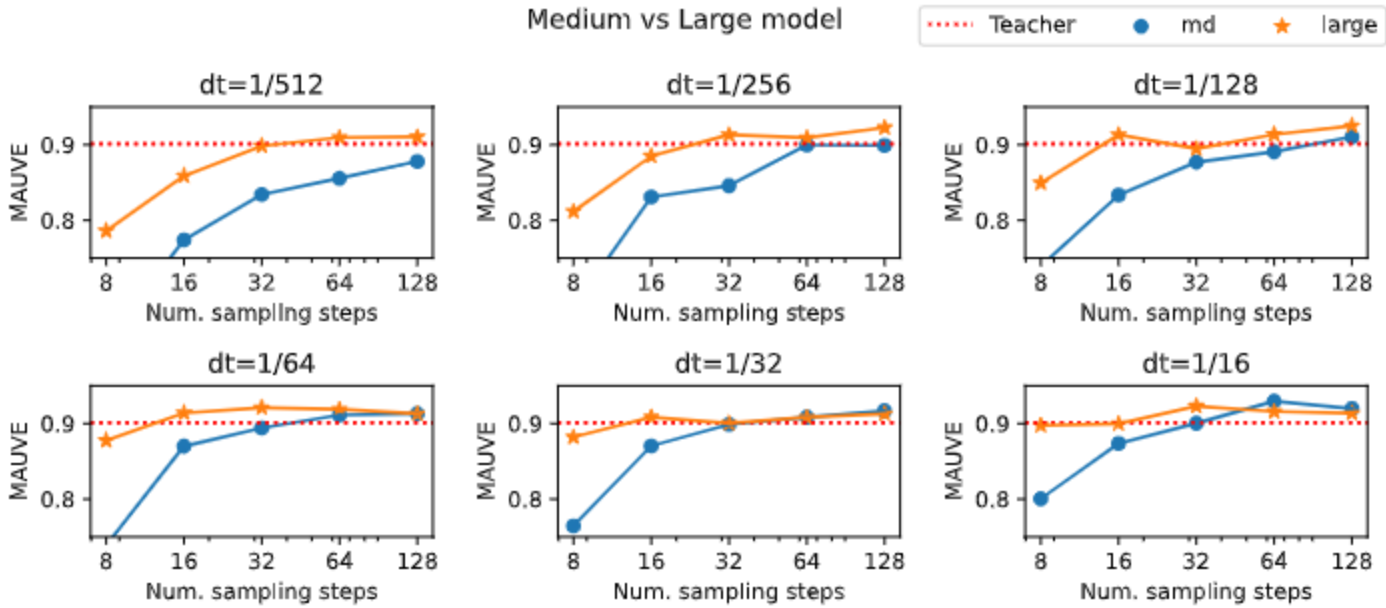
(b) Reset the optimizer state and use EMA of weights as teacher.

実験結果：スケーリング

- SDTTを最大860Mのパラメータを持つ大規模な離散拡散モデルに適用
  - 蒸留により、性能が改善していることから、大きなモデルにも適用可能な手法

Model size	small	medium	large
# params	169M	424M	863M
Num Layers	12	24	24
Embedding dim.	768	1024	1536
Num. heads	12	16	16

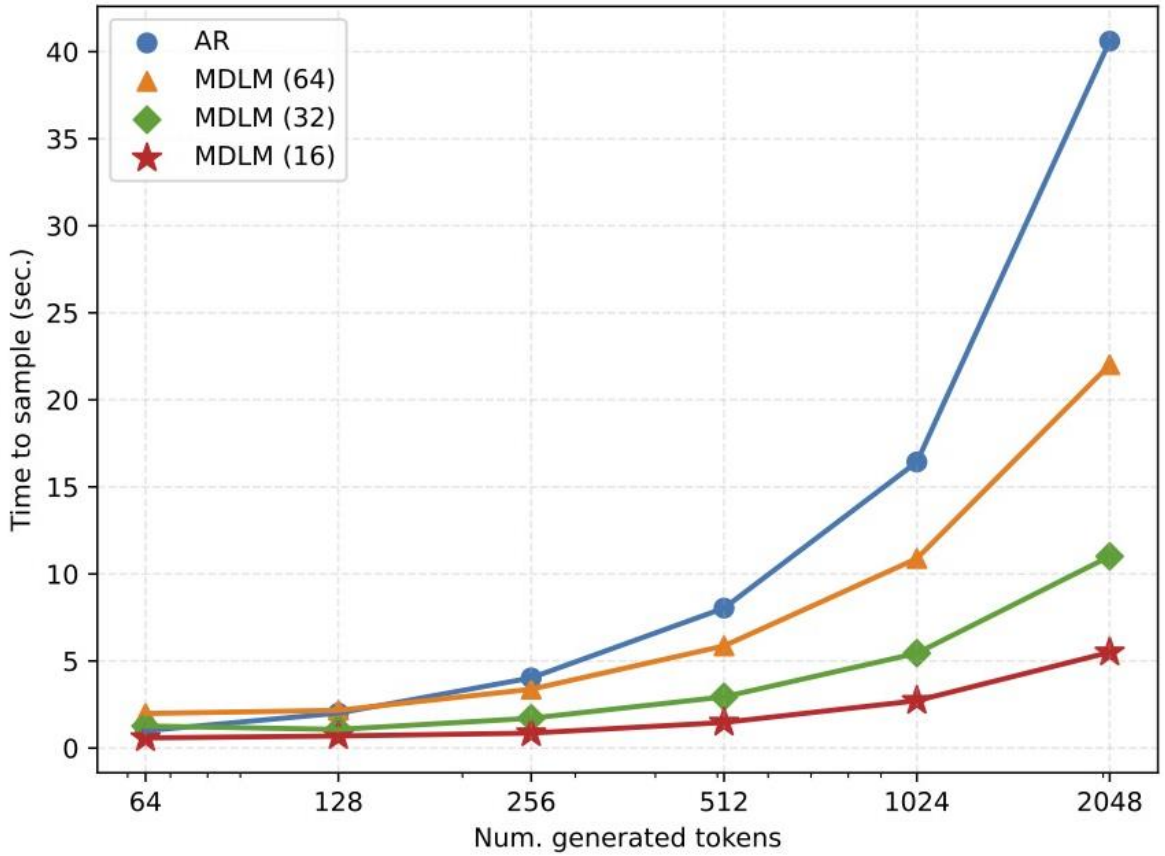




小型のモデルが、最大モデルよりも優れたperplexityを達成

実験結果：latency

- SDTTのレイテンシを、KVcachingを使用する自己回帰モデルと比較
  - 32ステップでサンプリングすると4倍、16ステップだと８倍の改善



まとめ・考察

まとめ

- SDTTは、パフォーマンスを維持しながらデコードステップの数を削減
- 生徒モデルは、KVキャッシングを使用するARモデルよりも最大8倍高速
- SDTTはより大きなモデルにも適用可能
- 今後の研究では、基本言語モデルから大量の補完を生成するタスクでSDTTを評価する予定

感想

- 高速化が実現されているが、理論的な説明が少ない
- 現状では、評価指標が少ないため、今後の研究に期待