ニューラル機械翻訳における ミニバッチ構成法の違いによる 影響の調査

奈良先端科学技術大学院大学
知能コミュニケーション研究室
森下 睦・小田 悠介・Graham Neubig・吉野 幸一郎・
須藤 克仁・中村 哲

言語処理学会第23回年次大会
2017/03/16

NAIST®

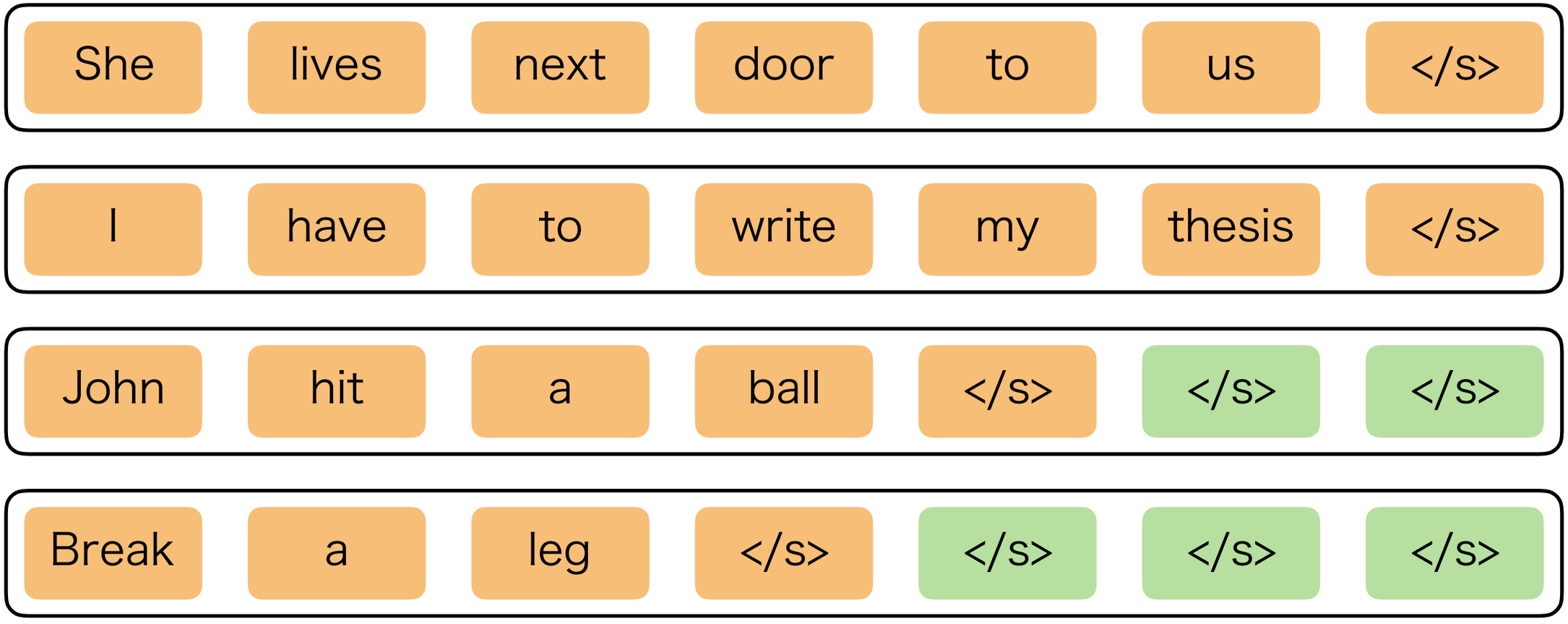
ニューラル機械翻訳におけるミニバッチ

- ニューラル機械翻訳の学習には時間がかかる
  - ミニバッチを用いた高速化が主流
- ミニバッチ: 複数のデータをまとめて学習
  - 高速化: 行列計算の回数が減るため
    - 特に行列計算が得意なGPUを用いた場合顕著
  - 勾配更新の安定化:
    ミニバッチ中の平均lossを用いて勾配更新を行う

パディング

- NMTの学習データは可変長
  - 可変長データを固定長に揃える必要がある

  → パディング



パディングの影響

- 1イテレーションにかかる時間は
  ミニバッチ内の最長文長に依存する
  - 短い文はパディングされて，最長文長となる.

- 改善案 [Sutskever et al., 2014; Bahdanau et al., 2015]
  - 事前にコーパスを文長でソートする
  - ソートすることで，近い文長が一つのミニバッチに固まる
  - パディングが減り，学習速度向上につながる

現在の問題

- ほぼ全てのツールが先行研究を信じて
  コーパスをソート
  - 確かに1イテレーションにかかる時間は短い
  - でも，ソートした際の収束時間は？
  - 色々なソート手法があるけど，結局どれが良いの？
- 様々なミニバッチ作成手法を
  各ツールが独自に実装
  - 誰もその影響を調査していない

本研究の目的

- ミニバッチ構成法の違いが
  NMTの学習に与える影響を調査
- 最善のミニバッチ構成法を調査

本研究で比較するミニバッチ構成法

- **コーパスのソート手法**
  - ソート時の基準
- **ミニバッチサイズ**
  - 一つのミニバッチに含まれる文数
- **ミニバッチサイズの単位**
  - ミニバッチサイズを決定する基準
  - 文数 or 目的言語文の単語数

コーパスのソート手法

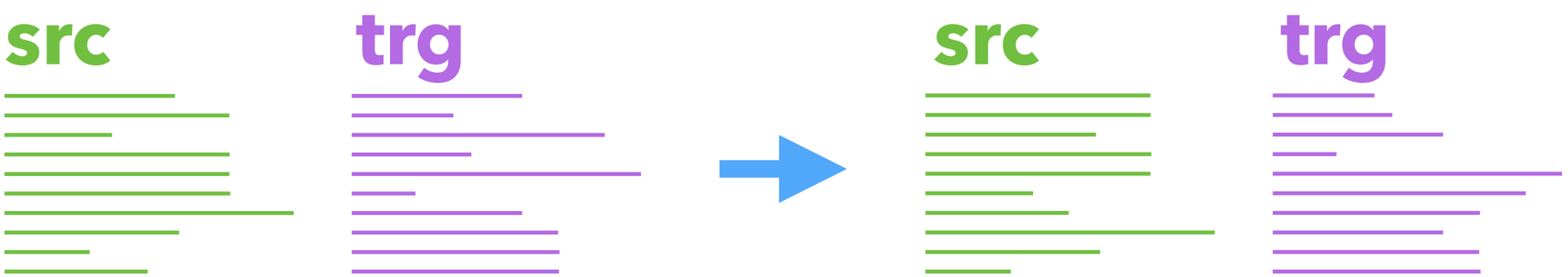
コーパスのソート手法

- 先行研究「コーパスはソートした方が良い」
  - ソートの種類はいろいろ
- 本研究では5つのソート手法を比較する

コーパスのソート手法

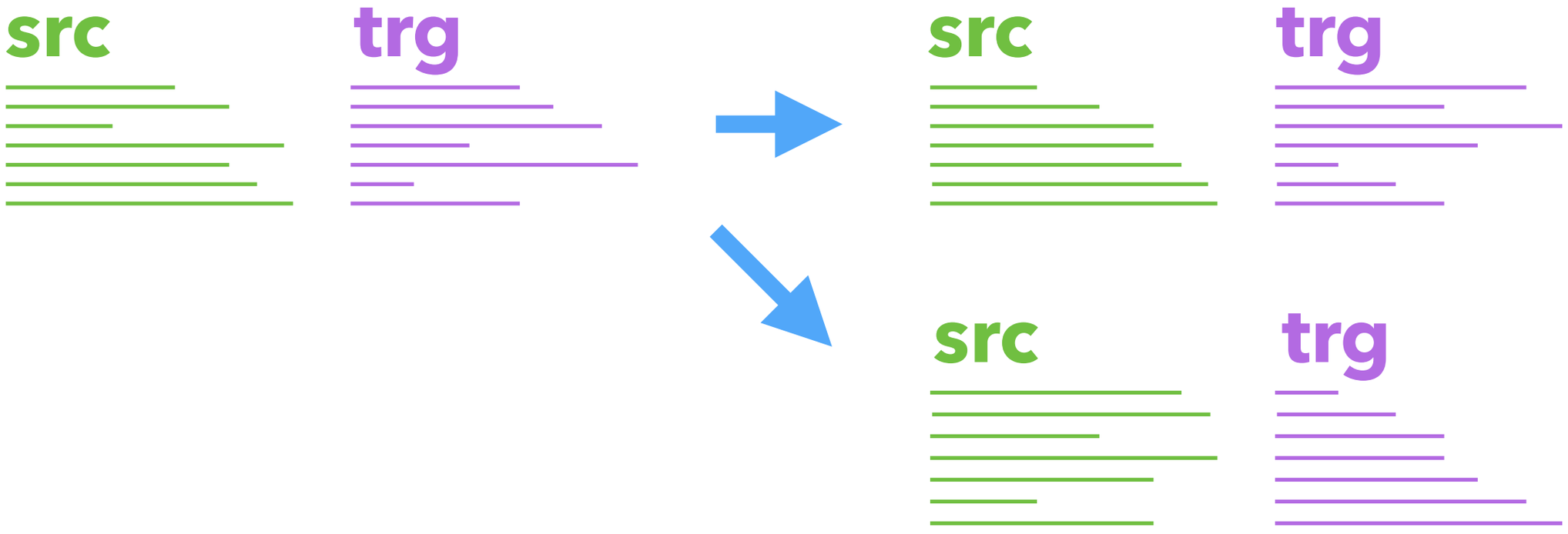
- 検討するソート手法

  - shuffle: ソートせずコーパスをシャッフル



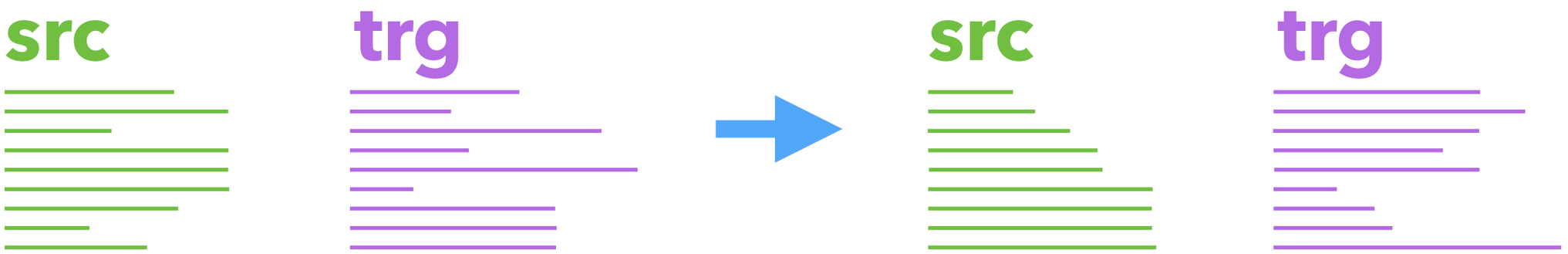
コーパスのソート手法

- 検討するソート手法
  - src: 原言語文長に従いコーパスをソート
  - trg: 目的言語文長に従いコーパスをソート



コーパスのソート手法

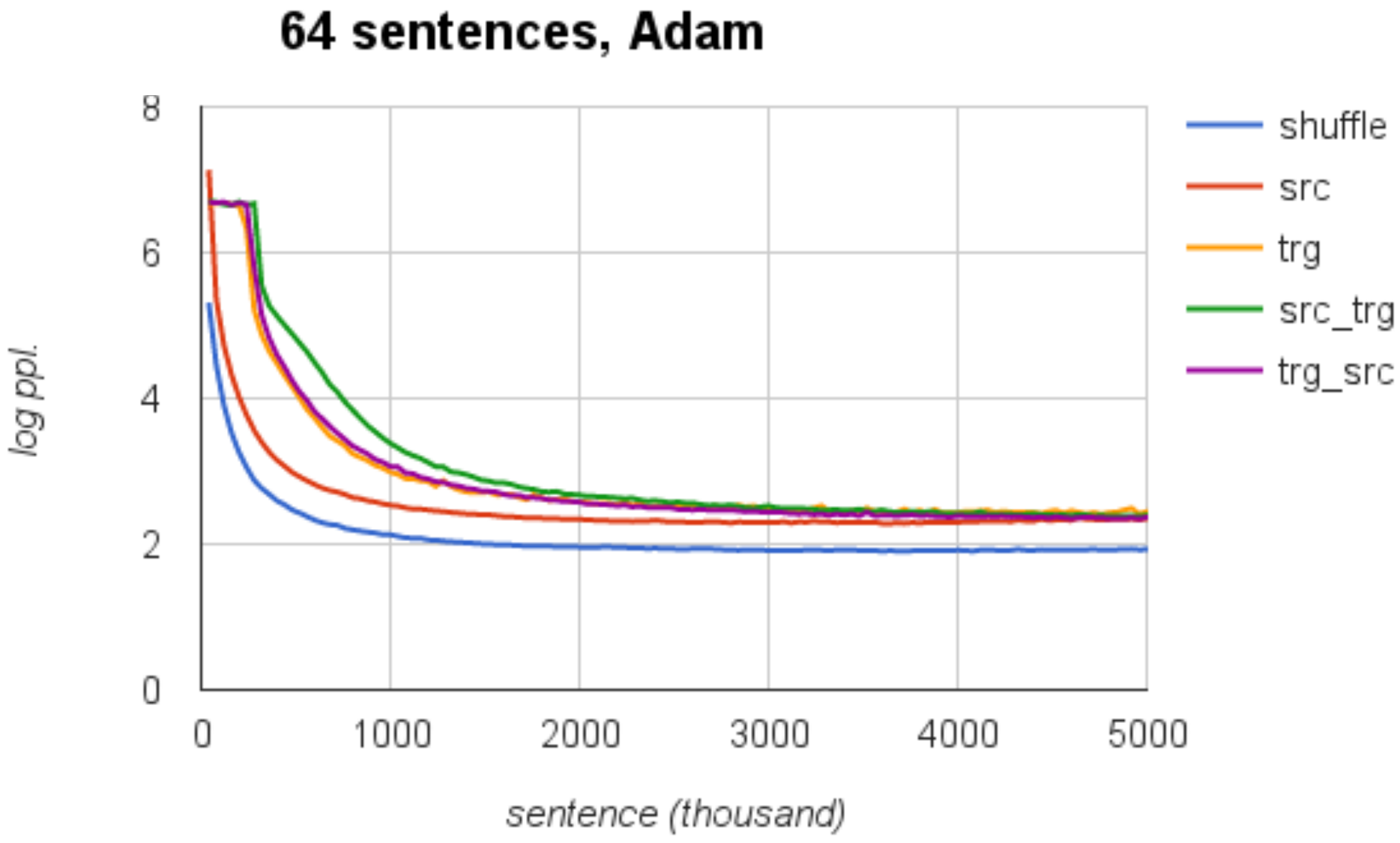
- 検討するソート手法
  - src_trg: 原言語文長に従いコーパスをソート
    文長が同じ場合は目的言語文長に従いソート
  - trg_src: 上記の逆



実験設定

- 使用コーパス: ASPEC (英日) 200万文
- 単語分割: Moses tokenizer, KyTea
- NMTモデル: Luong et al., 2015
  + Global attention
  + Input feeding
  + Bi-directional encoder
  + dropout (30%)
- 語彙数: 65536
- LSTM Unit数: 各layer 512
- 学習アルゴリズム: Adam(α=0.001)
- 初期値は固定

影響: コーパスのソート手法 (Adam)



- shuffle (青線) がPerplexityが低い
  - これまでの結果とは真逆に

影響: コーパスのソート手法 (Adam)

- shuffle, srcが良い理由

  - trgでソート→目的言語文のパディング回数が減る

  → 文末を表すトークンの数が減る

  → 推定精度の低下?

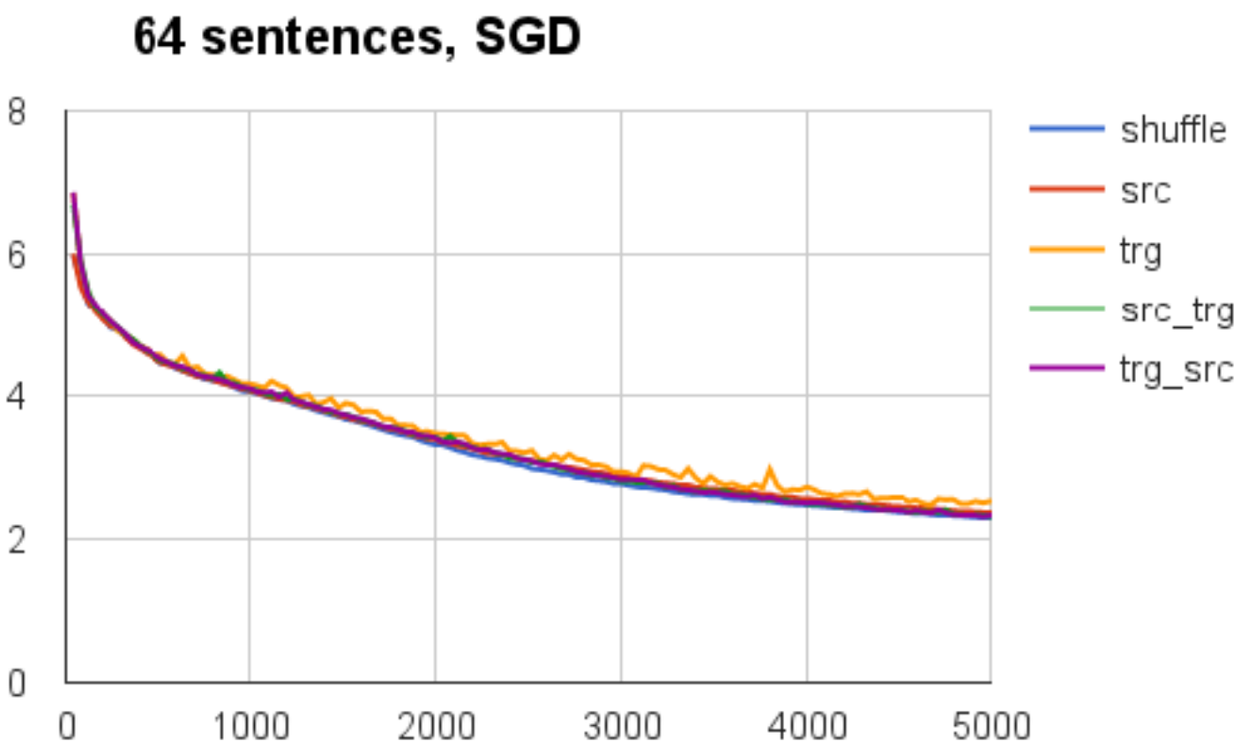
- ソートすると良くない?

  - 文長が類似している文 → 文の特徴も類似?

  - ソートすることで1つのミニバッチに

  類似した文が集まり局所解に?

影響: コーパスのソート手法 (SGD)



ソート手法	平均必要時間 (hour)
shuffle	8.08
src	6.45
trg	5.21
src_trg	4.35
trg_src	4.30

- SGDではソート手法によらず安定した収束
  - 1回の学習にかかる時間が短いtrg_srcが良い
  - 学習アルゴリズムによって最適なミニバッチ構成法が違う

ミニバッチサイズ

ミニバッチサイズ

- 一つのミニバッチに含まれる文数
- ミニバッチサイズ→大
  - より多くのデータを用いて平均lossを計算
    - 勾配更新の安定化
    - 計算効率向上

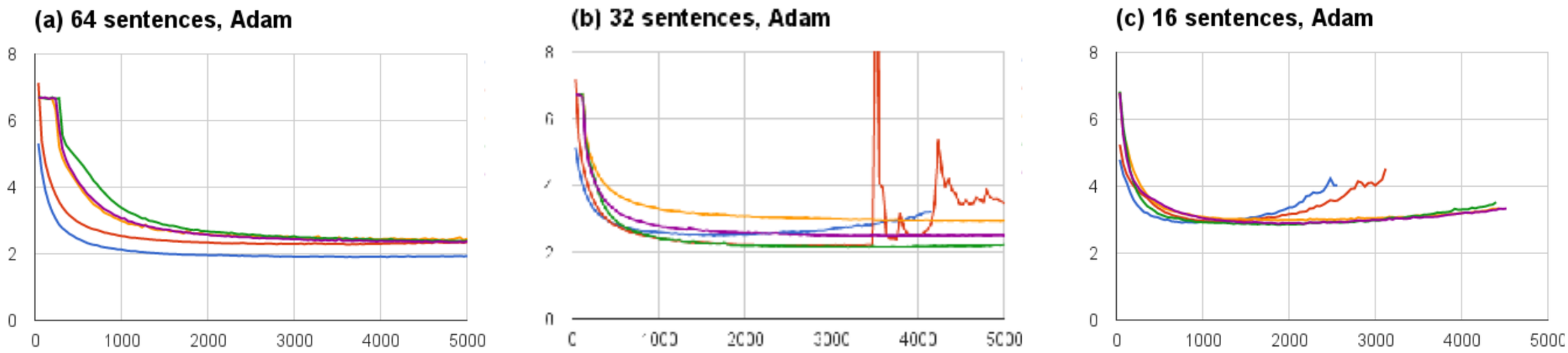
実験設定

- 比較する実験設定

	ミニバッチ構成法	学習アルゴリズム
(a)	64 sentences	Adam
(b)	32 sentences	Adam
(c)	16 sentences	Adam
(d)	8 sentences	Adam

- それぞれで，全てのソート手法を比較する

影響: ミニバッチサイズ



- ミニバッチサイズ大→低Perplexity
  - ミニバッチは大きい方が良い？

ミニバッチサイズの単位

ミニバッチサイズの単位

- ミニバッチサイズを決定する単位
- 本研究で検討する単位
  - 含まれる文数
  - 含まれる目的言語単語数

ミニバッチサイズの単位

- **文数**

  - 最も標準的

  - 現状のツールの大半はこれを使用

- **想定されるデメリット**

  - 各ミニバッチでloss計算回数が一定でない

    (lossの計算回数 = 目的言語単語数)

  - ミニバッチ間でlossの値がばらつく

  - AdamなどMomentumを使用する

    学習アルゴリズム使用時に悪影響が出る？

ミニバッチサイズの単位

- **目的言語文の単語数**

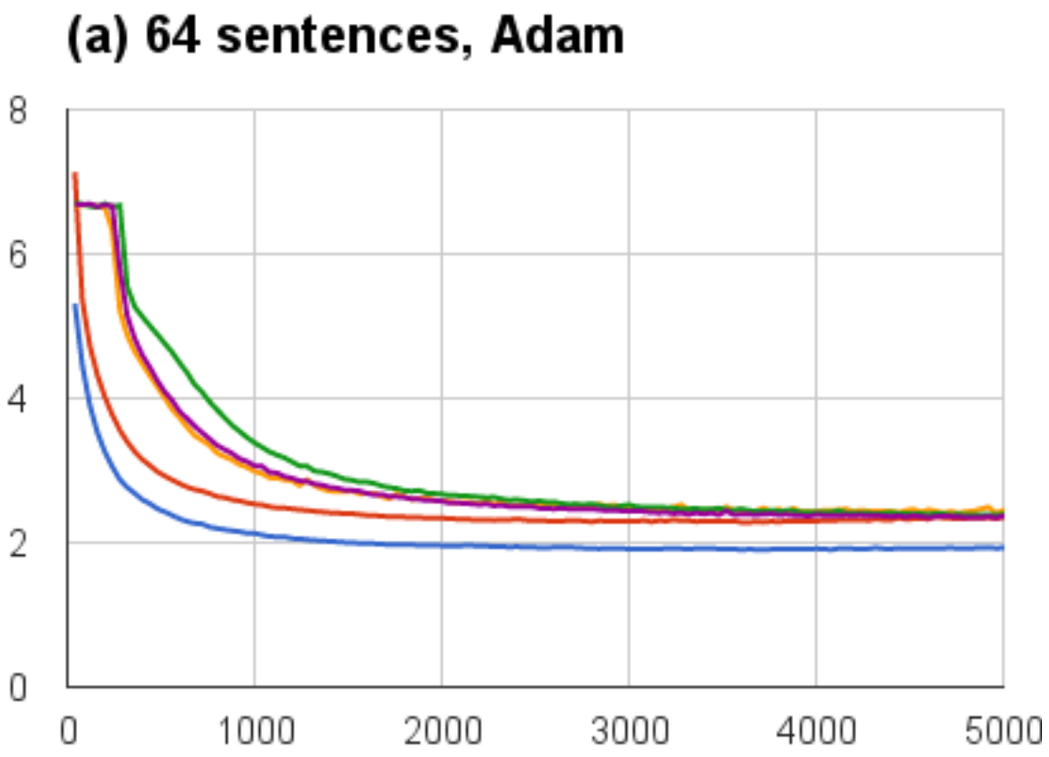
  - 使用しているツールは非常に少ない

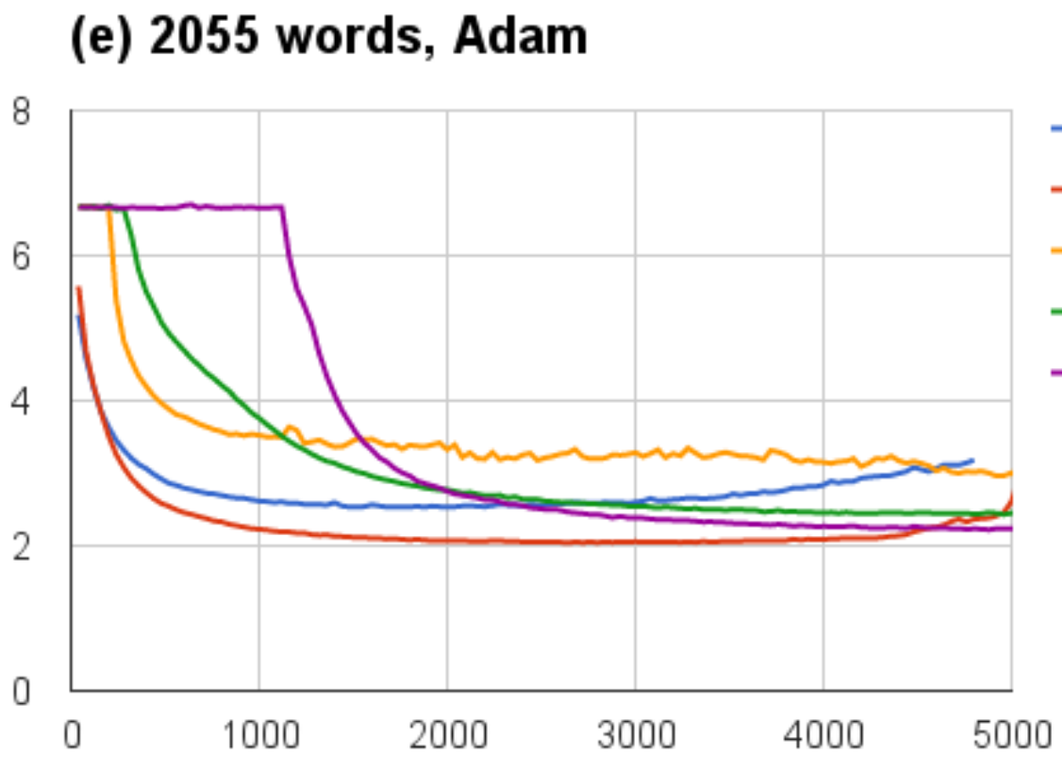
- **想定されるメリット**

  - 各ミニバッチでloss計算回数が一定になる

  - 使用するメモリ量が一定になる

影響: ミニバッチサイズの単位



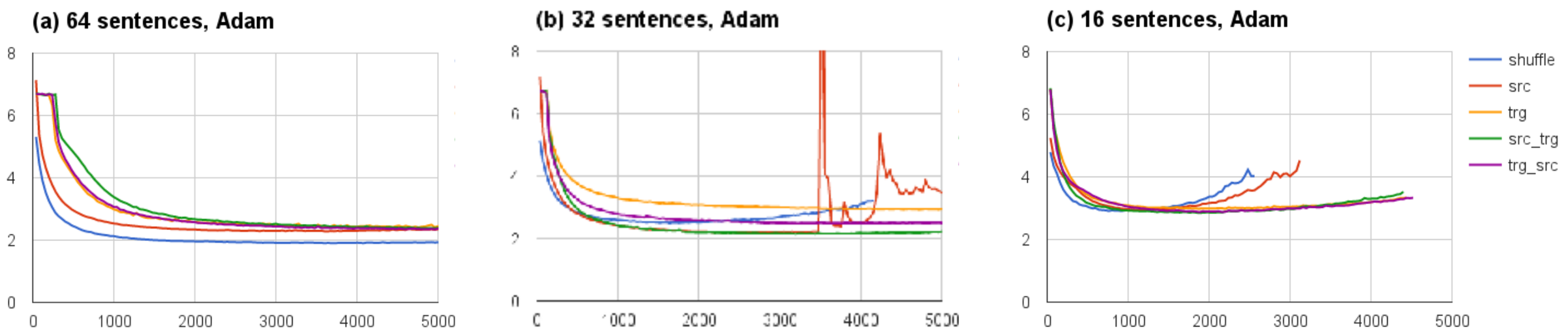


- ソート手法の差はあるものの,
  (a)のshuffle (青線) と (b)のsrc (赤線) はほぼ同じ挙動
- ミニバッチサイズの単位は速度および精度に
  大きな影響をおよぼさない

まとめ

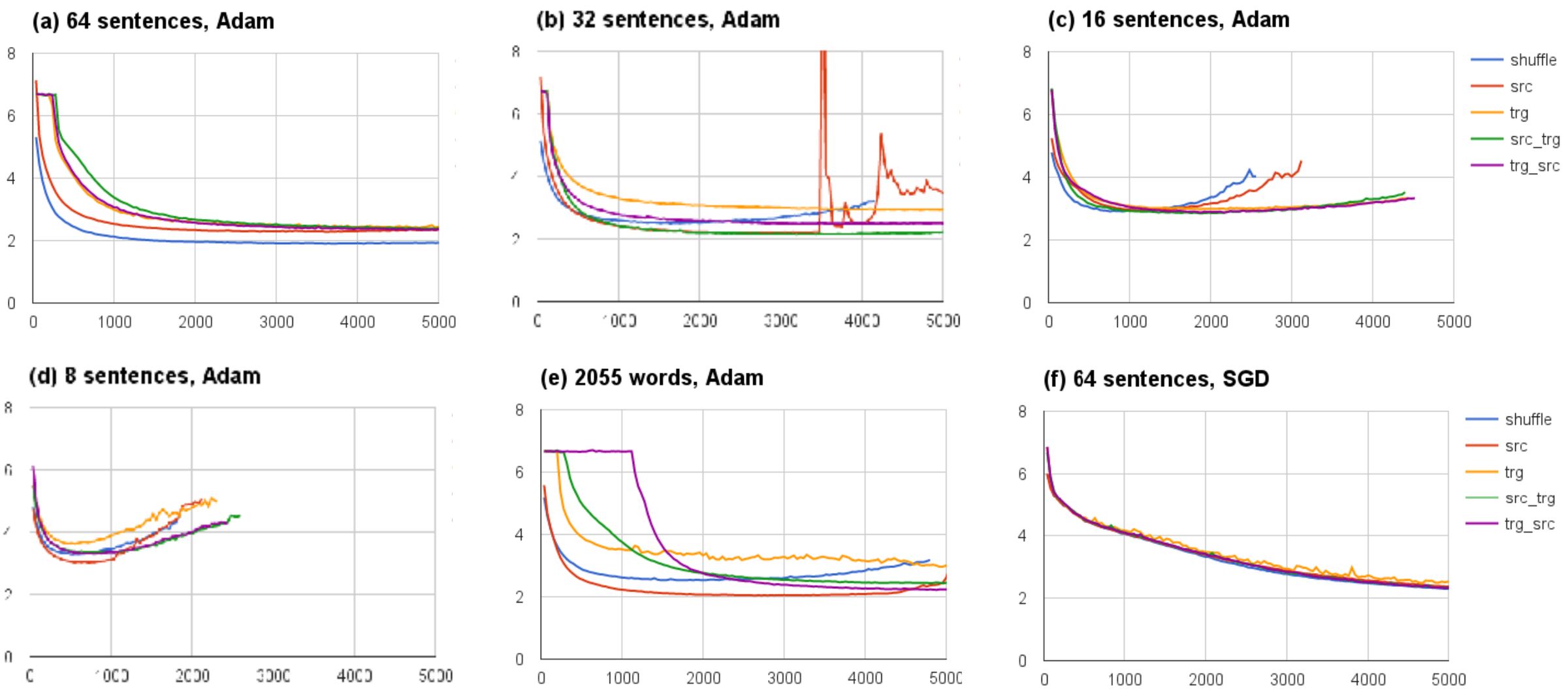
- ミニバッチの構成法は学習に大きく影響を与える
  - **学習時にはミニバッチの構成法も考慮すべき**
- **ソート手法**
  - Adam: shuffle, src がおすすめ
  - SGD: trg_src がおすすめ
- **ミニバッチサイズ**
  - 処理速度だけでなく，精度にも影響する可能性
- **今後の課題**
  - 他のコーパス，学習アルゴリズムでの比較

影響: コーパスのソート手法 (Adam)



- 全ての実験の共通点
  - shuffle (青線) または src (赤線) がPerplexityが低い

TestセットのPerplexity



- 縦軸: log perplexity
- 横軸: 学習文数

TestセットのBLEU