Dynamic Programming Encoding for Subword Segmentation in Neural Machine Translation

(He et al., ACL 2020)

出口 祥之

deguchi@ai.cs.ehime-u.ac.jp

2020/07/30 二宮研 論文輪読会

Links

Paper

https://www.aclweb.org/anthology/D19-1098/

Source Code

https://github.com/xlhex/dpe

Introduction

NMT におけるサブワード分割

貪欲法: バイトペア符号化 (BPE)[1], 最長一致法[2]

確率的アルゴリズム: ユニグラムLM[3], BPE-dropout[4]

原言語側，目的言語側ともに複数分割候補が得られる．訓練時に分割候補を確率的にサンプリングすることでモデルの頑健性向上．

動的計画法: 提案手法．サブワード分割の周辺化．

---

[1] Sennrich, Haddow, and Birch, ``Neural Machine Translation of Rare Words with Subword Units".

[2] Wu et al., *Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation*.

[3] Kudo, ``Subword Regularization: Improving Neural Network Translation Models with Multiple Subword Candidates".

[4] Provilkov, Emelianenko, and Voita, ``BPE-Dropout: Simple and Effective Subword Regularization".

Related Work (Greedy Algorithms)

BPE, Wordpiece

- unconscious $\rightarrow$ un + conscious, likes $\rightarrow$ like + s
- 隣り合った頻出サブワードから順に，予め指定した語彙数に到達するまで再帰的に結合 (BPE)
- 語彙数とデコード速度はトレードオフ
  - (語彙数を小さくするだけであれば文字単位でよい)
  - テキスト圧縮の技術を利用
  - 語彙数の上限を制約とし，文長が短くなるような分割を得るアルゴリズム

Related Work (Stochastic Algorithms)

ユニグラムLM, BPE-dropout

- unconscious $\rightarrow$ {un + concious, uncon + scious}
- 複数分割候補を得られる
  - ユニグラムLM: 尤度ベースでサンプリング
  - BPE-dropout: 結合時に確率的に棄却
  - NMT 訓練時に分割を確率的に得ることでデータ拡張 (Data Augumentation) の効果
    - モデルの頑健性，汎用化

Related Work (Dynamic Programming Algorithms)

音声認識[5]

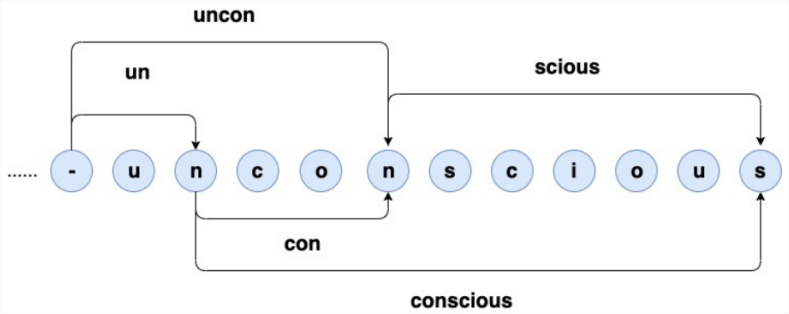
非自己回帰 NMT モデル[6][7]

[5] Wang et al., *Sequence Modeling via Segmentations*.

[6] Chan et al., *Imputer: Sequence Modelling via Imputation and Dynamic Programming*.

[7] Saharia et al., *Non-Autoregressive Machine Translation with Latent Alignments*.

Latent Subword Segmentation - Definitions



目的言語側の分割を潜在変数とみなす

- M 個のサブワード境界: $\{\boldsymbol{y}_{z_i, z_{i+1}}\}_{i=1}^{M}$
  - $\boldsymbol{y} = (y_1, \ldots, y_T)$: 目的言語文の文字列
  - $\boldsymbol{z} = (z_1(=0), \ldots, z_{M+1}(=T))$: 文字インデックス列
  - $\boldsymbol{y}_{a,b}$: $(a+1)^{\text{th}}$ から b^{th} まで結合したサブワード

例:

- 辞書 $\mathcal{V} = \{$'c', 'a', 't', 'ca', 'at'$\}$
- 単語: 'cat'

$\boldsymbol{z}$	サブワード列
$(0, 1, 3)$	('c', 'at')
$(0, 2, 3)$	('ca', 't')
$(0, 1, 2, 3)$	('c', 'a', 't')

Latent Subword Segmentation - Likelihood

連鎖律を用いてサブワード列の対数尤度を表現

- 各サブワードにおいて語彙のカテゴリ分布を生成

$$\log p(\boldsymbol{y}, \boldsymbol{z}|\boldsymbol{x}) = \sum_{i=1}^{|\boldsymbol{z}|} \log p(\boldsymbol{y}_{z_i,z_{i+1}}|\boldsymbol{y}_{z_1,z_2}, \ldots, \boldsymbol{y}_{z_{i-1},z_i}, \boldsymbol{x})$$

※ $\boldsymbol{x}$: 原言語文

- 殆どの NMT は $\boldsymbol{z}$ は $\boldsymbol{y}$ の決定論的関数とみなされる:

$$\log p(\boldsymbol{y}, \boldsymbol{z}) \approx \log p(\boldsymbol{y})$$

Latent Subword Segmentation - Latent Variable

$\boldsymbol{z} \in \mathcal{Z}_y$($\boldsymbol{y}$の分割集合) を潜在表現とみなす

- $p(\boldsymbol{y}|\boldsymbol{x}) = \sum_{\boldsymbol{z}} p(\boldsymbol{y}, \boldsymbol{z}|\boldsymbol{x})$ とする

$$\log p(\boldsymbol{y}|\boldsymbol{x}) = \log \sum_{\boldsymbol{z} \in \mathcal{Z}_y} \exp \sum_{i=1}^{|\boldsymbol{z}|} \log p(\boldsymbol{y}_{z_i, z_{i+1}} | \boldsymbol{y}_{z_1, z_2}, \ldots, \boldsymbol{y}_{z_{i-1}, z_i}, \boldsymbol{x})$$

※ 対数周辺尤度の下限: $\log p(\boldsymbol{y}|\boldsymbol{x}) \geq \log p(\boldsymbol{y}, \boldsymbol{z}|\boldsymbol{x})$

- 各サブワードの確率が条件部のコンテキストの分割に依存するため，巨大な空間 $\mathcal{Z}_y$ 上での厳密な周辺化は組み合わせ爆発を起こす

A Mixed Character-Subword Transformer

文字に基づいてサブワードを生成する Transformer

- 条件部のコンテキストを文字のみに

$$\log p(\boldsymbol{y}, \boldsymbol{z}|\boldsymbol{x}) = \sum_{i=1}^{|\boldsymbol{z}|} \log p(\boldsymbol{y}_{z_i, z_{i+1}}|y_{z_1}, \ldots, y_{z_i}, \boldsymbol{x})$$

- $\boldsymbol{y}$ の各文字位置 t において，次に来るサブワード $w \in \mathcal{V}$ の分布を以下に基づいて生成

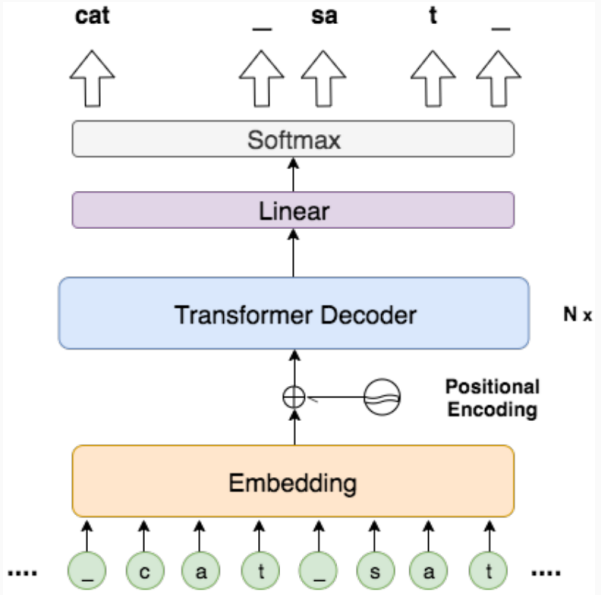
$$p(w|y_1, \ldots, y_t, \boldsymbol{x}) = \frac{\exp(f(y_1, \ldots, y_t)^\top e(w))}{\sum_{w' \in \mathcal{V}} \exp(f(y_1, \ldots, y_t)^\top e(w'))}$$

  - $f(\cdot)$: Transformer により条件部の計算
  - $e(\cdot)$: ソフトマックス層の重み

A Mixed Character-Subword Transformer

t ステップ目のモデル出力

1. t ステップ目でサブワード w を生成
2. サブワード w の文字をデコーダに入力 ($t+1$ から $t+|w|$ まで)
3. $t+|w|$ ステップ目で次のサブワードを生成



Optimization

目的関数 $\mathcal{L}(\theta)$ を最大化

$$\mathcal{L}(\theta) = \sum_{(\boldsymbol{x},\boldsymbol{y})\in\mathcal{D}} \log P(\boldsymbol{y}|\boldsymbol{x})$$

- 周辺化と対数周辺尤度の勾配計算が必要

Exact Maginalization

動的計画法を用いて周辺尤度を計算

- サブワードの出力確率が文字のみによって得られるため動的計画法によって対数周辺尤度が計算可能

Algorithm 1 Dynamic Programming (DP) for Exact Marginalization

Input: $\boldsymbol{y}$ is a sequence of T characters, V is a subword vocabulary, m is the maximum subword length
Output: $\log p(\boldsymbol{y})$ marginalizing out different subword segmentations.

$\alpha_0 \leftarrow 0$
for $k = 1$ **to** T **do**
  $\alpha_k \leftarrow \log \sum_{j=k-m}^{k-1} \mathbb{1}[\boldsymbol{y}_{j,k} \in V] \exp\left(\alpha_j + \log P_\theta(\boldsymbol{y}_{j,k}|y_1, .., y_j)\right)$
end for
return α_T ▷ the marginal probability $\log p(\boldsymbol{y}) = \log \sum_{\boldsymbol{z} \in \mathcal{Z}_y} p(\boldsymbol{y}, \boldsymbol{z})$

- 計算量: $\mathcal{O}(mT)$
  - m : 語彙に含まれる最長の単語の文字数

Gradient Computation

計算量増加への対処

- PyTorch での著者実装で通常の Transformer デコーダより 8 倍遅く，メモリ使用量も増加
  - DP アルゴリズムと文字レベルでの演算による系列長の増加が原因
- Transformer のレイヤ数を 6 から 4 に減らし，16 ステップ勾配蓄積 (Gradient Accumulattion) してからパラメタ更新

Segmenting Target Sentences

Dynamic Programming Encoding (DPE): 最大事後確率を持つ目的言語文の分割を探索

Algorithm 2 Dynamic Programming Encoding (DPE) for Subword Segmentation

Input: $\boldsymbol{y}$ is a sequence of T characters, V is a subword vocabulary, m is the maximum subword length
Output: Segmentation $\boldsymbol{z}$ with highest posterior probability.

for $k = 1$ **to** T **do**

$$\beta_k \leftarrow \max_{\{j \in [k-m,k-1] \,|\, \boldsymbol{y}_{j,k} \in V\}} \beta_j + \log P_\theta(\boldsymbol{y}_{j,k} | y_1, .., y_j)$$

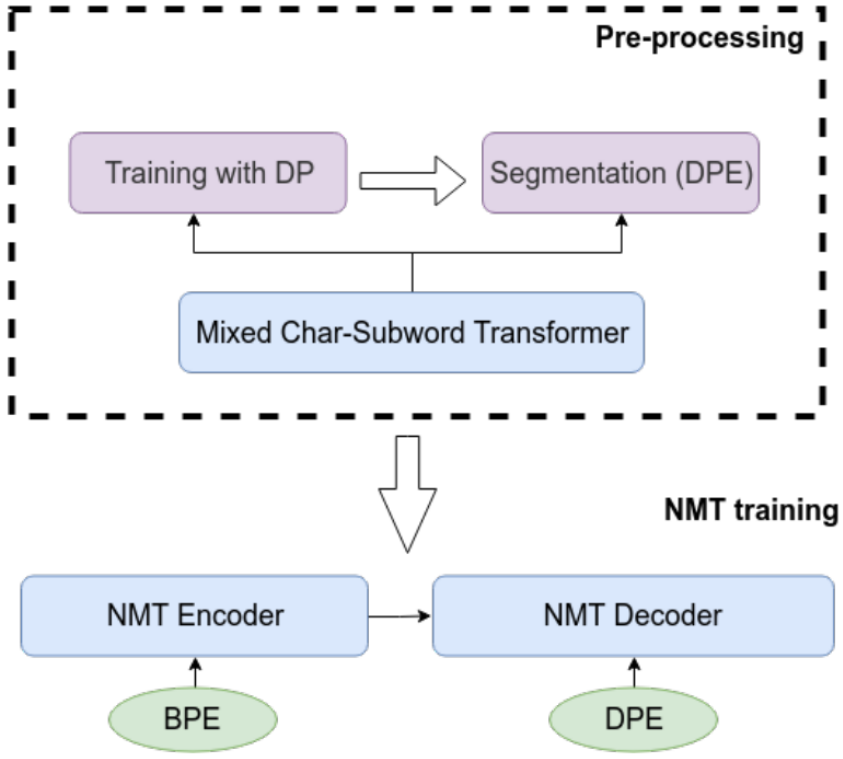
$$b_k \leftarrow \operatorname{argmax}_{\{j \in [k-m,k-1] \,|\, \boldsymbol{y}_{j,k} \in V\}} \beta_j + \log P_\theta(\boldsymbol{y}_{j,k} | y_1, .., y_j)$$

end for

$\boldsymbol{z} \leftarrow \text{backtrace}(b_1, .., b_T)$ ▷ backtrace the best segmentation using $\boldsymbol{b}$

Segmenting Target Sentences

- 混合文字サブワード Transformer は訓練データの目的言語文の分割のためのみに使用
- 分割した文で通常のサブワード Transformer を訓練



Experiments

データセット WMT09 En-Hu, WMT14 En-De, WMT15 En-Fi, WMT16 En-Ro, WMT18 En-Et

モデル

アーキテクチャ	Transformer base
分割 (原言語側)	BPE-dropout $(p = 0.05)$
(目的言語側)	DPE

Main Results

Method	BPE	BPE dropout	Δ_1	This paper	Δ_2
Source segmentation	BPE	BPE dropout		BPE dropout	
Target segmentation	BPE	BPE dropout		DPE	
En→De	27.11	27.27	+0.16	27.61	+0.34
En→Ro	27.90	28.07	+0.17	28.66	+0.59
En→Et	17.64	18.20	+0.56	18.80	+0.60
En→Fi	15.88	16.18	+0.30	16.89	+0.71
En→Hu	12.80	12.94	+0.14	13.36	+0.42
De→En	30.82	30.85	+0.03	31.21	+0.36
Ro→En	31.67	32.56	+0.89	32.99	+0.43
Et→En	23.13	23.65	+0.52	24.62	+0.97
Fi→En	19.10	19.34	+0.24	19.87	+0.53
Hu→En	16.14	16.61	+0.47	17.05	+0.44
Average	22.22	22.57	+0.35	23.12	+0.55

Segmentation Examples

BPE source:
Die G@@ le@@ is@@ anlage war so ausgestattet , dass dort elektr@@ isch betrie@@ bene Wagen eingesetzt werden konnten .
DPE target:
The railway system was equipped in such a way that electrical@@ ly powered cart@@ s could be used on it .
BPE target:
The railway system was equipped in such a way that elect@@ r@@ ically powered car@@ ts could be used on it .

BPE source:
Normalerweise wird Kok@@ ain in kleineren Mengen und nicht durch Tunnel geschm@@ ug@@ gelt .
DPE target:
Normal@@ ly c@@ oca@@ ine is sm@@ ugg@@ led in smaller quantities and not through tunnel@@ s .
BPE target:
Norm@@ ally co@@ c@@ aine is sm@@ ugg@@ led in smaller quantities and not through tun@@ nels .

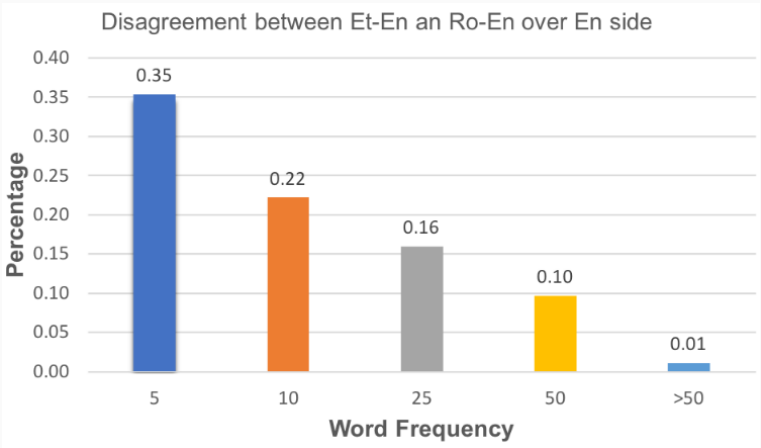
- 他の例は論文参照

Conditional Subword Segmentation

原言語文を条件部に入れず，LM で分割

Source Target	BPE drop BPE drop	BPE drop LM DPE	BPE drop DPE
En→Ro	28.07	28.07	28.66
En→Hu	12.94	12.87	13.36
Ro→En	32.56	32.57	32.99
Hu→En	16.61	16.41	17.05

同一の目的言語文で原言語側を変えて違いを比較



Conditional Subword Segmentation

原言語文が BPE-dropout によって変化することの有効性

Source Target	BPE drop DPE Fixed	BPE drop DPE On The Fly
En→Ro	28.58	28.66
En→Hu	13.14	13.36
En→Et	18.51	18.80
Ro→En	32.73	32.99
Hu→En	16.82	17.05
Et→En	24.37	24.62

DPE vs BPE

目的言語側の分割アルゴリズムを変えて比較

Source Target	BPE drop BPE	BPE drop BPE drop	BPE drop DPE
En→Ro	28.04	28.07	28.66
En→Et	18.09	18.20	18.80
Ro→En	32.40	32.56	32.99
Et→En	23.52	23.65	24.62

Conclusion

- **Dynamic Programming Encoding** を提案
  - 訓練時は目的言語側の分割を潜在変数とみなして周辺化
  - 推論時は事後確率が最も高くなる分割を出力
- BPE だけでなく BPE-dropout と比較しても翻訳性能が向上