ALBERT: A LITE BERT

and@OSK dl-study

2020/06/27

概要

自然言語処理分野の論文

BERT: Bidirectional Encoder Representations from Transformers をもとにパラメータ削減の方法を主張

→①メモリー制限に達しづらくなる

②学習スピードが速くなる

という恩恵がある

研究背景

- 自然言語処理において、大規模なモデルほど性能が向上されることが示された（具体的には、3つの自然言語処理タスクにおいて「隠れ層を大きくすること」「隠れ層を増やすこと」「attention headを増やすこと」はより良い性能につながることが示された）

関連研究

- Model parallelization(2019)
- Clever memory management(2016,2017)

Communication overheadに着目したのはALBERTが初めて

Communication Overheadとは:

The total number of packets are to be transferred or transmitted from one node to another is known as the **communication overhead**.

成果

精度をそこまで損なわないまま

①BERTより18倍少ないパラメータ数

②BERTより1.7倍速い学習時間

また、パラメータ削減したことによってBERTより大規模なモデルを作ることができ、その大規模なモデル（ALBERTxxlarge）は精度がBERTより良い

BERTとは？

2018年発表
汎用性が高いし精度が良いので画期的
事前学習→特定のタスクにfine tuningする

masked language modeling(MLM)
next-sentence prediction (NSP)を使って事前学習

BERT 手法

MLM

15%のトークン（単語）が[Mask]トークンで置き換えられる。
[Mask]トークンの元のトークンを予測する

NSP

2つの文章を見て連続している文章か予測する

ALBERTで紹介されている技術

パラメータ削減のための2つのテクニックを提唱

①a factorized embedding parameterization
　　-embedding行列の因数分解
②cross-layer parameter sharing
　　-レイヤー間パラメータ共有

①a factorized embedding parameterization

embedding行列の因数分解

（参考）embeddingとは→
http://yagami12.hatenablog.com/entry/2017/12/30/175113

[apple, orange, banana]

"apple" = [1, 0, 0]
"banana" = [0, 0, 1]

①a factorized embedding parameterization

One-hotベクトルから分散表現に変換するときの処理を
Embedding → Embedding + Linear と変更

これにより必要パラメータは以下のように変化

V x E → V x E + E x H

V：語彙数
E：埋め込みサイズ
H：隠れサイズ

$$V = 32000, H = 1024, E = 128$$
$$V * H = 32,768,000$$
$$V * E + E * H = 4,227,072$$

①a factorized embedding parameterization

なお、embeddingにはWord Piece Embeddingを使用している

- **Word**: Jet makers feud over seat width with big orders at stake
- **wordpieces**: _J et _makers _fe ud _over _seat _width _with _big _orders _at _stake

Jet と feud が sub-word units = Word Pieceに分割されている
※本論文の議論対象ではない

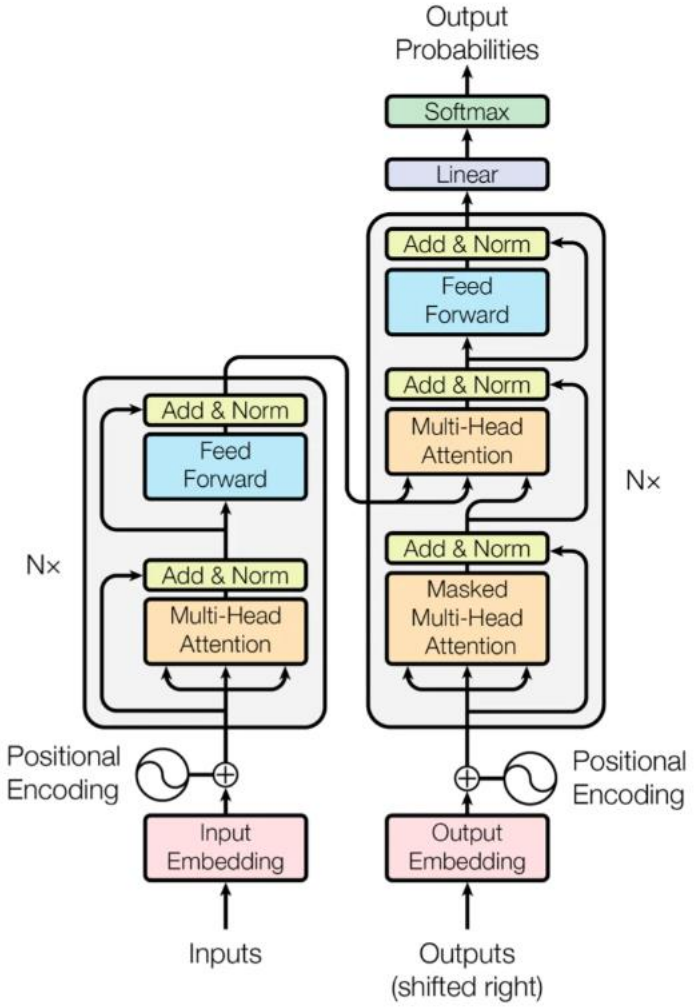
②cross-layer parameter sharing

レイヤー間パラメータ共有

レイヤー間で全てのパラメータを共有する。

・Feed-forward network parameter

・Attention parameter



②cross-layer parameter sharing

- Attention Parameter

https://medium.com/lsc-psd/%E8%87%AA%E7%84%B6%E8%A8%80%E8%AA%9E%E5%87%A6%E7%90%86%E3%81%AE%E5%B7%A8%E7%8D%A3-transformer-%E3%81%AEself-attention-layer%E7%B4%B9%E4%BB%8B-a04dc999efc5

・Feed-forward network（FFN） parameter

$$\mathrm{FFN}(x) = \max(0, xW_1 + b_1)W_2 + b_2$$

Inter-sentence coherence loss

パラメータ削減の手法ではなく、精度向上のための手法
先行研究で行われた意思決定に基づく

事前学習に使うタスクを以下のように変化

MLM&NSP → MLM&SOP

MLM：masked language modeling
NSP：next-sentence prediction
SOP：sentence-order prediction

Inter-sentence coherence loss

SOP

2つの文章の並び順が正しいか予測する

※NSPを使わない理由は以下のように考えられている。

・簡単

・トピック予測と一貫性予測を行うが、そのトピック予測の部分がMLMでやってることと重複する

MODEL SETUP

- BOOK CORPUS
- English Wikipedia を使用

Model		Parameters	Layers	Hidden	Embedding	Parameter-sharing
BERT	base	108M	12	768	768	False
	large	334M	24	1024	1024	False
ALBERT	base	12M	12	768	128	True
	large	18M	24	1024	128	True
	xlarge	60M	24	2048	128	True
	xxlarge	235M	12	4096	128	True

結果が似ていたため12層にしたそう。

評価

Model		Parameters	SQuAD1.1	SQuAD2.0	MNLI	SST-2	RACE	Avg	Speedup
BERT	base	108M	90.4/83.2	80.4/77.6	84.5	92.8	68.2	82.3	4.7x
	large	334M	92.2/85.5	85.0/82.2	86.6	93.0	73.9	85.2	1.0
ALBERT	base	12M	89.3/82.3	80.0/77.1	81.6	90.3	64.0	80.1	5.6x
	large	18M	90.6/83.9	82.3/79.4	83.5	91.7	68.5	82.4	1.7x
	xlarge	60M	92.5/86.1	86.1/83.1	86.4	92.4	74.8	85.5	0.6x
	xxlarge	235M	**94.1/88.3**	**88.1/85.1**	**88.0**	**95.2**	**82.3**	**88.7**	0.3x

・base同士,large同士を比べると精度はちょっと落ちてる。
・パラメータが超少ない
・ALBERTxxlargeは精度一番良い

実験-埋め込みサイズについて

V x E + E x H (V：語彙数 E：埋め込みサイズ H：隠れサイズ)
において、Eを変えて実験

Model	E	Parameters	SQuAD1.1	SQuAD2.0	MNLI	SST-2	RACE	Avg
ALBERT base not-shared	64	87M	89.9/82.9	80.1/77.8	82.9	91.5	66.7	81.3
	128	89M	89.9/82.8	80.3/77.3	83.7	91.5	67.9	81.7
	256	93M	90.2/83.2	80.3/77.4	84.1	91.9	67.3	81.8
	768	108M	90.4/83.2	80.4/77.6	84.5	92.8	68.2	82.3
ALBERT base all-shared	64	10M	88.7/81.4	77.5/74.8	80.8	89.4	63.5	79.0
	128	12M	89.3/82.3	80.0/77.1	81.6	90.3	64.0	80.1
	256	16M	88.8/81.5	79.1/76.3	81.5	90.3	63.4	79.6
	768	31M	88.6/81.5	79.2/76.6	82.0	90.6	63.3	79.8

実験-パラメータ共有について

Model		Parameters	SQuAD1.1	SQuAD2.0	MNLI	SST-2	RACE	Avg
ALBERT base E=768	all-shared	31M	88.6/81.5	79.2/76.6	82.0	90.6	63.3	79.8
	shared-attention	83M	89.9/82.7	80.0/77.2	84.0	91.4	67.7	81.6
	shared-FFN	57M	89.2/82.1	78.2/75.4	81.5	90.8	62.6	79.5
	not-shared	108M	90.4/83.2	80.4/77.6	84.5	92.8	68.2	82.3
ALBERT base E=128	all-shared	12M	89.3/82.3	80.0/77.1	82.0	90.3	64.0	80.1
	shared-attention	64M	89.9/82.8	80.7/77.9	83.4	91.9	67.6	81.7
	shared-FFN	38M	88.9/81.6	78.6/75.6	82.3	91.7	64.4	80.2
	not-shared	89M	89.9/82.8	80.3/77.3	83.2	91.5	67.9	81.6

・not-sharedがパラメータ共有をしないモデル
・attentionパラメータの共有は影響少ない
・FFNパラメータの共有はパフォーマンスに影響
・いろいろ考慮して論文はall-shared戦略をdefaultに決定

実験-SOP vs NSP

	Intrinsic Tasks			Downstream Tasks					
SP tasks	MLM	NSP	SOP	SQuAD1.1	SQuAD2.0	MNLI	SST-2	RACE	Avg
None	54.9	52.4	53.3	88.6/81.5	78.1/75.3	81.5	89.9	61.7	79.0
NSP	54.5	90.5	52.0	88.4/81.5	77.2/74.6	81.6	**91.1**	62.3	79.2
SOP	54.0	78.9	86.5	**89.3/82.3**	**80.0/77.1**	**82.0**	90.3	**64.0**	**80.1**

・SST-2(映画レビューのpositive/negative二値分類)だけNSPが精度高い

・全体的にSOPが精度高い

実験-学習時間を同じにする

- 学習長いほど精度向上する
- 図0で、ALBERT-xxlargeは最も学習時間遅かった

Models	Steps	Time	SQuAD1.1	SQuAD2.0	MNLI	SST-2	RACE	Avg
BERT-large	400k	34h	93.5/87.4	86.9/84.3	87.8	94.6	77.3	87.2
ALBERT-xxlarge	125k	32h	**94.0/88.1**	**88.3/85.3**	87.8	**95.4**	**82.5**	**88.7**

・ほぼ同じ学習時間にしてBERTlargeとALBERTxxlargeを比較

・ALBERTxxlargeのほうが精度高い

・パラメータが少ないだけでなく精度の面でも上といえる

パラメータ数：BERTlarge→334M, ALBERTxxlarge→235M

今後の課題

- ALBERTxxlargeはパラメータは少なくなったものの、単純に構造が大きいので学習/推論スピードが遅い
- 事前学習に使うタスクはMLM&SOPが最適とは限らない