Improving Language Understanding by Generative Pre-Training

HW7

ELIJAH RIPPETH AND DANIEL CAMPOS

05/20/2019

Background

- The Transformer framework performs highly in most NLP tasks
- Sentence and word representations have brought huge gains to countless NLP tasks
- Semi Supervised Learning has proven effective at leveraging corpus information in downstream tasks
- Unsupervised pre-training proved effective in image classification and regression tasks.
- Auxiliary Training Objectives leverages a wide variety of objectives to improve many disparate tasks.

Problem Statement

- Given a large corpus of unlabeled data can competitive models be trained by building a general model on the larger corpus and fine tuning to the targeted task?
  - Yes!

Related work

- Peters, 2018
  - Deep contextualized word representations (aka ELMo)
- Vaswani, 2017
  - Attention is all you need
- Bowman, 2015
  - A large annotated corpus for learning natural language inference
- Wang, 2018
  - GLUE: A Multi-Task Benchmark and Analysis Platform for Natural Language Understanding
- Lai, 2017
  - RACE: Large-scale ReAding Comprehension Dataset From Examinations

Algorithm: intuition

Unsupervised pre-training

- Leverage a language modeling objective

Supervised fine-tuning

- Adapt parameters to supervised task

Implementation details: pre-training

Given unsupervised corpus U = {u1, …, uN} and context window k, we want to maximize the following log-likelihood:

$$L_1(\mathcal{U}) = \sum_i \log P(u_i | u_{i-k}, \ldots, u_{i-1}; \Theta)$$

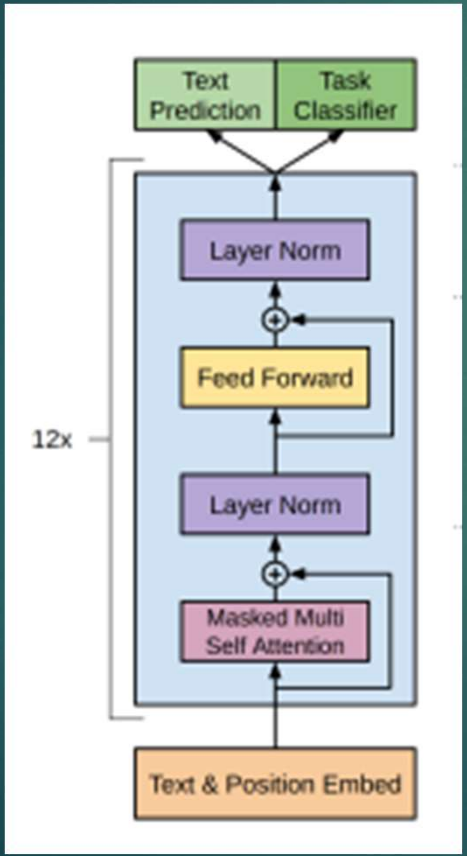
Implementation details: pre-training

Leverages a transformer decoder to model the probability space of inputs tokens

$$h_0 = UW_e + W_p$$
$$h_l = \texttt{transformer\_block}(h_{l-1}) \forall i \in [1, n]$$
$$P(u) = \texttt{softmax}(h_n W_e^T)$$

- We is the token embedding matrix
- Wp is the position embedding matrix
- U is the context vector (length k vector)

Implementation details: pre-training



Implementation details: fine-tuning

Given labeled corpus C where instances are {x1, x2, …, xm}, predict label:

$$P(y|x^1, \ldots, x^m) = \texttt{softmax}(h_l^m W_y).$$

Objective is to maximize log-likelihood L2 (though L3 is used in practice):

$$L_2(\mathcal{C}) = \sum_{(x,y)} \log P(y|x^1, \ldots, x^m).$$

$$L_3(\mathcal{C}) = L_2(\mathcal{C}) + \lambda * L_1(\mathcal{C})$$

Tasks

Task	Datasets
Natural language inference	SNLI, MultiNLI, Question NLI, RTE, SciTail
Question Answering	RACE, Story Cloze
Sentence similarity	MSR Paraphrase Corpus, Quora Question Pairs, STS Benchmark
Classification	Stanford Sentiment Treebank-2, CoLA

Wait... those task inputs don't look reflective?

- Some tasks can be fine-tuned directly by the aforementioned scheme.
  - e.g., seq2seq tasks will predict next word given context
- Others have structured inputs which aren't so clear. What do we do when we *aren't* predicting the next word?
  - e.g., QA: document, question and answer pairs
  - e.g., NLI: premise and hypothesis
- How can we transform the inputs to minimize task-specific customization of inputs?
  - Traversal-style preprocessing (Rocktäschel et al., 2015)

Task-specific input transformations

- NLI
  - Concatenate premise and hypothesis token sequences, delimited by $.
- Similarity
  - Modify the input sequence to contain both possible sentence orderings and process each independently to produce two sequence representations which are added element-wise.
- QA
  - Create a vector of document vector, question vector, a delimiter, and answer vector for each possible answer. Process independently and normalize via softmax for probability of answer correctness.

Evaluation (NLI)

Method	MNLI-m	MNLI-mm	SNLI	SciTail	QNLI	RTE
ESIM + ELMo [44] (5x)	-	-	89.3	-	-	-
CAFE [58] (5x)	80.2	79.0	89.3	-	-	-
Stochastic Answer Network [35] (3x)	80.6	80.1	-	-	-	-
CAFE [58]	78.7	77.9	88.5	83.3		
GenSen [64]	71.4	71.3	-	-	82.3	59.2
Multi-task BiLSTM + Attn [64]	72.2	72.1	-	-	82.1	**61.7**
Finetuned Transformer LM (ours)	**82.1**	**81.4**	**89.9**	**88.3**	**88.1**	56.0

Evaluation (QA)

Method	Story Cloze	RACE-m	RACE-h	RACE
val-LS-skip [55]	76.5	-	-	-
Hidden Coherence Model [7]	77.6	-	-	-
Dynamic Fusion Net [67] (9x)	-	55.6	49.4	51.2
BiAttention MRU [59] (9x)	-	60.2	50.3	53.3
Finetuned Transformer LM (ours)	**86.5**	**62.9**	**57.4**	**59.0**

Evaluation (Classification/Similarity)

Method	Classification		Semantic Similarity			GLUE
	CoLA (mc)	SST2 (acc)	MRPC (F1)	STSB (pc)	QQP (F1)	
Sparse byte mLSTM [16]	-	**93.2**	-	-	-	-
TF-KLD [23]	-	-	**86.0**	-	-	-
ECNU (mixed ensemble) [60]	-	-	-	81.0	-	-
Single-task BiLSTM + ELMo + Attn [64]	35.0	90.2	80.2	55.5	66.1	64.8
Multi-task BiLSTM + ELMo + Attn [64]	18.9	91.6	83.5	72.8	63.3	68.9
Finetuned Transformer LM (ours)	**45.4**	91.3	82.3	**82.0**	**70.3**	**72.8**

Discussion

Impact of number of layers transferred

- 9% gain per layer till 12

Zero shot behavior

- Tasks vary in their pretraining iterations required.
- Transformer learns faster than LSTM.

Ablation

- Larger datasets benefit from auxiliary task (L3 loss function) while smaller do not
- LSTM average drop 5.6% vs. transformer
- No pre-training average 14.8% drop

Language Models are Unsupervised Multitask Learners

Background

- ELMo , GPT, BERT show huge gains on most NLP Tasks
- Fine-tuning is great but can we avoid it by making the LM bigger?
- How useful is unidirectional text representations such as BERT's?
- Are we all Bayesians?
- P(Y | X) for everything!

Updates

- New Dataset
  - No longer Unpublished book dataset
  - 40 million documents linked from reddit with Karma + 3
    - 8 Million unique documents after deduping
    - Remove Wikipedia
- 4 model sizes explored
  - 117M, 345M, 762M, 1542M parameters
  - Smallest is equivalent to GPT, second is Large BERT
- Transformer block have new normalization
- ***NO MORE FINE-TUNING!***

GPT-2 Zero Shot Learning

Language Models are Unsupervised Multitask Learners

	LAMBADA (PPL)	LAMBADA (ACC)	CBT-CN (ACC)	CBT-NE (ACC)	WikiText2 (PPL)	PTB (PPL)	enwik8 (BPB)	text8 (BPC)	WikiText103 (PPL)	1BW (PPL)
SOTA	99.8	59.23	85.7	82.3	39.14	46.54	0.99	1.08	18.3	**21.8**
117M	**35.13**	45.99	**87.65**	**83.4**	**29.41**	65.85	1.16	1.17	37.50	75.20
345M	**15.60**	55.48	**92.35**	**87.1**	**22.76**	47.33	1.01	**1.06**	26.37	55.72
762M	**10.87**	**60.12**	**93.45**	**88.0**	**19.93**	**40.31**	**0.97**	**1.02**	22.05	44.575
1542M	**8.63**	**63.24**	**93.30**	**89.05**	**18.34**	**35.76**	**0.93**	**0.98**	**17.48**	42.16

Hype

SYSTEM PROMPT (HUMAN-WRITTEN)

In a shocking finding, scientist discovered a herd of unicorns living in a remote, previously unexplored valley, in the Andes Mountains. Even more surprising to the researchers was the fact that the unicorns spoke perfect English.

MODEL COMPLETION (MACHINE-WRITTEN, 10 TRIES)

The scientist named the population, after their distinctive horn, Ovid's Unicorn. These four-horned, silver-white unicorns were previously unknown to science.

Now, after almost two centuries, the mystery of what sparked this odd phenomenon is finally solved.

Dr. Jorge Pérez, an evolutionary biologist from the University of La Paz, and several companions, were exploring the Andes Mountains when they found a small valley, with no other animals or humans. Pérez noticed that the valley had what appeared to be a natural fountain, surrounded by two peaks of rock and silver snow.

Pérez and the others then ventured further into the valley. "By the time we reached the top of one peak, the water looked blue, with some crystals on top," said Pérez.

Pérez and his friends were astonished to see the unicorn herd. These creatures could be seen from the air without having to move too much to see them – they were so close they could touch their horns.

Discussion

Does Fine-tuning still make sense?

Can this model be used elsewhere?

- Musenet

Did OpenAI make the right choice to not release the model?

What impact does corpus have on LM model accuracy